A Novel Approach to E-voting with Group Identity-based Identification and Homomorphic Encryption Scheme

Apurva K Vangujar*, Buvana Ganesh, Alia Umrani, and Paolo Palmieri

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

Abstract. This paper presents a novel e-voting scheme that combines Group Identity-based Identification (GIBI) with Homomorphic Encryption (HE) based on the discrete logarithmic assumption. The proposed scheme uses the Schnorr-like GIBI scheme for voter identification and authorization using Zero-Knowledge (ZK) proof to ensure the *anonymity* and *eligibility* of voters. The use of Distributed ElGamal (DE) provides *fairness* and *receipt-freeness*, while the use of partial shares for decryption enables *individual* and *universal verifiability* without the need for a central authority. Voters can maintain *uncoerability* by encrypting their votes and casting ballots based on their own choice. The proposed scheme is secure under various scenarios and *robust* in the Random Oracle (RO) model. The GIBI-HE scheme offers a promising solution for e-voting, providing a sustainable and accessible environment for voters while supports *unreusability* of votes and protecting *privacy* of voter.

Keywords: Identity-based Identification, Group Identity-based Identification, 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 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 *privacy* during e-voting process. E-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 printing, distribution, and staffing costs in a long-term period. Ensuring the accuracy and security of e-voting systems is critical to maintaining trust in the electoral process so that it can be implemented in practice. Therefore, there is a need for research on secure cryptographic e-voting schemes.

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

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 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 (ID) and the ballot of the voter should be protected to provide *privacy* and *anynomity* respectively. In the literature, a combination of signature schemes [27] and encryption algorithms [38] are used to secure the overall e-voting process. However this is not enough for the registration and validation of eligible voters, as can be seen in attacks proposed on e-voting schemes used in practice such as the sVote system in Switzerland used by SwissPost [21]. To tackle this, we propose a solution using Identity-based Identification (IBI) scheme and Homomorphic Encryption (HE).

First proposed by Adi Shamir in 1984, a Standard Identification (SI) scheme [31] is a method for identifying individuals using their unique identities, such as a name, date of birth, or other personal information. SI scheme enables a Prover P to verify their ID to a Verifier V without disclosing any personal information. Boneh and Franklin [6] pioneered the ID-based encryption scheme that led to the flourishing of ID-based cryptography. In later years, Bellare et al. [4] 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 to authenticate the ID of voters and ensure that they are eligible to vote. By satisfying the requirement of *eligibility* and *anonymity*, IBI schemes can enhance the security and integrity of e-voting systems.

HE is a cryptographic technique that allows to perform different operations or computations on encrypted data without decrypting it, it helps to preserve *privacy* of voter and make e-voting system *robust*. Authentication protocols are important for the users to prove their ID to access the encrypted data which is stored in the cloud. In literature today, we do not find many ZK identification protocols for HE schemes, except some focusing on proof of storage as an application [18,3]. We do not find ZK combined with partially HE cryptosystems like ElGamal. The ElGamal cryptosystem [19] 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 satisfies *uncoercibility* requirement. 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 *fairness* to the tallying in e-voting process.

1.1 Motivation

Apart from ElGamal encryption algorithm, several other cryptographic algorithms such as Digital Signature (DS), hash function, and ZK proofs have been used in e-voting scheme, 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 to provide *privacy* for voters in e-voting systems. 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 e-voting system. The initial GS scheme was proposed by Chaum et al. [13] and improved overtime in the universal design proposed by Boneh et al. [7] and Cocks [15]. Canard et al. [8] introduced List signatures are a variant of GS that set a limit on the number of signatures each group member may issue without involving the group manager. The ability of a designated group manager or other authority to reveal the specific ID of a voter could undermine the voting process's integrity. With potentially a large number of voters and ballots [28], this is significant disadvantage and violate the *privacy* of the voter that needs to be addressed.

On the other hand, Group Identity-based Identification (GIBI) scheme by Vangujar et al. [36] do not have this vulnerability, as they do not allow the specific ID of a voter to be revealed by any authority and the signature is verified using a group public key, but the ID of the specific voter remains secret. This can provide *anonymity* and *privacy* for voters, which is the most important in e-voting systems. Additionally, GIBI scheme may be more efficient than GS scheme in terms of the amount of computation required to generate and verify the ID of a voter. Voters can validate their right to vote without revealing their identities 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 ballot. Therefore, we adopt GIBI and ZK-proofs for our proposed scheme.

The Distributed ElGamal (DE) scheme has gained popularity in the literature as a means to handle the confidentiality aspect of e-voting schemes. 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 ID. DE can be easier for voters and election officials to use, as the encryption process is handled automatically by the servers [37]. This can help to simplify the voting process and reduce the burden on voters and election officials. E-voting is a major application for exponential variant of ElGamal besides, the multiple voters use one machine to cast the vote in a voting system. There are several reasons why it will be beneficial to develop a novel e-voting scheme which is a combination of GIBI and HE:

– **Improved security**. The e-voting scheme can provide *privacy* and *anonymity* for voters by using GIBI. This can help safeguard their identities and voting choices throughout the process. By incorporating HE into the e-voting scheme, it becomes possible to perform secure computations on encrypted votes, ensuring the integrity of the voting process. 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**. GIBI scheme 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. A novel e-voting scheme that combines these technologies could be more user-friendly than existing schemes.
- **Confidentiality**. ZK protocols are used in e-voting scheme to ensure the eligibile voter only caste the ballot. 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 DE scheme 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 methods. Any e-voting scheme that combines these algorithms could increase the accessibility of voting.

A novel e-voting scheme combining GIBI and HE can improve security and accessibility while simplifying the voting process for voters and satisfying all the e-voting requirements mentioned in Table 2.

1.2 Related Work

E-voting Models. E-voting schemes, first developed by Chaum [10], have been the topic of an extensive amount of research. There are now three election models used for e-voting and those are: Mixed nets with encryption, signatures, and HE-based models described as follows:

1. In the mix-net model proposed by Chaum [12], different linked servers, referred 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 [23,9,2].
2. The signature-based model involves a 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 [11] presented one of the earliest ideas for the use of Blind Signatures (BS) scheme in e-voting, and BS allows voters to sign their ballots without exposing the content of their votes to the voting authority. The schemes proposed by Rivest et al. [29] and Benaloh [5] are later approaches for adopting BS scheme in e-voting.
3. 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. [38] 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.

ZK Proof. Ateniese et al. [3] introduced the framework for building homomorphic linear authenticators from the identification protocol by Shoup. Zhang et al. [40] feature ZK proof by using ID-based on ElGamal on conic and suggested to eliminate the use of TA. A combination of Non-interactive ZK (NIZK) and exponential ElGamal is provided by Steffan et al. [32] for private smart contracts. We see that ZK can be used in an effective way for verifying voters and checking the validity of ballot. Yang et al. [38] use ZK proofs to demonstrate the verifiability of ballots, the final tally, etc. by utilizing proof of ZK, but do not provide the means to verify ID of the voter before they enter the voting process.
ID-based Schemes. ID-based Cryptosystems (IBC) [31] are mainly aimed at simplifying certificate management and public key revocation issues. IBC associates the user's ID with the public key, therefore, the public key is obtained directly from the user's ID. Zhang et al. [41] presented the ID-BS approach using the IBC concept. Choon and Cheon [14] in 2003 introduced an innovative protocol for ID-BS. Kurosawa and Heng [26] proposed method for transformation DS to IBI scheme. After reviewing all the existing ID-BS schemes for the e-voting system, Vangujar et al. [36] give e-voting as an application of GIBI scheme in their paper. In 2021, once again Vangujar et al. [35] constructed GIBI using the Schnorr IBI [33] and Schnorr DS [24] scheme and it is proven more secure for group-like structure. Malina et al. [27] proposed a GS scheme based approach to e-voting with group manager 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 voter's *eligibility* by default. By providing additional efficiency, an IBI scheme streamlines and enhances the security of the voting process.
Distributed ElGamal. 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 [21], where it helped in encrypting the votes as tuples and scalar multiplication rather than exponentiation. In addition, Haines et al. [20] used codes and ZK to demonstrate the validity of votes using partial codes and Oblivious Randomness (OR) proofs. In their scheme, encryption and decryption keys are distributed to multiple parties, and votes are encrypted using the ElGamal encryption algorithm. The encrypted votes are subsequently sent to a tallying authority, which can count them without decrypting them. Yang et al. [38] proposed rank-based e-voting using DE, where each cast ballot is encrypted and uses the additive homomorphic property of the exponential ElGamal for tallying all votes. They use ZK proof to show the verifiability of ballots and the final tally.

1.3 Contribution

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

- We construct the modified schnorr-like GIBI scheme to use base scheme for our novel GIBI-HE scheme and we show the requirement of ZK proof in GIBI scheme. We also prove modified GIBI scheme is secure under Random Oracle (RO) for malicious as another group member.

- A novel e-voting scheme that uses Schnorr-like GIBI and HE-DE cryptosystem follows the hardness of the DL assumption. GIBI in a revolutionary solution that not only meets the *eligibility* requirements of e-voting but also provides a new and innovative approach to voter identification and authorization. It ensures the *anonymity* and *privacy* of the voters.
- A DL-based ZK proof is developed to authenticate eligible voters and allow them to cast their votes homomorphically encrypted, providing *uncoercibility* and *fairness*.
- A novel scheme that uses DE for tallying all cast ballots and checks the validity of the votes with ZK proof makes it *receipt-free* and overall *robust* e-voting scheme.
- A token is generated and given to the voter who casted the valid ballot, supporting the *unreusability* of the votes. The scheme allows voters to verify their ballot submissions and the final tally while keeping the encrypted ballots and vote counts secure.
- The use of partial shares for decryption makes the system independent of any central authority for vote decryption, ensuring the *individual* and *universal verifiability* in tallying phase. The combination of GIBI, DE, and ZK cryptographic primitives makes the election secure and trustworthy.
- The proposed e-voting scheme is secure in the RO model and efficient compared to existing DS-based e-voting schemes given in Table 4.
- The proposed scheme is proven secure considering different security models for possible scenarios, and proof is given in the RO model.
- We provide suggestions for future work in this area.

The current paper introduces enhanced versions of key algorithms using schnorr key generation method, including the ”ballot encryption algorithm,” ”verification algorithm,” and ”tallying algorithm.” Significant improvements have been made to these algorithms, and additional details regarding these enhancements are provided in this paper.

1.4 Paper Organization

The paper is structured as follows. Section 2 introduces the mathematical notations and primitives used in the rest of the paper. Section 3 gives the construction and security analysis of the modified GIBI scheme from Schnorr DS scheme and also provides the requirement of the ZK proof in GIBI scheme. Section 4 discusses the requirements, participants, system description, and GIBI-HE definition. Section 5 provides detailed construction of novel GIBI-HE scheme. In Section 6 gives key arrangement for trusted and malicious participants in our e-voting and also list out possible security models. Section 7 gives the proof under RO model for our GIBI-HE scheme and Section 8, we discuss the security and efficiency of this proposed scheme. Finally, Section 9 points out the future work in this area, followed by the conclusion at last.

2 Preliminaries

2.1 Discrete Logarithmic (DL) Assumption

We adopt the definition of DL assumption from [26] and [25] follows:

Definition 1. *Let $\mathbb{G}$ be a finite cyclic group of order q. Let g be a generator of $\mathbb{G}$, and let $x \in \mathbb{G}$. When (g, g^x) is known, x can be determined. The definition of the DL assumption is $Y = g^x$, which returns output x.*

2.2 Digital Signature

The standard definition of DS scheme [26] is described as follows:

Definition 2. *A DS scheme is denoted by a triple* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ *of polynomial-time algorithms, called key generation algorithm, signing algorithm and verification algorithm, respectively. The first two algorithms are probabilistic:*

- Gen. *On input 1^k, the algorithm produces a pair of matching public parameters* (param) *and public key and master-key* (pk, sk).
- Sign. *By taking* (sk, m) *as input, the algorithm returns a signature* $\sigma = \mathsf{Sign}_{\mathsf{sk}}(m)$*, where m is a message.*
- Verfiy. *On input* (pk, m, σ)*, the algorithm returns 1 (accept) or 0 (reject). We require that* $\mathsf{Verify}_{\mathsf{pk}}(m, \sigma) = 1$ *for all* $\sigma \leftarrow \mathsf{Sign}_{\mathsf{sk}}(m)$.

2.3 Transformation of DS to IBI Scheme

Any DS = $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ from definition 2 can be used as a building block to construct a canonical IBI = $(\mathsf{KeyGen}, \mathsf{Extract}, \mathsf{Verification})$. The transformation is given by Kurosawa and Heng [26], the setup algorithm Gen of DS with the key generation algorithm KeyGen of IBI and highlights their similarities. Both the setup algorithm Gen of DS and the key generation algorithm KeyGen of IBI take 1^λ as input and gives two outputs:

$$\text{DS scheme} \rightarrow \text{IBI scheme}$$

- param or mpk, which represent the public parameters or master public key.
- master-key or msk, which is used by the PKG in the Extract algorithm or as a signing key by the user.

Hence, we can compare that param = mpk and master-key = msk. The Extract algorithm Extract is same as Sign algorithm as it takes input ID and m and outputs the user secret key (usk) d and signature σ, respectively. We substitute ID = m and d = σ in the IBI scheme. P holds a secret key $d = \sigma$ corresponding to his public ID and communicates with the V by ZK proof.

Definition 3. *The standard definition of IBI scheme is given by Kurosawa [22] has three probabilistic polynomial-time (PPT) algorithms, which are as follows:*

1. KeyGen. PKG *takes the security parameter* 1^λ *and generates an output pair* (mpk, msk).
2. Extract. *It takes a user* ID, msk *and outputs the* usk.
3. Verification. P *takes the input* (mpk, usk, ID) *whereas* V *takes input as* (mpk, ID) *and communicates using a three-move canonical ZK proof: (i)* P *begins by sending commitment* CMT *to* V*. (ii)* V *picks a random challenge* CHA *and passes to* P*. (iii)* P *calculates a response* RES *using* usk *and sends it to* V*. (iv) If the response is correct, then it is accepted by* V*; otherwise, it is rejected.* RES *should satisfy the hardness of the assumption.*

The ZK protocol of DS to obtain the ZK protocol of IBI is shown below:

$$\begin{tikzcd}
\textbf{\textit{Prover}} & & \textbf{\textit{Verifier}} \\
d=\sigma & & \mathsf{param}=\mathsf{mpk}, \mathrm{ID}=m \\
{} \arrow[r, "\textit{Commitment}"] & {} & \\
{} & {} \arrow[l, "\textit{Challenge}"'] & \mathsf{CHA}\in\mathbb{Z}_q^* \\
{} \arrow[r, "\textit{Response}"] & {} & \mathsf{V}(\mathsf{param},\mathsf{CMT},\mathsf{CHA},\mathsf{RES})
\end{tikzcd}$$

2.4 Security Model for IBI Scheme

There are two phases of the security model to prove the IBI scheme shown in Table 1. The first phase is the training phase. It consists of the key setup, a hashing function, extraction, and verification using ZK. The second phase is the breaking phase where an impersonator tries to break the security of the scheme. This simulator is described in Kurosawa and Heng [26]. We are adopting simulator and probability distribution given in Vangujar et al. [34] for breaking phase of our GIBI-HE scheme. The two phases of security models are elaborated in Table 1:

Table 1. Security Model of IBI Scheme

Training Phase
Initially, an impersonator creates a pair of public and private keys in the first algorithm of training phase. The hashing function sets the hashing value for ID in the RO. Extract oracle extracts usk. Except KeyGen, each algorithm follows two cases: one is ID=ID* and another is ID $\neq$ ID*.
Breaking Phase
An impersonator attempts to forge complete security with a known set of entities. If an impersonator forges the key successfully, then it is deemed insecure.

We follow the same flow of simulator for our modified GIBI and novel GIBI-HE scheme to prove it secure under RO for different security Models.

2.5 Schnorr IBI Scheme

The Schnorr DS scheme [30] is converted to Schnorr IBI using transformation given in 2.3 considering definition 3. The Schnorr IBI scheme follows the DL assumption and has three PPT algorithms:

1. KeyGen. It takes 1^λ where λ is security parameter, and picks two prime numbers p and q in such a way that $p|(q-1)$. It generates the cyclic group $\mathbb{G}$ of prime

order q, generator $g \in \mathbb{G}$, and hashing function $H : \{0,1\}^* \in \mathbb{G}$. It chooses an arbitrary integer $x \in \mathbb{Z}_q^*$ and sets up $y = g^{-x}$. The output is $(\mathsf{mpk}, \mathsf{msk})$ where $\mathsf{mpk} = (\mathbb{G}, p, q, g, y, H)$ and $\mathsf{msk} = x$.
2. Extract. It takes (ID, mpk, x) as input and chooses a random integer $a \in \mathbb{Z}_q^*$. It computes $A = g^a$ and $s = a + x\alpha$ where $\alpha = H(\text{ID}, A, y)$. It returns an output usk as $d = (\alpha, s)$.
3. Verification. (i) CMT. P sets up $E = g^s y^\alpha$ and chooses a random integer $d \in \mathbb{Z}_q^*$ to computes $X = g^d$ and sends (E, X) to V. (ii) CHA. V picks a challenge $c \in \mathbb{Z}_q^*$ randomly and passes to P. (iii) RES. P calculates a response $Y = d + cs$ and forwards to V. (iv) V checks $g^Y = X(E/y^\alpha)^c$. If equality holds by DL assumption, then it outputs accept as 1, otherwise reject as 0.

2.6 Group Identity-based Identification Scheme

Group Identity-based Identification (GIBI) scheme by Vangujar et al. [36] is constructed using the Schnorr IBI [33] and Schnorr DS [24] scheme and defined as transactions between the TA, the Group Manager (GM), different groups ($\mathsf{G}_1, \mathsf{G}_2, ...\mathsf{G}_\mathsf{i}, ..., \mathsf{G}_\mathsf{n}$), and group members.

1. KeyGen. TA sets up param and after calculation outputs the pair $(\mathsf{mpk}, \mathsf{msk})$.
2. Extract.
   (a) *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.
   (b) *Phase 2*. Run by the GM for each group member who possess an ID. To register as a member of the group, the ID is sent to the GM. Taking ancestor $(\mathsf{gpk}, \mathsf{gsk})$ as input, GM generates user keys $(\mathsf{upk}, \mathsf{usk})$ for corresponding group member ID.
   (c) *Phase 3*. Run by the GM. GM stores $(\text{ID}, \mathsf{upk})$ for that ID and do not store usk.
3. Verification.
   (a) *Phase 1* (CMT). Assume a group member G_i wants to run ZK proof as a group. Taking G_1 as an example, for input usk, G_1 outputs a signature σ_1. G_1 then asks GM_1 for a request to verify, and sends $(\mathsf{upk}, \sigma_1, \text{ID})$ to GM_1.
   (b) *Phase 2* (CHA). GM_1 checks if σ_1 is valid and verifies if the associated ID is within the list of members. If G_1 is a valid member in the list, GM_1 issues a notice to all other members in the group to generate their signatures and attach their upk. As GM_1 receives the values for each members in group, GM_1 also checks if the provided values are valid or not.
   (c) *Phase 3* (RES). Once all values are valid, GM then performs verification by V, by attaching a signature generated from gsk, σ_g as a representation of the group verification. GM performs ZK with V.

We will use this GIBI definition to construct our proposed scheme but we will first construct modified GIBI using Schnorr IBI and eliminate the use of signature σ from Verification *Phase 1* and *Phase 2* algorithm to make it more relevant for e-voting system and only run ZK proof to check *eligibility* of all voters.

2.7 Homomorphic Encryption

ElGamal The popular ElGamal cryptosystem [19] is secure under the DL hardness assumption and also homomorphic under multiplication. Let q be a prime and g be the generator of the cyclic multiplicative group $\mathbb{Z}_q^*$. Let Alice and Bob be the two parties who want to perform the computation. The ElGamal cryptosystem has three algorithmic steps (KeyGen, $\mathsf{Encrypt}$, $\mathsf{Decrypt}$) as follows:

1. KeyGen. It takes the input as 1^λ and outputs the pair of $(\mathsf{sk}, \mathsf{pk})$ where sk is the secret key, randomly chosen from $\mathbb{Z}_q^*$ and $\mathsf{pk} = g^{\mathsf{sk}} \bmod q$. Alice sends Bob (pk, g, q).
2. $\mathsf{Encrypt}$. By taking input (pk, m), Bob encrypts and provides the output as (m_1, m_2). Bob calculates $m_1 = g^k \bmod q$ and $m_2 = (\mathrm{pk}^k \cdot m) \bmod q$, where k is a randomly chosen value. Bob then sends the tuple (m_1, m_2) to Alice.
3. $\mathsf{Decrypt}$. Alice finally takes (sk, m_1, m_2) as input and calculates $m_2/m_1^{\mathsf{sk}} \bmod q$.

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 4. *Let the group of plaintexts space be denoted by $\mathcal{M}$ under addition, and the group of ciphertexts space be denoted by $\mathcal{C}$ under multiplication. An encryption function E is homomorphic if, given $c_1 = \mathsf{E}(m_1)$ and $c_2 = \mathsf{E}(m_2)$ for plaintexts $m_1, m_2 \in \mathcal{M}$, it holds that $c_1 \cdot c_2 = \mathsf{E}(m_1 + m_2)$, where $+$ and $\cdot$ represents the corresponding group operations.*

Distributed ElGamal The distributed version of the Exponential ElGamal from definition 4 is DE for n user. For each user i where $1 \leq i \leq n$ considering all operations under mod q. The DE exponential ElGamal algorithm is adopted from [39] and used in our proposed e-voting scheme to support additive homomorphism.

1. KeyGen. It takes the input $(\mathbb{G}, g, q)$ and gives output as $(\mathsf{pk}_i, \mathsf{sk}_i)$, 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 cpk used for encryption for all voters and it is calculated as:
$$\mathsf{cpk} = \prod_{i=1}^{n} y_i = g^{x_1 + \ldots + x_n}$$
2. $\mathsf{Encrypt}$. For encrypting message m, it chooses a random $k_i \in \mathbb{Z}_q^*$ and compute $c_i = (a_i, b_i)$. For user i, encrypting their message m:
$$a_i = g^{k_i}; \quad b_i = g^m \mathsf{cpk}^{k_i} \tag{1}$$
The encrypted message is $\mathsf{Enc}(m) = c_i = (a_i, b_i)$.
3. $\mathsf{Decrypt}$. User takes input (x_i, c_i) and outputs $m = b_i/a_i^{x_i}$ for one message. For multiple message, DE requires each user to calculate and broadcast partial decryptions and then homomorphically combined to calculate the final tally to reveal message. During the tallying phase, the decryption procedure for a combined ciphertext (a, b) is as follows:

– Each i-th user computes $a_i^{x_i}$ and broadcasts commitment of computed values $H(a_i^{x_i})$ so that anyone can check if each $a_i^{x_i}$ matches with $H(a_i^{x_i})$.
– Each i-th user sends the partial share to the authority to decrypt the combined message $m = m_1 + ... + m_n$.

$$\mathsf{Dec}(c_i) = \frac{b_i}{\prod_{i=1}^{n} a_i^{x_i}} = \frac{b_i}{a_i^{x_1+...+x_n}} = g^m \tag{2}$$

Finally, m can be revealed by computing hardness of DL assumption.

3 Modified GIBI Scheme and ZK Proof for Proposed E-voting

3.1 Requirement of ZK in GIBI scheme

We consider the requirement of ZK in DS proposed by Kurosawa and Heng [26] and eliminate the need of signature in ZK and shows a canonical (three-move) ZK interactive proof system on knowledge of ID for GIBI scheme using the tranformation given in Section 2.3 as follows: Let mpk be a master public key, ID be an identity and d be a usk of ID. The common input to (P, V) is $(\mathsf{mpk}, \mathrm{ID})$. The secret input to P is d. Let ZK $=(\mathsf{CMT}, \mathsf{CHA}, \mathsf{RES})$ be a transcript of the conversation between P and V. We say that Δ-ZK proof is acceptable if V accepts it.

Definition 5. *We say that GIBI scheme has a Δ-ZK protocol if there exists a canonical protocol* (P, V) *or any* $(\mathsf{mpk}, \mathrm{ID})$ *as follows:*

– *Completeness. If* P *knows* usk*, then accepts* $\Pr[\mathsf{V}] = 1$*.*
– *Soundness. The number of possible* CHA *is equal to* Δ*.* usk *is computed efficiently from any two transcripts* $(\mathsf{CMT}, \mathsf{CHA}_1, \mathsf{RES}_1)$ *and* $(\mathsf{CMT}, \mathsf{CHA}_2, \mathsf{RES}_2)$ *where* $\mathsf{CHA}_1 \neq \mathsf{CHA}_2$*.*
– *Zero-knowledgeness.* (P, V) *is perfectly ZK for the honest verifier. There is a simulator such that its output follows the same probability distribution as ZK.*

Now, we use definition 5 to construct the modified GIBI and later used in construction of novel GIBI-HE scheme for our e-voting scheme. We show our GIBI-HE scheme satisfies these proprieties in simulator and make the scheme secure under RO model.

3.2 Modified GIBI Scheme Using Schnorr IBI

We initialize the GIBI using the Schnorr IBI and eliminate the requirement of Schnorr DS given by Vangujar et al. [35]. We construct new scheme GIBI using construction given in Section 2.6 and definition for ZK proof. It is obvious that signature part from original scheme will be discarded in the modified GIBI and makes it purely identification scheme. The GroupIBI involves the TA and GM to perform a collective verification as a group, whereas the usk involves the transaction between the GM and the group members $(\mathrm{ID}_1, \mathrm{ID}_1, ..., \mathrm{ID}_i, ..., \mathrm{ID}_m)$ to prove their ID as a valid group member. We refer to [33]'s tight Schnorr IBI variant to construct the Group-IBI, with consideration it use the hard DL assumption. Fig 1 gives the generic arrangement of participants for the modified GIBI scheme and also shows how we expand same arrangement for our GIBI-HE e-voting scheme.

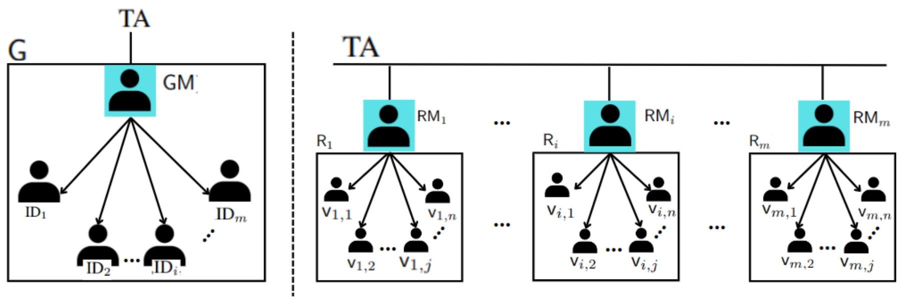

Fig. 1. Generic Arrangement of Participants for Modified GIBI and GIBI-HE Scheme

1. KeyGen. On a security parameter 1^λ as input, TA generates two large primes p and q, such that $q|(p-1)$. TA also generates $x \in \mathbb{Z}_q^*$ to compute $y_1 = g_1^{-x}$ and $y_2 = g_2^{-x}$ where $g_1, g_2 \in \mathbb{G}$. Compute a hash function $H : \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \in \mathbb{Z}_q^*$. The master public key mpk is $(\mathbb{G}, p, q, g_1, g_2, y_1, y_2, H)$ while the master secret key msk is x.
2. Extract.
   (a) *Phase 1.* Given a group GM for G, TA selects a random integer $t \in \mathbb{Z}_q^*$. Then, TA computes $A = g_1^t, B = g_2^t$, and also calculate $s = t + x\alpha$ where $\alpha = H(\mathsf{GM}, A, B, y_1, y_2)$. TA generates the pair where the group public keys gpk $= (\mathsf{GM}, g_1, g_2, y_1, y_2)$ and group secret keys gsk $= (\alpha, s)$ to pass to GM.
   (b) *Phase 2.* This phase is run by GM, consider a group member ID_i where $1 \leq i \leq m$ from group G wants to register as its group member, he sends ID_i to GM. GM generates a random integer $a_i \in \mathbb{Z}_q^*$ and then computes $y_{i,1} = g_1^{a_i}$ and $y_{i,1} = g_2^{a_i}$. Next, GM sets $\beta = H(\mathrm{ID}_i, y_{i,1}, y_{i,2}, A, B)$ and calculates $\hat{s} = a_i + t\beta$. GM finally outputs group user public key upk_i where $\mathsf{upk}_i = (\mathrm{ID}_i, g_1, g_2, A, B)$ and group user secret key usk_i where $\mathsf{usk}_i = (\beta, \hat{s})$. GM passes pair to ID_i and do not store usk_i.
3. Verification. ID_i as P performs the transaction with a GM as V. The ZK protocol are carried out as follows:
   (a) CMT. ID_i computes $E = g_1^{\hat{s}} {y_{i,1}}^{\beta}$ and $F = g_2^{\hat{s}} {y_{i,2}}^{\beta}$. Then, ID_i generates random integers $r_{i,1}, r_{i,2} \in \mathbb{Z}_q^*$, computes $X = g_1^{r_{i,1}} g_2^{r_{i,2}}$ and then sends (E, F, X) to GM.
   (b) CHA. GM then generates a challenge $c \in \mathbb{Z}_q^*$ randomly and sends c to ID_i.
   (c) RES. ID_i computes the response value $Y_{i,1} = r_{i,1} + c\hat{s}$, $Y_{i,2} = r_{i,2} + c\hat{s}$ and then sends the value of $(Y_{i,1}, Y_{i,2})$ to GM.

   GM accepts the value if and only if the value of $g_1^{y_{i,1}} g_2^{y_{i,2}} = X \cdot (EF/y_{i,1}^{\beta} y_{i,2}^{\beta})^c$.

This modified constructed scheme is using only one group but our idea to make many groups and incorporate DE and ZK to make it more appropriate for e-voting systems considering all the requirements, participants, and security model.

3.3 Security Analysis of GIBI Scheme

Depending on the scenario, multiple security models can be mentioned, but here we will provide examples for malicious as another group member. We will provide a security model based on malicious entities for our main GIBI-HE scheme. We demonstrate the security of the model by simulating security against the RO model.

We define the security proof against impersonation as another group member, 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 a member of the group, respectively.

Theorem 1. *The Group-IBI scheme above is secure against impersonation in the RO model if the DL hard assumption holds.*

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

1. **Training Phase**.
   (a) KeyGen. $\mathbb{C}$ obtains mpk.
   (b) $\mathsf{Extract}$. For an extract query of $\mathrm{ID}_{\mathbb{I}}$ queried by $\mathbb{I}$, $\mathbb{C}$ computes and sends $(\mathsf{upk}_{\mathbb{I}}, \mathsf{usk}_{\mathbb{I}})$ to GM.
   (c) $\mathsf{Verification}$. For ZK query on $\mathrm{ID}_{\mathbb{I}}$ queried by $\mathbb{I}$, $\mathbb{C}$ checks if $\mathrm{ID}_{\mathbb{I}}$ has been queried in extract query before. If so, $\mathbb{C}$ uses the existing $(\mathsf{upk}_{\mathbb{I}}, \mathsf{usk}_{\mathbb{I}})$ to return a valid transcript/conversation for $\mathbb{I}$; else, $\mathbb{C}$ runs extract query algorithm to generate $(\mathsf{upk}_{\mathbb{I}}, \mathsf{usk}_{\mathbb{I}})$.
2. **Breaking Phase** . $\mathbb{I}$ pretends to be a valid group member using $\mathrm{ID}^*_{\mathbb{I}}$, where $\mathrm{ID}^*_{\mathbb{I}}$ was queried during the extract query. $\mathbb{I}$ generates a $\mathsf{usk}_{\mathrm{ID}^*_{\mathbb{I}}}$ and then sends the it to $\mathbb{C}$. After $\mathbb{C}$ obtains the $\mathsf{usk}_{\mathrm{ID}^*_{\mathbb{I}}}$, $\mathbb{C}$ checks the validity of the group user. [1]. If the usk^*_i produced by $\mathbb{I}$ is not valid, $\mathbb{C}$ aborts and it fails in the security game. Else, $\mathbb{C}$ can use the forgery $\mathsf{usk}_{\mathrm{ID}^*_{\mathbb{I}}}$ to solve the hard problem used in the scheme and wins in the security game.

It is noted that during the query phase, the probability of aborts occurring is highly dependent on the ZK used. However, the abort that may occur during the query phase due to a hash collision is negligible. Therefore, it can be supposed that if $\mathbb{I}$ is able to come up with a valid $\mathsf{usk}_{\mathbb{I}}$, $\mathbb{I}$ has broken the user secret key scheme used in the modfied GIBI scheme. We consider the same security proof simulator for different scenario under RO and prove it secure if the DL assumption holds for GIBI-HE scheme.

4 Our E-voting System

4.1 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 2 for e-voting schemes and what we wish to achieve in the one we propose. *Eligibility, anonymity*, and *privacy* are assured by the use of the GIBI scheme. *Robustness, individual* and *universal verifiabilty* are given by ZK proofs in decrypting votes. *Fairness* and *receipt-freeness* are provided using the HE tally portion. *Uncoercibilty* is maintained due to use of strong DE under DL assumption.*Unresuability* is satisfied in $\mathsf{Validate}$ phase where elgibile vote can cast only one ballot.

[1] It is noted that $\mathbb{I}$ has to produce the ID of a valid member in the group, else $\mathrm{ID}^*_{\mathbb{I}}$ will fail when $\mathbb{C}$ does cross-checking on the validity of $\mathrm{ID}^*_{\mathbb{I}}$ as a group member.

Table 2. Requirements for E-voting Scheme

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.
Anonymity	All eligible 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.

4.2 Participants

- **Trusted Authority.** Trusted Authority TA is responsible for setting up the system.
- **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 voters v. For example: For $\mathsf{R}_1 = (\mathsf{RM}_1, \mathsf{v}_{1,1}, \mathsf{v}_{1,2}...\mathsf{v}_{1,j}, .., \mathsf{v}_{1,n})$, RM supervises the generation of private and public keys for each voter in v. The arrangement of TA, RM, and voters are shown and compared with modified GIBI scheme in Fig. 1.
- **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. For example: For voter $\mathsf{v}_{i,j}$, it belongs to R_i positioning j^{th} in v as shown in Fig 1.
- **Candidates.** There are set of candidates $\mathsf{C} = (\mathsf{C}_1, \mathsf{C}_2, ..., \mathsf{C}_k, ..., \mathsf{C}_l)$, 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. Each voter from v can then select their preferred single choice from C by clicking on their name or otherwise indicating their choice.
- **Voting Authorities.** Voting Authorities $\mathsf{VA} = (\mathsf{VA}_1, \mathsf{VA}_2, ..., \mathsf{VA}_i, ..., \mathsf{VA}_m)$ where number of R is equal in number as VA in our e-voting system. The VA manages ZK proofs for all voters in $\mathbf{v}$. VA allow only eligible voter to cast the ballot. For example: VA_i will be responsible for verifying all eligible voters v under R_i.

– **Tallying Authorities.** 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 final tally to respective VA to declare results. The number of Ta is equal 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.

4.3 System Description

The proposed GIBI-HE e-voting process includes five major phases described as follows and shown in Fig. 2: The setup, registration, and authorizing phase in Fig. 3; voting and tallying phase is shown in Fig. 4.

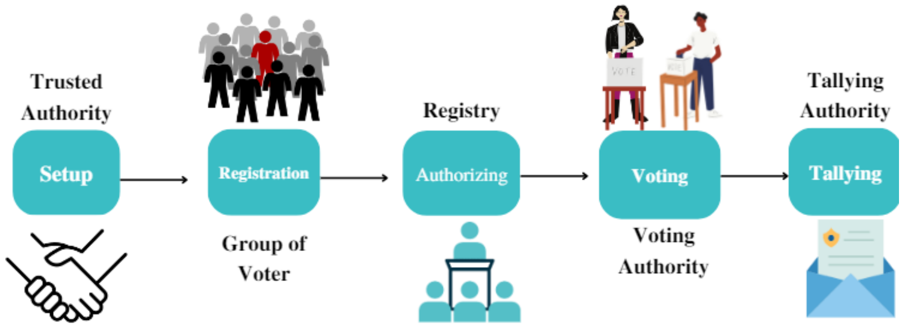

Fig. 2. Flow of Proposed E-voting Scheme

1. **Setup**. In this phase, the 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.
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 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 BB.

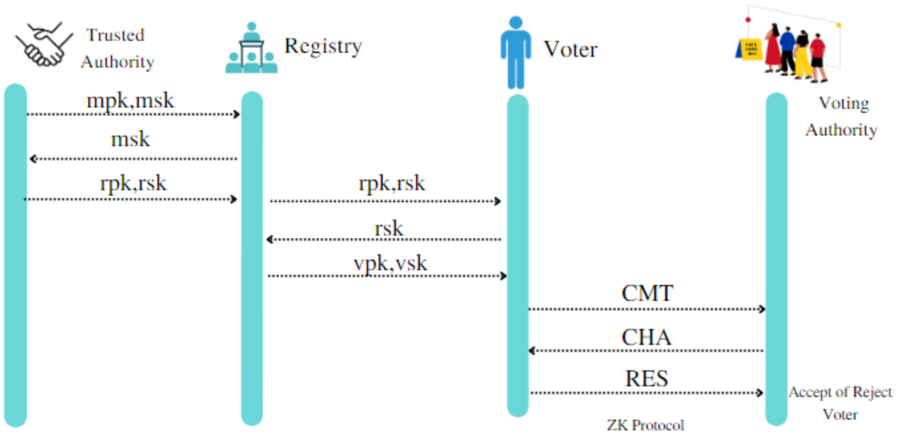

Fig. 3. Setup, Registration, and Authorizing Phase of Proposed E-voting Scheme

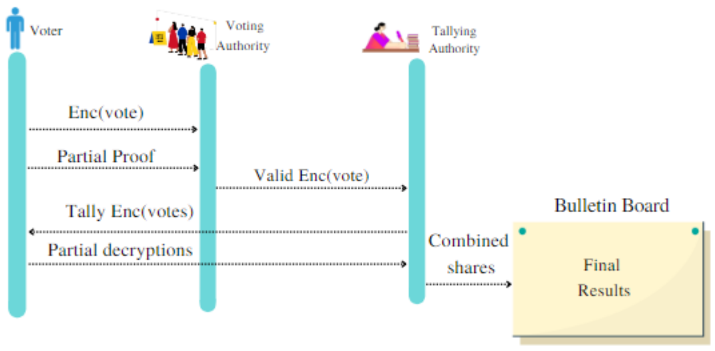

Fig. 4. Voting and Tallying Phase of Proposed e-Voting Scheme

4.4 Definition of GIBI-HE Scheme

Proposed GIBI-HE scheme consists of seven PPT algorithms such that GIBI-HE = $(\mathsf{KeyGen}, \mathsf{Extract}, \mathsf{Verification}, \mathsf{Encrypt}, \mathsf{Validate}, \mathsf{Tally}, \mathsf{Decrypt})$ and is constructed using definition 3, newly constructed Schnorr-like modified GIBI from Section 3.2, and DE from definition 4 run among TA, v, R, VA, C, Ta, and BB.

1. KeyGen. Using security parameter 1^λ and secret key chosen by the TA, pair of master public and secret key $(\mathsf{mpk}, \mathsf{msk})$ is generated as output.
2. Extract. Considering the generic scenario for e-voting scheme as follows:
   (a) *Phase 1*. For RM_i, TA calculates registry public and secret key $(\mathsf{rpk}_i, \mathsf{rsk}_i)$ using ancestor msk.
   (b) *Phase 2*. For any $\mathsf{v}_{i,j}$, RM_i calculates voter public and secret key $(\mathsf{vpk}_{i,j}, \mathsf{vsk}_{i,j})$ using ancestor rsk_i.

   (c) *Phase 3*. Using the key of the voter for encryption, the common public key cpk is generated by multiplying all the voter's vpk, cpk is sent back to all voters in v for encryption.
3. $\mathsf{Verification}$. Following are the three stages of communication between the voter $\mathsf{v}_{i,j}$ (acting as P) and the VA_i (acting as V):
   (a) CMT. $\mathsf{v}_{i,j}$ chooses a random integer to calculate value and sends to VA_i.
   (b) CHA. VA_i generates a random challenge and forwards to $\mathsf{v}_{i,j}$.
   (c) RES. $\mathsf{v}_{i,j}$ accepts the challenge and generates response, further passes to VA_i.
   (d) VA_i accepts $\mathsf{v}_{i,j}$ as eligible voter for voting process if and only if, it verifies the final equation using DL-tuple.
4. $\mathsf{Encrypt}$. After the authorization using ZK, the voter $\mathsf{v}_{i,j}$ casts the vote $v_{i,j}$ and encrypts it using cpk. For C_k candidates and $\mathsf{v}_{i,j}$, the vote has form the ballot of encrypted vote $ev_{i,j} = (\mathsf{Enc}(b_1), ..., \mathsf{Enc}(b_k), ..., \mathsf{Enc}(b_l))$, where ballot $b_k = 0$ or 1 where $1 \leq k \leq l$ depending on which candidate the ballot was cast for.
5. $\mathsf{Validate}$. 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 equals to 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 casted two votes rather than simply one.
6. Tally. Ta_i collect the encrypted votes from the voters and tally them using homomorphic additive property.
7. $\mathsf{Decrypt}$. To decrypt the vote, VA_i collects partial shares from all v to compute the total combined tally and VA_i shows the final result on BB.

5 Construction of GIBI-HE e-voting Scheme

The GIBI-HE is a new e-voting scheme that combines the use of Schnorr IBI 2.5 with GIBI 2.6 and definition DE 4 to provide a secure and efficient e-voting system. We first constructed modified GIBI in Section 3.2 and we will be using this as new base GIBI for GIBI-HE scheme which is proven security under RO model. GIBI-HE 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 ID with the VA and obtaining the necessary rights to cast their ballots. The GIBI-HE 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 $\mathsf{Extract}$ algorithm. A new type of e-voting scheme is based on the six PPT algorithms for five phases setup, registration, authorization, voting, and tallying:

5.1 Setup

1. KeyGen. On a security parameter 1^λ, TA takes cyclic group $\mathbb{G}$ and 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 (mpk, msk) is passed to registry R_i to it respective RM_i. KeyGen of GIBI-HE scheme is similar to the KeyGen of modified GIBI scheme in Section. 3.2 which follows Schnorr key generation technique. We are considering only one prime number i.e q in GIBI-HE than two i.e p and q.

5.2 Registration

2. $\mathsf{Extract}$.
Phase 1. It is run by TA. TA will be responsible for generating key pairs for all RM. There are multiple Registries $(\mathsf{R}_1, \mathsf{R}_2, ..., \mathsf{R}_i, ..., \mathsf{R}_m)$ 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 $\mathsf{R}_i = (\mathsf{RM}_i, \mathsf{v}_{i,1}, \mathsf{v}_{i,2}, ..., \mathsf{v}_{i,j}, ..., \mathsf{v}_{i,n})$ and Fig. 1 shows this arrangement for our e-voting scheme.. TA 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 pair of 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 RM_i.
Phase 2. This phase is run by RM_i. Consider voter $\mathsf{v}_{i,j}$ from set of all v who wants to register as a member of the voting list of the registry R_i. The voter sends their ID $\mathsf{v}_{i,j}$ to RM_i, which takes as input $(\mathsf{rpk}_i, \mathsf{rsk}_i, \mathsf{v}_{i,j}, \mathsf{RM}_i)$. RM_i selects a random integer $\hat{a}_i \in \mathbb{Z}_q^*$ and passes $\hat{a}_i$ for encryption to *Phase 3* and computes $\hat{A}_i = g_1^{\hat{a}_i}$, $\hat{B}_i = g_2^{\hat{a}_i}$ and $\hat{s}_i = \hat{a}_i + t_i\beta_i$, where $\beta_i = H(\mathsf{v}_{i,j}, \mathsf{RM}_i, \hat{A}_i, \hat{B}_i, A_i, B_i)$. The RM_i outputs the voter's public key $\mathsf{vpk}_{i,j} = (\mathsf{v}_{i,j}, \mathsf{RM}_i, g_1, g_2, A_i, B_i)$ and the voter's secret key $\mathsf{vsk}_{i,j} = (\beta_i, \hat{s}_i)$. The registry only stores all $\mathsf{vpk}_{i,j}$ along with their respective $\mathsf{v}_{i,j}$, and it passes $\mathsf{vsk}_{i,j}$ to all voters, respectively.
Phase 3. This phase is addition to modified GIBI and it shows the encryption keys for every voter is generated. 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 encryption public and secret key $\mathsf{epk}_i = \hat{A}_i\hat{B}_i = g_1^{\hat{a}_i}g_2^{\hat{a}_i}$ and $\mathsf{esk}_i = \hat{a}_i$, 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.

5.3 Authorization

3. $\mathsf{Verification}$. Each voter has to prove their $eligibity$ to VA to cast their ballot. $\mathsf{v}_{i,j}$ as P performs the transaction with a VA_i as V. The ZK is carried out as follows:
   (a) CMT. $\mathsf{v}_{i,j}$ computes $E_i = g_1^{\hat{s}_i} g_2^{\hat{s}_i} \hat{A}_i^{\beta_i} \hat{B}_i^{\beta_i}$. Then, $\mathsf{v}_{i,j}$ generates 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) RES. $\mathsf{v}_{i,j}$ computes response $Y_{i,1} = r_{i,1} + c_i\hat{s}_i$, $Y_{i,2} = r_{i,2} + c_i\hat{s}_i$ 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:

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

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

Correctness Proof. The correctness of the ZK by definition 5 can be proven as such:

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

5.4 Voting

4. $\mathsf{Encrypt}$. In this phase, eligible voter are given right to cast a ballot and encrypt it. By taking input cpk, each voter $\mathsf{v}_{i,j}$ casts a ballot $v_{i,j}$ and encrypt it $ev_{i,j} = \mathsf{Enc}(v_{i,j})_{\mathsf{cpk}}$. Encrypted vote is given as $ev_{i,j} = (a_{i,j}, b_{i,j})$ where for random integers $r_{i,j_1}, r_{i,j_2} \in \mathbb{Z}_q^*$ and computes $ev_{i,j} = (g_1^{r_{i,j_1}} g_2^{r_{i,j_2}}, g_1^{v_{i,j}} \cdot \hat{A}_i^{r_{i,j_1}} g_2^{v_{i,j}} \cdot \hat{B}_i^{r_{i,j_2}})$ following Eq. 1. After the ballot is cast and sent to the VA_i, $v_{i,j}$ is checked for validity. To accommodate our overall scheme based on the DL assumption, our modified DE construction utilizes a pair of generators (g_1, g_2).
5. $\mathsf{Validate}$. This phase is run by VA_i and it checks the validity of the ballots to final tally or it also discards invalid votes. We use the proof of ZK using partial shares for checking the validity of the votes. Using ZK proofs, VA_i verify that the vote is valid, i.e., the voter has voted only for one candidate C_k:
   (a) CMT. VA_i takes the input $ev_{i,j}$ and also calculate the collapsed vote for the partial proof $\mathsf{pp}_{i,j} = \prod_{k=1}^{l} (ev_{i,j}[k]) = (\mathsf{pp}_{i,j}[0], \mathsf{pp}_{i,j}[1])$.
   (b) CHA. VA_i picks self generated random challenge $\hat{c}_{i,j} \in \mathbb{Z}_q$.
   (c) RES. Using the challenge $\hat{c}_{i,j}$, the voter VA_i calculates the following:
      - $T_{i,j} = g_1^{\hat{c}_{i,j}} g_2^{\hat{c}_{i,j}}$ and $\hat{T}_{i,j} = (\hat{A}^{r_{i,j_1}} \hat{B}^{r_{i,j_2}})^{\hat{c}_{i,j}}$.
      - The token is obtained by hashing the voter's ballot such that $\mathtt{token} = H\big(\mathsf{pp}_{i,j} || T_{i,j} || \hat{T}_{i,j}\big)$ and it can be later useful for voters to check if their ballot is being counted or not in final tally.
      - $S_{i,j} = \hat{r}_{i,j_1} \cdot \mathtt{token} + \hat{c}_{i,j}$ and $\hat{S}_{i,j} = \hat{r}_{i,j_2} \cdot \mathtt{token} + \hat{c}_{i,j}$.
      - Finally RES = $(a_{i,j}, b_{i,j}, \mathsf{pp}_{i,j}, T_{i,j}, \hat{T}_{i,j}, \mathtt{token}, S_{i,j}, \hat{S}_{i,j})$ is generated.

To verify the validity of a vote in our e-voting scheme, i.e, to check if the addition of the component of the vote equals to 1. `token` makes vote anonymous and ensures *privacy* and keeps no record on ballot makes it *receipt-free*. If Eq. 3 verification holds equal then VA_i passes ballots to final tally or else it will be discarded.

$$g_1^{S_{i,j}} g_2^{\hat{S}_{i,j}} = a_{i,j}^{\mathtt{token}} \cdot T_{i,j}; \quad \mathsf{cpk}^{S_{i,j} \hat{S}_{i,j}} = \left(\frac{b_{i,j}}{g_1{}^{v_{i,j}} g_2{}^{v_{i,j}}} \right)^{\mathtt{token}} \cdot \hat{T}_{i,j} \qquad (3)$$

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

$$\begin{aligned} g_1^{S_{i,j}} g_2^{\hat{S}_{i,j}} &= a_{i,j}^{\texttt{token}} \cdot T_{i,j} \\ &= (g_1^{r_{i,j_1}} g_2^{r_{i,j_2}})^{\texttt{token}} \cdot g_1^{\hat{c}_{i,j}} g_2^{\hat{c}_{i,j}} \\ &= g_1^{r_{i,j_1}\texttt{token}} g_2^{r_{i,j_2}\texttt{token}} \cdot g_1^{\hat{c}_{i,j}} g_2^{\hat{c}_{i,j}} \\ &= g_1^{r_{i,j_1}\texttt{token}+\hat{c}_{i,j}} g_2^{r_{i,j_2}\texttt{token}+\hat{c}_{i,j}} \\ &= g_1^{S_{i,j}} g_2^{\hat{S}_{i,j}} \end{aligned}$$

If the equation 3 holds, it means that the exponents on both sides of the equation are equal, and thus, the voter $\mathsf{v}_{i,j}$ has provided a valid vote for a single candidate C_k. If not, the vote is not valid and not included in the final tally. This helps ensure the *unresuability* and *recipet-freeness* of our GIBI-HE e-voting scheme.

$$\begin{aligned} \mathsf{cpk}^{S_{i,j}\hat{S}_{i,j}} &= \left(\frac{b_{i,j}}{g_1{}^{v_{i,j}} g_2{}^{v_{i,j}}}\right)^{\texttt{token}} \cdot \hat{T}_{i,j} \\ &= \left(\frac{g_1^{v_{i,j}} \cdot \hat{A}_i^{r_{i,j_1}} g_2^{v_{i,j}} \cdot \hat{B}_i^{r_{i,j_2}}}{g_1^{v_{i,j}} g_2^{v_{i,j}}}\right)^{\texttt{token}} \cdot \hat{T}_{i,j} \\ &= (\hat{A}_i^{r_{i,j_1}} \hat{B}_i^{r_{i,j_2}})^{\texttt{token}} \cdot (\hat{A}_i^{r_{i,j_1}} \hat{B}_i^{r_{i,j_2}})^{\hat{c}_{i,j}} \\ &= \mathsf{cpk}^{S_{i,j}\hat{S}_{i,j}} \end{aligned}$$

5.5 Tallying

6. Tally. This phase is where valid votes are considered by Ta_i. The encrypted votes are collected from the voters by VA_i and tallied by the Ta_i. This algorithm just requires the collected ciphertexts $\{(a_{i,j}, b_{i,j})\}_{j=1}^{n}$ of valid votes. Ta_i combines all the encrypted votes without decrypting them. The Ta_i 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 ballots though only valid votes are considered for ease of reading.
Correctness proof. Here is a correctness proof to get summation of all the cast ballot valid votes by voter under R_1 for $\mathsf{Enc}\left(\prod_{j=1}^{n} v_{1,j}\right)$:

$$\begin{aligned} &= (g_1^{\hat{r}_{1,1_1}} g_2^{\hat{r}_{1,1_2}}, g_1^{v_{1,1}} g_2^{v_{1,1}} \cdot \hat{A}_1^{\hat{r}_{1,1_1}} \hat{B}_1^{\hat{r}_{1,2_2}}) \cdots (g_1^{\hat{r}_{1,n_1}} g_2^{\hat{r}_{1,n_2}}, g_1^{v_{1,n}} g_2^{v_{1,n}} \cdot \hat{A}_1^{\hat{r}_{1,n}} \hat{B}_1^{\hat{r}_{1,n}}) \\ &= (g_1^{\sum_{j=1}^{n} \hat{r}_{1,j_1}} g_2^{\sum_{j=1}^{n} \hat{r}_{1,j_2}}, g_1^{\sum_{j=1}^{n} v_{1,j}} g_2^{\sum_{j=1}^{n} v_{1,j}} \cdot \hat{A}_i^{\sum_{j=1}^{n} \hat{r}_{1,j_1}} \hat{B}_i^{\sum_{j=1}^{n} \hat{r}_{1,j_2}}) \\ &= (a_{1,n}, b_{1,n}) \end{aligned}$$

For any registry R_i, (a_i, b_i) represents the encrypted sum of all valid votes cast by voters under registry R_i, we generally define tally $\mathsf{Enc}\Bigg(\prod_{j=1}^{n} v_{i,j}\Bigg)$ as defined below :

$$
\begin{aligned}
&= (g_1^{\sum_{j=1}^{n} \hat{r}_{i,j_1}} g_2^{\sum_{j=1}^{n} \hat{r}_{i,j_2}}, g_1^{\sum_{j=1}^{n} v_{i,j}} g_2^{\sum_{j=1}^{n} v_{i,j}} \cdot \hat{A}_i^{\sum_{j=1}^{n} \hat{r}_{i,j_1}} \hat{B}_i^{\sum_{j=1}^{n} \hat{r}_{i,j_2}}) \\
&= (a_{i,j}, b_{i,j})
\end{aligned}
$$

7. $\mathsf{Decrypt}$. As there is no central secret/decryption key, decrypt involves two rounds of communication as follows: $\mathsf{Enc}\Bigg(\prod_{n=1}^{i} v_{m,n}\Bigg) = (a_{m,n}, b_{m,n})$
   (a) The combined ciphertext $(a_{m,n}, b_{m,n}) = (a, b)$ from the tallying stage is forwarded to all voters. All voters compute their partial shares $a^{\hat{a_i}}$ for the secret key $\hat{a}_i$ and sent back to the VA_i.
   (b) Then by Eq. 2, the partial shares are combined together and DL assumption is performed to retrieve the final tally. VA_i to decrypt the combined vote v.

$$
\frac{b}{\prod_{i=1}^{n} a^{\hat{a_i}}} = \frac{b}{a^{\hat{a_1}+\ldots+\hat{a_n}}} = v. \tag{4}
$$

   Finally, the total number of votes v can be revealed by computing DL assumption. Any voter can check their vote is counted and included in final tally by decrypting and checking final result on BB. It ensures *individual verifiability*. Additionally, any eligible voter can check election is fair or not which satisfy the requirement of *universal verifiability*.

6 Security Models

In this section, we present the security models of the GIBI-HE scheme considering different scenarios. Table 3 provided illustrates the distribution of keys among different authorities in the GIBI-HE e-voting scheme. It showcases the availability of public components and their accessibility to both trusted and malicious participants, while addressing various key requirements for a secure and transparent e-voting system.

– *Eligibility*. It ensures that only eligible voters can participate in the voting process after passing the ZK proof with VA. In the scheme, the distribution of keys ensures that malicious authorities do not have access to thE $\mathsf{vsk}_{i,j}$ that determine voter *eligibility*, thus upholding the integrity of the system.
– *Privacy* and *Anonymity*. By restricting the accessibility of public components to malicious authorities and strong ZK for twice in GIBI-HE, it protects the *privacy* of voters, ensuring that their choices remain confidential using DE. Moreover, the controlled ZK verify voters will help to satisfy DL-tuple to maintain the *anonymity* of voters, preventing malicious authorities from linking votes to specific individuals.

- *Unreusability*. This prevents voters from casting multiple votes in our e-voting scheme. By not giving the accessibility of cpk and $\texttt{token}$ as showing in Table 3 and $\mathsf{Validate}$ algorithm helps to checking validity of vote using $\mathsf{pp}_{i,j}$, the scheme ensures that any malicious authorities cannot manipulate or reuse $v_{i,j}$, thereby maintaining the *unreusability* of cast ballots.

Table 3. Key Distribution among Various Authorities in GIBI-HE E-voting Scheme

Keys — Trusted Participants							Malicious Participants					
	TA	RM	v	VA	Ta	BB	TA	RM	v	VA	Ta	Algorithm
mpk	✓	✓	✓	✓	✓		✓	✓	✓	✓		KeyGen, Verification
msk	✓											KeyGen
rpk	✓	✓										Extract- *Phase 1*, Verification
rsk		✓										Extract- *Phase 1*
vpk		✓	✓						✓			Extract- *Phase 2*, Verification
vsk			✓									Extract- *Phase 2*
epk		✓	✓									Extract- *Phase 3*
esk			✓									Extract- *Phase 3*
cpk			✓	✓								Extract- *Phase 3*, Validate
$v_{i,j}$			✓									Encrypt
$ev_{i,j}$			✓	✓	✓				✓	✓	✓	Encrypt, Validate, Tally
$\mathsf{pp}_{i,j}$				✓								Validate
$\texttt{token}$			✓	✓								Validate
(a, b)			✓		✓						✓	Tally
v						✓			✓	✓		Decrypt

- *Uncoercibility*. It aiming to protect voters from coercion or undue influence. As per Table 3 where malicious voter has restricted component, malicious authorities are unable to force or manipulate voters into casting votes against their will, safeguarding the *uncoercibility* of the voting process.
- *Receipt-freeness* which ensures that $\mathsf{v}_{i,j}$ cannot prove or demonstrate their $v_{i,j}$ to others and only has $\texttt{token}$ to himself which is hashed in $\mathsf{Validate}$ algorithmic step. By restricting access to $(\texttt{token}, v_{i,j}, \mathsf{esk})$ keys, the GIBI-HE scheme guarantees that malicious authorities cannot link voters to their respective votes, ensuring *receipt-freeness*.
- *Robustness* and *fairness*. GIBI-HE scheme ensures that it can withstand various attacks under RO, or failures without compromising the *privacy* of the voters. The controlled distribution of keys for the VA and Ta makes it *robust* under RO by minimizing the potential vulnerabilities and mitigating the impact of malicious authorities for the Tally algorithm.
- *Universal verifiability* and *Individual verifiability*. The public components allow anyone to independently verify the integrity and correctness of the election using Eq. 4. Also, voters themselves can verify the accuracy of their cast votes and ensure they were counted correctly in tally or not using their unique $\texttt{token}$, ensuring *individual verifiability*. Malicious authorities can not really decrypt anything in this case.

As shown in Table 3, the key distribution for trusted and malicious authorities, it is clear that the GIBI-HE e-voting scheme effectively addresses the requirement of proposed e-voting system and ensures the integrity, transparency, and security of the e-voting process, providing voters with confidence and trust in the system. Now we discuss security model outlines the different potential threats and how our scheme addresses them.

6.1 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. 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. 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. 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. 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.

6.2 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. 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 using ZK.

Voter Acts as Another Registry Voter. 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 KeyGen.

6.3 Malicious Registry Manager RM_i

RM_i Act as Another Voter within same registry. 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. 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.

6.4 Malicious Voting Authority VA_i

VA_i plays role of verification for eligible voters using ZK. It does not have access to private keys of valid voters.

VA_i Acts as Registry Manager RM_i. 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. 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.

6.5 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. 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 $\mathsf{v}_{i,j}$.

7 Security Proof

The proposed GIBI-HE e-voting scheme satisfies the security requirements of *eligibility, privacy, anonymity unreusability, fairness, receipt-freeness, individual* and *universal verifiability, uncoercibility*, and protection against attack under RO model, provided that at least one of the authorities is honest.

7.1 Security Against Impersonation as Malicious Third Party and Malicious Voter.

Theorem 2. *The GIBI-HE scheme is (t, q, ε)-secure against impersonation in the RO model if the DL assumption of the vote holds such that:*

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

Proof. In this security game, Impersonator $\mathbb{I}$ who (t, q, ε) breaks the GIBI-HE scheme where t maximum number of corrupted or malicious authorities or participants that the scheme can tolerate while still maintaining its security guarantees, q is the maximum number of queries that an $\mathbb{I}$ can provide, and ε is quantifies the level of security or the maximum allowable advantage that an adversary can have in breaking the scheme. Challenger $\mathbb{C}$ acts as simulator which helps to find the value of DL assumption. $\mathbb{C}$ will simulate $\mathbb{I}$ as following with addition key size k:

- KeyGen. $\mathbb{C}$ 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 $\mathbb{I}$.
- **Training Phase**. $\mathbb{I}$ 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. The voters belong 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 $(\mathsf{epk}_i, \mathsf{cpk})$ to pass to $\mathbb{I}$.
    - $*$ Verify Query. $\mathbb{C}$ responses with ZK proof. In simulation, P takes input $(\mathsf{v}_{i,j}, \mathsf{vsk}_{i,j}, \mathsf{rpk}_i)$ whereas V takes input $(\mathsf{rpk}_i, \mathsf{v}_{i,j})$. P generates (E_i, X_i) and $\mathbb{C}$ throws a random challenge $c_i \in \mathbb{Z}^*_q$. Based on challenge P calculate the response $(Y_{i,1}, Y_{i,2})$ and send to V. Lastly V verifies $g_1^{Y_{i,1}} g_2^{Y_{i,2}} = X_i \left(\frac{E_i}{\hat{A}_i^{\beta_i}}\right)^{c_i}$. This ZK proof help to check *eligibility* of voter and all to procced further to cast ballot.
    - $*$ Encrypt Query. $\mathbb{I}$ casts vote $v_{i,j}$, encrypts $ev_{i,j}$, and passes to $\mathbb{C}$. It protects the integrity of the cast ballot using DE encryption method.
    - $*$ Validation Query. Using ZK proof, in a simulator P as $\mathsf{v}_{i,j}$ and V as VA communicates and based on random self challenge by $\mathsf{v}_{i,j}$, it generate response RES = $(a_{i,j}, b_{i,j}, \mathsf{pp}_{i,j}, T_{i,j}, \hat{T}_{i,j}, \texttt{token}, S_{i,j}, \hat{S}_{i,j})$, 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 $a^{\hat{a_i}}$ and send to $\mathbb{C}$. Only valid voters can decrypt the casted ballot using $a^{\hat{a_i}}$ 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 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 $\mathsf{vsk}_{i,j^*} = (\hat{s}^*_i, \beta^*_i)$ to $\mathbb{I}$.
  - $*$ Verify Query. When transcript will create even if not yet queried before as an Extract query. $\mathsf{v}^*_{i,j}$ as P participate in transcript and add in the set. VA^*_i will not be able to issue transcript for the already corrupted voter. $\mathsf{v}^*_{i,j}$ is targeted ID 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 ZK. 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. P takes input $(\mathsf{rpk}^*_i, \mathsf{v}^*_{i,j}, \mathsf{vsk}^*_{i,j})$ where the V takes input $(\mathsf{rpk}^*_i, \mathsf{v}^*_{i,j})$. P 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 P calculates $(Y^*_{i,1}, Y^*_{i,2})$ to VA^*_i as its response. Lastly VA^*_i verifies $\mathsf{v}^*_{i,j}$ of $g_1^{Y^*_{i,1}} g_2^{Y^*_{i,2}} = X^*_i \left(\frac{E^*_i}{\hat{A}_i^{\beta_i}}\right)^{c^*_i}$. 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 $\mathbb{C}$.
  - $*$ Validate 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.

– **Breaking Phase**. Impersonation phase where $\mathbb{I}$ acts as a cheating V 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 V to accept with non-negligible probability. Breaking phase calculates and satisfies *soundness* as follows:

$[r_{i,1}, c_1, E_i, Y_{i,1}]$ and $[r_{i,2}, c_2, E_i, Y_{i,2}]$ 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.

$$\begin{aligned}
g_1^{\beta_i} g_1^{\hat{s}_i} g_2^{\beta_i} g_2^{\hat{s}_i} &= g_1^{\widetilde{\beta}_i} g_1^{\widetilde{\hat{s}}_i} g_2^{\widetilde{\beta}_i} g_2^{\widetilde{\hat{s}}_i} \\
g_1^{\beta_i + a\hat{s}_i} g_2^{\beta_i + a\hat{s}_i} &= g_1^{\widetilde{\beta}_i + a\widetilde{\hat{s}}_i} g_2^{\widetilde{\beta}_i + a\widetilde{\hat{s}}_i} \\
g_1^{a\hat{s}_i} - g_1^{a\widetilde{\hat{s}}_i} g_2^{a\hat{s}_i} - g_2^{a\widetilde{\hat{s}}_i} &= g_1^{\widetilde{\beta}_i} - g_1^{\beta_i} g_2^{\widetilde{\beta}_i} - g_2^{\beta_i} \\
g_1^a g_2^a &= g_1^{(\widetilde{\beta}_i - \beta_i)(\hat{s}_i - \widetilde{\hat{s}}_i)} g_2^{(\widetilde{\beta}_i - \beta_i)(\hat{s}_i - \widetilde{\hat{s}}_i)} \\
aa &= \frac{\widetilde{\beta}_i - \beta_i}{\hat{s}_i - \hat{s}_i} \frac{\widetilde{\beta}_i - \beta_i}{\hat{s}_i - \hat{s}_i} \\
a &= \frac{\widetilde{\beta}_i - \beta_i}{\hat{s}_i - \hat{s}_i}
\end{aligned}$$

For probability distribution to prove *zero-knowledgeness* for $\mathbb{C}$, it is winning the game after solving the DL assumption. We shall successfully Extract *Phase 2* valid conversations to derive $(\beta_i, \hat{s}_i)$ and encrypt $ev_{i,j} = (a_{i,j}, b_{i,j})$ and later calculating with the probability $\varepsilon_{\text{GIBI-HE}} = (-\frac{1}{2^k} - \frac{1}{2^k})^l$ where The value of l determines how many times the expression is repeated and affects the final value of $\varepsilon_{\text{GIBI-HE}}$. Assume $\mathbb{C}$ solves the DL assumption 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 considering probability of $\mathbb{C}$ successfully solving the problem is equal to the probability of both events A and B happening together and $A|B$ joint probability represents the probability of winning the game by successfully solving the DL assumption by $\mathbb{C}$ while not aborting the computation B.

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

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

$$\varepsilon^{\text{DL}}_{\mathbb{G},\mathbb{C}}(k) \geq \left(\varepsilon_{\text{GIBI-HE}} - \frac{1}{2^k} - \frac{1}{2^k}\right)^l - \left(\frac{1}{2^k}\right)^l$$

$$\varepsilon^{\text{DL}}_{\mathbb{G},\mathbb{C}}(k) + \left(\frac{1}{2^k}\right)^l \geq \left(\varepsilon_{\text{GIBI-HE}} - \frac{1}{2^k} - \frac{1}{2^k}\right)^l$$

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

7.2 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 assumption of the GIBI-HE scheme and to impersonate as RM_i, respectively.

Theorem 3. *The* GIBI-HE *scheme above is* (t, q, ε)*-secure against impersonator as registry manager in the RO 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 $\mathbb{I}$.

– KeyGen. Similar to the proof described as KeyGen in proof of Theorem 2.
– **Training Phase**. $\mathbb{I}$ tries to send 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. $\mathsf{RM}_i \neq \mathsf{RM}^*_i$ and $\mathsf{v}_{i,j} \neq \mathsf{v}^*_{i,j}$ Outside malicious identity acts as RM_i and tries to give access and generate $\mathsf{vsk}_{i,j}$ for authorized voters.
  • **Case 1**. $\mathsf{RM}_i \neq \mathsf{RM}^*_i$ and $\mathsf{v}_{i,j} \neq \mathsf{v}^*_{i,j}$
    * 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 is 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.
    * Verification Query. In the simulator, RM_i act as the P and VA_i act as V. 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 $\mathsf{v}_{i,j}$.
    * Validate Query. Using ZK, simulator holds the communication script for $\mathsf{v}_{i,j}$ by generating self challenge to generate RES and passes to $\mathbb{I}$.

  * $\mathsf{Decrypt}$ Query. $\mathbb{I}$ will provide with (a, b) from the tally and forward to only eligibile 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 $\mathsf{v}_{i,j} = \mathsf{v}_{i,j}^*$.
  * $\mathsf{Extract}$ Query. It 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 2.
  * $\mathsf{Verificaton}$ Query. For *eligibility*, 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 P 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.
  * $\mathsf{Encrypt}$ Query. Similar to Case 2 $\mathsf{Encrypt}$ oracle in proof of Theorem 2.
  * $\mathsf{Validate}$ 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.
  * $\mathsf{Decrypt}$ Query. $\mathsf{v}_{i,j}^*$ discard from $\mathsf{Decrypt}$ 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.

– **Breaking Phase**. $\mathbb{I}$ 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 [2]. 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 to solve the DL hard assumption used in the scheme for GIBI-HE 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-HE scheme, the occurrence of aborts depends on the specific $ev_{i,j}$ used. However, the probability of aborts due to a hash collision is negligible. Thus, if an $\mathbb{I}$ is able to come up with valid $\mathsf{vsk}_{i,j}^*$ during the Extract 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 $\mathbb{I}$ 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 $\mathbb{I}$ has access to the private key used for GIBI and DE scheme.

[2] 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.

These proofs will show that our scheme satisfies all of the required security properties, including *eligibility, privacy, anonymity, unreusability, fairness, receipt-freeness, individual* and *universal verifiability, uncoercibility*, and *robust* 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 2 proves the semantic security of the scheme for security model 6.1 and 6.2, which means that an attacker who has access to the public parameters and the Encrypt oracle cannot distinguish between encryptions of two different votes; makes it *robust* in nature. Theorem 3 proves the security of the scheme for security model 6.3 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 6.4 and Ta_i 6.5, 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 $\mathbb{I}$ cannot distinguish between them, which implies the security of the scheme.
Thus, with the combination of Theorem 2 and Theorem 3, we can provide a convincing proof that the scheme is secure against malicious authorities in proposed GIBI-HE e-voting scheme under RO, and that simulators can be constructed for such attacks.

8 Efficiency Analysis

In this section, we evaluate the efficiency of previous related e-voting scheme and then determine the computational cost of the GIBI-HE scheme for each PPT algorithm. There are no IBI-based e-voting schemes, thus we explore the closest substitute: GS scheme for e-voting.
Yang et al. [38] 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 is candidate.
Next we consider the e-voting GS scheme by Malina et al. [27], which use ElGamal encryption and GS for verification. Bilinear Pairings (BP) are used 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 4, we consider alternative schemes in order to calculate the efficiency in $\mathcal{O}$. We consider the IBI scheme which is an efficient scheme in the presence of targeted ID.

Table 4. Comparison with other similar schemes

Work	**Assumption**	**Scheme**	**Encryption**	**Efficiency**	**Security**
Yang et al. [38]	DL	DS	DE	$\mathcal{O}(2t \times k)$	Standard
Malina et al. [27]	BP	GS	ElGamal	$\mathcal{O}(n \times k)$	unknown
Our work	DL	GIBI	DE	$\mathcal{O}(k\log n)$	RO

Legends: t is time of one exponentiation, k number of candidates, n is time of group operation, and big-$\mathcal{O}$ is complexity.

In Table 5, we calculate the computational cost for our new GIBI-HE e-voting scheme. We consider k candidates and n voters, KeyGen to $\mathsf{Validate}$ phase run $n \times k$ times. Tally is run n times for each VA. $\mathsf{Decrypt}$ is distributed and run once involving n voters. According to computational cost analysis, the GIBI-HE scheme

Table 5. Efficiency Analysis for the GIBI-HE Scheme.

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

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

is more efficient and secure than the other group DS and IBI schemes.

9 Future Work

We have several ideas for enhancing and improving our e-voting scheme. The full implementation are being worked on at the moment and shall be added to the paper. One possibility is the introduction of one-time-use or time-based ZK proof, 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.

10 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 identities through the use of a ZK protocol. proposed scheme is secure under various scenarios and *robust* in the RO model. GIBI-HE scheme maintain the voter *privacy* and *anonymity* throughout the e-voting process. The use of DE encryption in a group-like structure allows for secure multiparty communication and ensures *fairness*, *uncoercibility*, and *receipt-freeness* in the voting process.
The scheme also generates proof of ballot for each voter and allows for *individual* and *universal verifiability* through using partial shares in ZK proof. The use of partial shares for decryption makes the system independent of any central authority for vote decryption. The voters can not obtain information on partial tally. Thus, it satisfied requirement of *fairness* in our e-voting GIBI-HE scheme. The proposed scheme is secure under various scenarios and *robust* in the RO model for DL assumption. Therefore, the proposed 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`, accessed: 2010-09-30
2. Adida, B.: Helios: Web-based open-audit voting. In: USENIX security symposium. vol. 17, pp. 335–348 (2008)
3. Ateniese, G., Kamara, S., Katz, J., et al.: Proofs of storage from homomorphic identification protocols. In: Asiacrypt. vol. 9, pp. 319–333. Springer (2009)
4. Bellare, M., Namprempre, C., Neven, G.: Security proofs for identity-based identification and signature schemes. Journal of Cryptology **22**(1), 1–61 (2009)
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. Canard, S., Schoenmakers, B., Stam, M., Traoré, J.: List signature schemes. Discrete Applied Mathematics **154**(2), 189–201 (2006)
9. 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)
10. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM **24**(2), 84–88 (1981)
11. Chaum, D.: Blind signatures for untraceable payments. In: Advances in cryptology. pp. 199–203. Springer (1983)
12. 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)
13. Chaum, D., van Heyst, E., Pfitzmann, B.: Group signatures. International Conference on the Theory and Application of Cryptographic Techniques pp. 257–265 (1991)

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. Damgård, I., Fazio, N., Nicolosi, A.: Non-interactive zero-knowledge from homomorphic encryption. In: Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3. pp. 41–59. Springer (2006)
19. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE transactions on information theory **31**(4), 469–472 (1985)
20. 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)
21. 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)
22. Heng, S.H.: Design and analysis of some cryptographic primitives. Ph.D. thesis, Tokyo Institute of Technology (2004)
23. 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)
24. Katz, J., Wang, N.: Efficiency improvements for signature schemes with tight security reductions. In: Proceedings of the 10th ACM conference on Computer and communications security. pp. 155–164 (2003)
25. Keromytis, J.I.A., Yung, M.: Applied cryptography and network security
26. 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)
27. 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)
28. 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)
29. 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)
30. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Advances in Cryptology—CRYPTO'89 Proceedings 9. pp. 239–252. Springer (1990)
31. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Workshop on the theory and application of cryptographic techniques. pp. 47–53. Springer (1984)
32. Steffen, S., Bichsel, B., Baumgartner, R., Vechev, M.: Zeestar: Private smart contracts by homomorphic encryption and zero-knowledge proofs. In: 2022 IEEE Symposium on Security and Privacy (SP). pp. 179–197. IEEE (2022)

33. Tan, S.Y., Heng, S.H., Phan, R.C.W., Goi, B.M.: A variant of schnorr identity-based identification scheme with tight reduction. In: International Conference on Future Generation Information Technology. pp. 361–370. Springer (2011)
34. Vangujar, A., Chin, J.J., Tan, S.Y., Ng, T.: A hierarchical identity-based identification scheme without pairing. Malaysian Journal of Mathematical Sciences **13**, 93–109 (2019)
35. Vangujar, A., Ng, T., Chia, J., Chin, J., Yip, S.: Group identity-based identification: Definitions, construction and implementation
36. Vangujar, A.K., Ng, T.S., Chin, J.J., Yip, S.C.: Group identity-based identification: Definitions and construction. In: Cryptology and Information Security Conference 2020. p. 51
37. 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)
38. 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)
39. Yi, X., Paulet, R., Bertino, E.: Homomorphic Encryption and Applications. Springer Briefs in Computer Science, Springer (2014). `https://doi.org/10.1007/978-3-319-12229-8`, `https://doi.org/10.1007/978-3-319-12229-8`
40. Zhang, D., Liu, M., Yang, Z.: Zero-knowledge proofs of identity based on elgamal on conic. In: IEEE International Conference on E-Commerce Technology for Dynamic E-Business. pp. 216–223. IEEE (2004)
41. 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)