IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 1 April 2024, accepted 8 May 2024, date of publication 3 June 2024, date of current version 13 November 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3408670

APPLIED RESEARCH

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, (Member, IEEE)
School of Computer Science and IT, University College Cork, Cork 21, T12 R5CP Ireland
Corresponding author: Apurva K. Vangujar (a.vangujar@cs.ucc.ie)

The work was supported by Science Foundation Ireland under Grant 18/CRT/6222 and Grant 18/CRT/6223.

ABSTRACT This article 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 proofs to ensure the *anonymity* and *eligibility* of voters. The voter $\mathsf{V}_{i,j}$ is granted the authorization to cast a valid vote for a single candidate C_k. The use of distributed ElGamal provides *fairness* while the use of partial shares for decryption enables *individual* and *universal verifiability* without the need for a central authority. The proposed scheme is secure under various scenarios and *robust* in the random oracle model. The GIBI-HE scheme offers a promising solution for e-voting, providing a sustainable and accessible environment for voters while supporting the *unreusability* of votes and protecting the *privacy* of voters.

INDEX TERMS Identity-based identification, group identity-based identification, homomorphic encryption, e-voting, ElGamal, distributed ElGamal, discrete logarithmic assumption.

I. 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 modernise the voting process and increase 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 *privacy* and *anonymity* during the e-voting process. E-voting systems, compared to traditional paper-based elections, promise that election results will be calculated quickly with less chance of human error and will also reduce printing, distribution, and staffing costs over a long 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.

As of 2021, 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 *anonymity* respectively. The literature says that to keep the whole e-voting process safe, signature schemes [2] and encryption algorithms [3] are used together. However, this is not enough for the registration and validation of eligible voters, as can be seen in attacks proposed on e-voting

The associate editor coordinating the review of this manuscript and approving it for publication was Barbara Masucci.

schemes used in practice, such as the sVote system in Switzerland used by SwissPost [4]. To tackle this, we propose a solution using Identity-based Identification (IBI) scheme and Homomorphic Encryption (HE).

A Standard Identification (SI) scheme, which Adi Shamir first proposed in 1984, is a method for identifying people by using their distinctive identities, such as a name, date of birth, or other personal information. The SI scheme enables a Prover (P) to verify their ID with a Verifier (V) without disclosing any personal information. Boneh and Franklin [5] pioneered the ID-based encryption scheme that led to the flourishing of ID-based cryptography. In later years, Bellare et al. [6] constructed a more secure IBI scheme based on a Zero-Knowledge (ZK) proof that results in higher efficiency. In IBI schemes, 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 IDs of voters and ensure that they are eligible to vote. By satisfying the requirements 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 voters and makes e-voting systems *robust* [7]. Authentication protocols are important for users to prove their ID to access the encrypted data that is stored in the cloud. Numerous studies delve into the intricacies of e-auction and e-voting. The paper by McCarthy et al. [8] explores the underlying cryptographic principles shared between these two areas. Both fields have prominently employed methods like mix-nets, HE, and trapdoor bit-commitments. Additionally, using the Fiat-Shamir transformation, the paper introduces a novel scheme that draws inspiration from the Helios system. In literature today, we do not find many ZK proofs for HE schemes, except some focusing on proof of storage as an application [9], [10]. We do not find ZK proofs combined with partially HE cryptosystems like ElGamal. The ElGamal cryptosystem [11] is a homomorphic public-key encryption scheme that allows computation on encrypted data. Consequently, it can be used to encrypt ballots so that the vote is not disclosed to any party. There have been many works with other cryptosystems or variants of ElGamal for e-voting schemes, which we discuss below in Section I-B. In conjunction with ZK proof, the ElGamal cryptosystem is used to provide *fairness* to the tallying in e-voting scheme.

A. MOTIVATION

Apart from the ElGamal encryption algorithm, several other cryptographic algorithms, such as the Digital Signature (DS) and hash function, have been used in e-voting schemes, and the algorithms selected normally depend 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. Chaum et al. [12] proposed the initial GS scheme. and developed over time in the universal design that Boneh and Franklin [13] and Cocks [14] proposed. Canard et al. [15] introduced list signatures are a variant of GS that sets a limit on the number of signatures each group member may issue without involving the GM. The ability of a designated GM 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 [2], this is a significant disadvantage and violates the *privacy* of the voters.

On the other hand, the Group Identity-based Identification (GIBI) scheme by Vangujar et al. [16] does 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, the GIBI scheme may be more efficient than the 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 proof in the GIBI scheme for e-voting. Using a ZK proof can aid in the prevention of voter fraud by ensuring that only eligible voters can cast ballot. The use of ZK proofs, GIBI protects voter identities, ensuring anonymity and preventing any unauthorised link between voters and their cast ballots. This robust cryptographic technique is critical for maintaining voter confidentiality in a secure e-voting environment. 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 [17]. This can help simplify the voting process and reduce the burden on voters and election officials. The proposed scheme distributes the decryption process across multiple parties using DE encryption, eliminating the need for a TA and improving

security. This decentralised approach ensures that no single entity can compromise the election, thereby enhancing the overall trustworthiness of the e-voting system. E-voting is a major application for the exponential variant of ElGamal; besides, multiple voters use one machine to cast their vote in a voting system.There are several reasons why it will be beneficial to develop a novel e-voting scheme that is a combination of GIBI and HE:

1) IMPROVED SECURITY

The e-voting scheme can provide *privacy* and *anonymity* for voters by using the GIBI scheme. 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.

2) EASE OF USE

The 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.

3) CONFIDENTIALITY

ZK proofs are used in the e-voting scheme to ensure that only the eligible voter casts the ballot. The use of ZK proofs can help protect against vote buying and other forms of voter interference, as well as prevent the revelation of individual votes after the election.

4) INCREASED ACCESSIBILITY

The GIBI and DE schemes 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.

B. RELATED WORK

1) E-VOTING MODELS

E-voting schemes, first developed by Chaum [18], 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 are described as follows:

1) In the mix-net model proposed by Chaum [19], 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 literatures [20], [21], and [22].
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 [14] and Boneh and Franklin [13] are examples of schemes that use DS for e-voting. Chaum [23] presented one of the earliest ideas for the use of the 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. [24] and Benaloh [25] are later approaches for adopting the BS scheme in e-voting.
3) The HE-based model allows votes to be encrypted and counted without decryption in e-voting. Cramer et al. [26] uses an 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. [3] 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.

2) ZK PROOF

Ateniese et al. [10] introduced the framework for building homomorphic linear authenticators from the identification protocol by Shoup. Zhang et al. [27] feature ZK proof by using ID-based on ElGamal on conic and suggested eliminating the use of TA. A combination of non-interactive ZK and exponential ElGamal is provided by Steffan et al. [28] 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. [3] use ZK proofs to demonstrate the verifiability of ballots, the final tally, etc. by utilising proof of ZK, but do not provide the means to verify the ID of the voter before they enter the voting process.

3) ID-BASED SCHEMES

ID-based Cryptosystems (IBC) [29] 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 and Kim [30] presented the ID-BS approach using the IBC concept. Choon and Cheon [31] in 2003 introduced an innovative protocol for the ID-BS scheme. Kurosawa and Heng [32] proposed a method for transforming DS into the IBI scheme. After reviewing all the existing ID-BS schemes for the e-voting system, Vangujar et al. [16] give e-voting as an application of the GIBI scheme in their paper. In 2021,

once again Vangujar et al. [33] constructed GIBI using the Schnorr IBI [34] and Schnorr DS [35] scheme, and it was proven more secure for group-like structure. Malina et al. [2] proposed a GS scheme-based approach to e-voting with a GM 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 voters' *eligibility* by default. By providing additional efficiency, an IBI scheme streamlines and enhances the security of the voting process. The paper by Isa et al. [36] went into detail about the Estonian approach and focused on the importance of anonymous credentials in protecting voter *privacy* and *anonymity* using self-sovereign ID and ZK proof.

4) DISTRIBUTED ElGamal

The DE assumption was introduced in [37] and used first in the work by Adida et al. [22] following the Helios framework for e-voting. Since then, DE has been utilised in many e-voting schemes in different forms, including the Swiss sVote system [4], where it helped in encrypting the votes as tuples and scalar multiplication rather than exponentiation. In addition, Haines et al. [38] used codes and ZK proofs 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. [3] suggested using DE for rank-based e-voting. Each ballot is encrypted, and the cumulative homomorphic property of the exponential ElGamal is used to count all the votes. They use ZK proof to show the verifiability of ballots and the final tally. Blockchain-based platform-independent e-voting scheme by [39] ensures voters' privacy and the accuracy of the votes using HE, linkable ring signatures, and proofs of knowledge between the voter and the blockchain.

C. CONTRIBUTION

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

- We construct the modified Schnorr-like GIBI scheme to use as a building block for our GIBI-HE scheme, and we show the requirement of ZK proof in the GIBI scheme. We also prove the 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 schemes based on the hardness of the DL assumption. GIBI is 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 *privacy* and *anonymity* of the voters.
- A DL-based ZK proof is developed to authenticate eligible voters and allow them to cast their votes homomorphically encrypted.
- A novel scheme that uses DE for tallying all cast ballots and checks the validity of the votes with ZK proof makes it a *robust* e-voting scheme.
- A token is generated and given to the voter who cast a 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 the tallying phase. The combination of cryptographic primitives like GIBI, DE, and ZK proof make the e-voting process secure and trustworthy.
- The proposed e-voting scheme is secure under the RO model and efficient compared to existing DS 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 the 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.

D. PAPER ORGANIZATION

The paper is structured as follows. Section II introduces the mathematical notations and primitives used in the rest of the paper. Section III 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 the GIBI scheme. Section IV discusses the requirements, participants, system description, and GIBI-HE definition. Section V provides a detailed construction of the novel GIBI-HE scheme. Section VI gives key arrangements for trusted and malicious participants in our e-voting and also lists out possible security models. Section VII gives the proof under the RO model for our GIBI-HE scheme, and in Section VIII, we discuss the security and efficiency of this proposed scheme. The conclusion comes last, and Section IX highlights the future work in this area.

II. PRELIMINARIES

A. DISCRETE LOGARITHMIC ASSUMPTION

We consider the definition of DL assumption from [32] and [40] as follows:

Definition 1: Let $\mathbb{G}$ be a finite cyclic group of order q with a generator g. The DL assumption is defined as follows: given

$Y \in \mathbb{G}$ and g, find an integer x such that $0 \leq x < q$ and $Y = g^x$. The number x is referred to as the DL of Y to the base g.

B. DIGITAL SIGNATURE

The standard definition of DS scheme [32] is described as follows:

Definition 2: A DS scheme is denoted by (Gen, Sign, Verify) of polynomial-time algorithms called key generation algorithm, signing algorithm, and verification algorithm.

- Gen. On input 1^k, the algorithm produces a pair of matching public parameters (param) and public key and secret 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 $\sigma \leftarrow \mathsf{Sign}_{\mathsf{sk}}(m)$.

C. TRANSFORMATION OF DS TO THE IBI SCHEME

Any DS = (Gen, Sign, Verify) from Definition 2 can be used as a building block to construct a canonical IBI = (KeyGen, Extract, Verification). The transformation is given by Kurosawa and Heng [32], 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 give two outputs.

DS scheme $\rightarrow$ IBI scheme

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

Hence, we can compare that param = mpk and master-key = msk. The algorithm Extract is the 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 $d = \sigma$ 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 the IBI scheme, given by Kurosawa [41], 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 (ID, msk) as input 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 proof of DS to obtain the ZK proof of IBI is shown below:

1) CMT. P takes an input as $d = \sigma$ whereas V takes an input as param = mpk, ID = m.
2) CHA. V generates a random challenge $c \in \mathbb{Z}_q^*$ and passes to P.
3) RES. P calculates a response based on CHA and passes for verification to V.
4) V then decides whether to accept or reject based on a RES.

D. SECURITY MODEL FOR THE 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 KeyGen, a hashing function, Extract, and Verification using ZK proof. 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 [32]. We are adopting the simulator and probability distribution given in Vangujar et al. [42] for the breaking phase of our GIBI-HE scheme. The two phases of security models are elaborated in Table 1:

TABLE 1. Security model of the IBI scheme.

Training Phase
Initially, an impersonator creates a pair of public and private keys in the first algorithm of the 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 GIBI-HE schemes to prove it secure under the RO model for different security models in Section VII.

E. SCHNORR IBI SCHEME

The Schnorr DS scheme [43] is converted to the Schnorr IBI scheme using the transformation given in Section II-C considering Definition 3. The Schnorr IBI scheme is based on DL assumptions and has three PPT algorithms:

1) KeyGen. It takes 1^λ where λ is a 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 (mpk, msk), where mpk = $(\mathbb{G}, p, q, g, y, H)$ and 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 compute $X = g^d$ 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.

F. GROUP IDENTITY-BASED IDENTIFICATION SCHEME

GIBI scheme by Vangujar et al. [16] is constructed using the Schnorr IBI [34] and Schnorr DS [35] scheme and defined as transactions between the TA, the GM, different groups $(\mathsf{G}_1, \mathsf{G}_2, \ldots, \mathsf{G}_i, \ldots, \mathsf{G}_n)$, and group members.

1) KeyGen. TA sets up param and, after calculation, outputs the pair (mpk, msk).
2) Extract.
   a) *Phase 1*. Run by the TA, (mpk, msk) are taken as input, and the group key pair (gpk, gsk) is generated and passed to respective GM.
   b) *Phase 2*. Run by the GM for each group member who possesses an ID. To register as a member of the group, the ID is sent to the GM. Taking ancestor (gpk, gsk) as input, GM generates user keys (upk, usk) for the corresponding group member ID.
   c) *Phase 3*. Run by the GM. GM stores (ID, 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 (upk, σ_1, 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 member in the 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 a modified GIBI using Schnorr IBI and eliminate the use of signature σ from Verification *Phase 1* and *Phase 2* algorithms to make it more relevant for the e-voting system and only run ZK proof to check the *eligibility* of all voters.

G. HOMOMORPHIC ENCRYPTION

1) ElGamal

The popular ElGamal cryptosystem [11] 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, Encrypt, Decrypt) as follows:

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

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 $\cdot$ represent the corresponding group operations.

2) 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 [44] 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) Encrypt. For encrypting message m, it chooses a random $k_i \in \mathbb{Z}_q^*$ and computes $c_i = (a_i, b_i)$. 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) Decrypt. The user takes input (x_i, c_i) and outputs $m = b_i/a_i^{x_i}$ for one message. For multiple messages, DE requires each user to calculate and broadcast partial decryptions and then homomorphically combine to calculate the final tally to reveal the 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 the 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 $m = m_1 + \ldots + m_n$.

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

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

III. MODIFIED GIBI SCHEME AND ZK PROOF FOR PROPOSED E-VOTING

A. REQUIREMENT OF ZK IN GIBI SCHEME

We consider the requirement of ZK in DS proposed by Kurosawa and Heng [32], eliminate the need for signatures in ZK, and show a canonical (three-move) ZK interactive proof system on knowledge of ID for GIBI scheme using the transformation given in Section II-C 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 (mpk, ID). The secret input to P is d. Let ZK = (CMT, CHA, 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 the GIBI scheme has a Δ-ZK protocol if there exists a canonical protocol (P, V) or any (mpk, 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, which is later used in the construction of the novel GIBI-HE scheme for our e-voting scheme. We show our GIBI-HE scheme satisfies these properties in the simulator and makes the scheme secure under the RO model.

B. MODIFIED GIBI SCHEME USING SCHNORR IBI

We initialise the GIBI using the Schnorr IBI and eliminate the requirement of the Schnorr DS given by Vangujar et al. [33]. We construct a new scheme, GIBI, using the construction given in Section II-F. It is obvious that the signature part of the original scheme will be discarded in the modified GIBI, making it a purely identification scheme. The Group-IBI 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 $(\mathrm{ID}_1, \mathrm{ID}_2, \ldots, \mathrm{ID}_i, \ldots, \mathrm{ID}_m)$ to prove their ID as a valid group member. We refer to [34]'s tight Schnorr IBI variant to construct the Group-IBI, taking into consideration the hard DL assumption. Figure 1 gives the generic arrangement of participants for the modified GIBI scheme and also shows how we expand the same arrangement for our GIBI-HE e-voting 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 $H : \{0,1\}^* \times \mathbb{G} \in \mathbb{Z}_q^*$. 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 calculates $s = t + x\alpha$ where $\alpha = H(\mathsf{GM}, A, B, y_1, y_2)$. TA generates the pair where the group public keys $\mathsf{gpk} = (\mathsf{GM}, g_1, g_2, y_1, y_2)$ and group secret keys $\mathsf{gsk} = (\alpha, s)$ 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 pairs to ID_i and does not store usk_i.
3) Verification. ID_i as P performs the transaction with a GM as V. The ZK proof is 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 selects 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 $g_1^{y_{i,1}} g_2^{y_{i,2}} = X \cdot (EF/y_{i,1}^{\beta} y_{i,2}^{\beta})^c$.

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

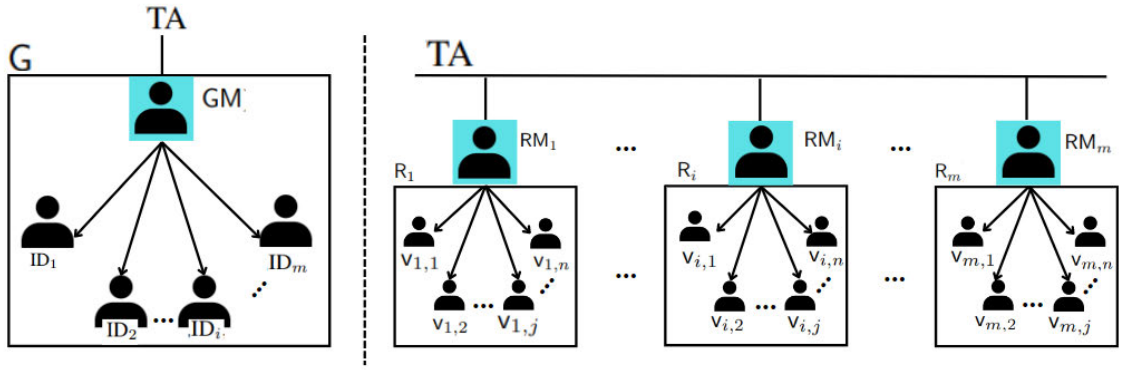

FIGURE 1. Generic arrangement of participants for modified GIBI and GIBI-HE scheme.

C. SECURITY ANALYSIS OF GIBI SCHEME

Depending on the scenario, multiple security models can be mentioned, but here we will provide examples of malicious behaviour 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 assumption 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 1: 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) 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) 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 or conversation for $\mathbb{I}$; else, $\mathbb{C}$ runs an 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 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 $\mathsf{usk}_{\mathrm{ID}^*_{\mathbb{I}}}$ 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 assumption used in the scheme and win in the security game.

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

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 modified GIBI scheme. We consider the same security proof simulator for different scenarios under RO and prove it secure if the DL assumption holds for the GIBI-HE scheme.

IV. OUR E-VOTING SYSTEM

A. REQUIREMENTS

E-voting 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, privacy*, and *anonymity* are assured by the use of the GIBI scheme. *Robustness, individual* and *universal verifiabilty* are given by ZK proofs in decrypting votes. *Fairness* is provided using the HE tally portion. *Unresuability* is satisfied in Validate phase where eligible vote can cast only one ballot.

B. PARTICIPANTS

- **Trusted Authority.** The TA is responsible for setting up the system.
- **Registry.** The Registry R is in charge of managing the authorization phase. Each registry will have Registry Manager RM in such a way that $\mathsf{RM} = (\mathsf{RM}_1, \mathsf{RM}_2, \ldots, \mathsf{RM}_i, \ldots, \mathsf{RM}_m)$. One registry will include RM and a set of voters v. For example, for $\mathsf{R}_1 = (\mathsf{RM}_1, \mathsf{v}_{1,1}, \mathsf{v}_{1,2}, \ldots, \mathsf{v}_{1,j}, \ldots, \mathsf{v}_{1,n})$, RM supervises the generation of private and public keys for each voter in v. The arrangement of TA, RM, and voters is shown and compared with the modified GIBI scheme in Figure 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

TABLE 2. Requirements for E-voting Scheme.

Requirement	Explanation
Privacy	All votes remain confidential.
Eligibility	Only authorised individuals can vote.
Unreusability	Each eligible voter is limited to one vote. It is against the rules to vote by proxy.
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.
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.

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 Figure 1.

- **Candidates.** There are a set of candidates $\mathsf{C} = (\mathsf{C}_1, \mathsf{C}_2, \ldots, \mathsf{C}_k, \ldots, \mathsf{C}_l)$, nominated by a political party, an organisation, 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.** $\mathsf{VA} = (\mathsf{VA}_1, \mathsf{VA}_2, \ldots, \mathsf{VA}_i, \ldots, \mathsf{VA}_m)$ where the number of R is equal to the number of VA in our e-voting system. The VA manages ZK proofs for all voters in $\mathbf{v}$. VA allows only eligible voters 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 the final tally to the 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 election statistics, but nobody can alter their content.

C. SYSTEM DESCRIPTION

The proposed GIBI-HE e-voting scheme includes five major phases described as follows and shown in Figure 2: The setup, registration, and authorising phases are shown in Figure 3; the voting and tallying phases are shown in Figure 4.

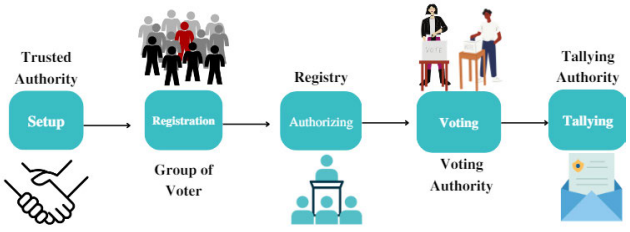

FIGURE 2. Flow of proposed E-voting scheme.

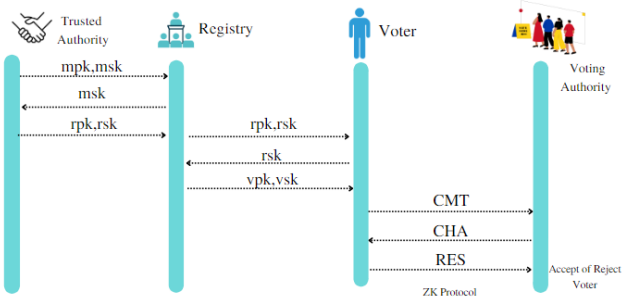

FIGURE 3. Setup, registration, and authorizing phase of proposed E-voting Scheme.

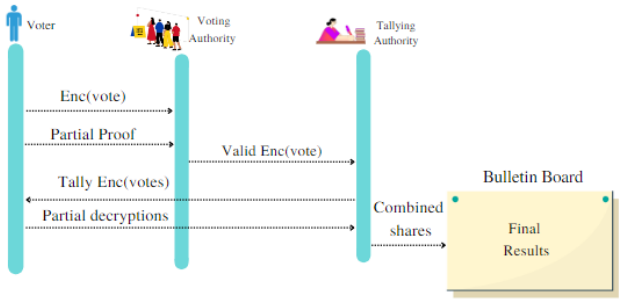

FIGURE 4. Voting and tallying phase of proposed E-Voting Scheme.

1) **Setup**. In this phase, the TA sets up the key for the group registry manager RM, then the registry setting up key for all voters v.
2) **Registration**. This may involve Registry R checking voter's 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 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 aggregating all the votes to determine the election's outcome. This involves decrypting the votes, confirming the accuracy of the vote count, declaring the decrypted results by VA, and listing them on the BB.

D. DEFINITION OF GIBI-HE SCHEME

The proposed GIBI-HE scheme consists of seven PPT algorithms such that GIBI-HE = (KeyGen, Extract, Verification, Encrypt, Validate, Tally, Decrypt) and is constructed using Definition 3, newly constructed Schnorr-like modified GIBI from Section III-B, and DE from Definition 4 run among TA, v, R, VA, C, Ta, and BB.

1) KeyGen. Using the security parameter 1^λ and the secret key chosen by the TA, a pair of master public and secret keys (mpk, msk) is generated as output.
2) Extract. Considering the generic scenario for the e-voting scheme as follows:
Phase 1. For RM_i, TA calculates registry public and secret key (rpk_i, rsk_i) using ancestor msk.
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.
Phase 3. Using the key of the voter for encryption, the common public key cpk is generated by multiplying all the voter's vpk, and cpk is sent back to all voters in v for encryption.
3) 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 the value and sends it 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 a response; further passes to VA_i.
   d) VA_i accepts $\mathsf{v}_{i,j}$ as an eligible voter for the voting process if and only if it verifies the final equation using DL-tuple.
4) 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 $ev_{i,j} = (\mathsf{Enc}(b_1), \ldots, \mathsf{Enc}(b_k), \ldots, \mathsf{Enc}(b_l))$, where ballot $b_k = 0$ or 1 where $1 \leq k \leq l$ depending on which candidate the ballot was cast for.
5) 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 1 using ZK proof. For 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 collects the encrypted votes from the voters and tally them using the homomorphic additive property.
7) 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.

V. CONSTRUCTION OF GIBI-HE E-VOTING SCHEME

GIBI-HE is not a straightforward combination of the existing GIBI and DE schemes. We attempted to construct GIBI, taking into account that we will use DE later in the scheme and maintaining both under the DL assumption. It was difficult to implement *Phase 3* in the Extract algorithm of GIBI. DE is always constructed using g, but for this construction, we used g_1 and g_2. It was difficult to verify equations in the Validate phase, but introducing randomness in $\hat{T}_{i,j}$ helped us verify equations. Consideration is given to the group, voter $\mathsf{v}_{i,j}$, for the construction of algorithmic steps with distinct steps. We can easily extend construction by bringing together all voters. The GIBI-HE is a new e-voting scheme that combines the use of Schnorr IBI II-E with GIBI II-F and DE Definition 4. We first constructed modified GIBI in Section III-B, and we will be using the modified GIBI scheme as the building block for the GIBI-HE scheme, which is proven secure under the RO model using the DL assumption. The 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 utilises the same underlying assumptions (such as the difficulty of the DL assumption) and key generation techniques, with the addition of a *Phase 3* in the Extract algorithm. A new type of e-voting scheme is based on the six PPT algorithms for five phases: setup, registration, authorization, voting, and tallying.

A. 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 a selects random integer $x \in \mathbb{Z}_q^*$ to compute $y_1 = g_1^{-x}$ and $y_2 = g_2^{-x}$ where generator $g_1, g_2 \in \mathbb{G}$. Computes 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 III-B 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.

B. REGISTRATION

2) Extract.
Phase 1: It is run by TA. TA will be responsible for generating key pairs for all RM. There are multiple

Registries ($\mathsf{R}_1, \mathsf{R}_2, \ldots, \mathsf{R}_i, \ldots, \mathsf{R}_m$), which has Registry Manager ($\mathsf{RM}_1, \mathsf{RM}_2, \ldots, \mathsf{RM}_i, \ldots, \mathsf{RM}_m$) and group of voters v. Take the sample as $\mathsf{R}_1 = (\mathsf{RM}_1, \mathsf{v}_{1,1}, \mathsf{v}_{1,2}, \ldots, \mathsf{v}_{1,j}, \ldots, \mathsf{v}_{1,n})$. A more generic way could be$\mathsf{R}_i = (\mathsf{RM}_i, \mathsf{v}_{i,1}, \mathsf{v}_{i,2}, \ldots, \mathsf{v}_{i,j}, \ldots, \mathsf{v}_{i,n})$, and Figure 1 shows this arrangement for our e-voting scheme. TA takes $(\mathsf{mpk}, \mathsf{msk}, \mathsf{RM}_i)$ as input and 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 key $\mathsf{rsk}_i = (\alpha_i, s_i)$ to RM_i.

Phase 2: This phase is run by RM_i. Consider voter $\mathsf{v}_{i,j}$ from the set of all v who want 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 an addition to modified GIBI, and it shows that the encryption keys for every voter are generated. For every voter $\mathsf{v}_{i,j}$, RM_i uses the 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 a common public key $\mathsf{cpk} = \prod_{i=1}^{n} \mathsf{epk}_i$. Distributes cpk to all v. It will be useful for validating encrypted votes.

C. AUTHORIZATION

3) Verification. Each voter has to prove their *eligibility* to VA to cast their ballot. $\mathsf{v}_{i,j}$ as P performs the transaction with a VA_i as V. The ZK proof 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:

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

Once the vote has been authenticated with ZK proof, it is only validated once afterward. If a voter attempts to vote again, he or she must go through ZK again, and if it has already been authenticated, it cannot be authenticated again.

D. VOTING

4) Encrypt. In this phase, eligible voters are given the 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 encrypts it $ev_{i,j} = \mathsf{Enc}(v_{i,j})_{\mathsf{cpk}}$.The 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 Equation 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 utilises a pair of generators (g_1, g_2).
5) Validate. This phase is run by VA_i and it checks the validity of the ballots for the final tally, or it also discards invalid votes. We use the proof of ZK using partial shares to check 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 calculates 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 a 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 $\mathtt{token} = H\left(\mathsf{pp}_{i,j} || T_{i,j} || \hat{T}_{i,j}\right)$ and it can be later useful for voters to check if their ballot is being counted or not in the final tally.
      - $S_{i,j} = \hat{r}_{i,j_1} \cdot \mathtt{token} + \hat{c}_{i,j}$ and $\hat{S}_{i,j} = \hat{r}_{i,j_2} \cdot \mathtt{token} + \hat{c}_{i,j}$.
      - Based on the above calculations, the response RES is generated where $\mathsf{RES}=(a_{i,j}, b_{i,j}, \mathsf{pp}_{i,j}, T_{i,j}, \hat{T}_{i,j}, \mathtt{token}, S_{i,j}, \hat{S}_{i,j})$.

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

To verify the validity of a vote in our e-voting scheme, i.e., to check if the sum of the vote equals 1, a simple validation procedure is carried out by adding the individual components and ensuring that the resulting sum equals 1. `token` makes votes anonymous and ensures *privacy*. If Equation 3 verification holds equal, then VA_i passes ballots to the final tally, or else it will be discarded.

If 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 is not included in the final tally. This helps ensure the *unresuability* of our GIBI-HE e-voting scheme.

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

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

E. 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 does not authorise the display of the 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 the sum of all the cast ballot valid votes by voter under 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 $\mathsf{Enc}\left(\prod_{j=1}^{n} v_{i,j}\right)$ as defined as $\left(g_1^{\sum_{j=1}^{n}\hat{r}_{i,j_1}} g_2^{\sum_{j=1}^{n}\hat{r}_{i,j_2}}, g_1^{\sum_{j=1}^{n}v_{i,j}} g_2^{\sum_{j=1}^{n}v_{i,j}} \cdot \hat{A}_i^{\sum_{j=1}^{n}\hat{r}_{i,j_1}} \hat{B}_i^{\sum_{j=1}^{n}\hat{r}_{i,j_2}}\right) = (a_{i,j}, b_{i,j})$.

7) Decrypt. As there is no central secret or decryption key, decryption involves two rounds of communication, as follows: $\mathsf{Enc}\left(\prod_{n=1}^{i} v_{m,n}\right) = (a_{m,n}, b_{m,n})$
   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 send them back to the VA_i.
   b) Then, by Equation 2, the partial shares are combined together, and the DL assumption is performed to retrieve the final tally. VA_i to decrypt the combined vote v.

$$\frac{b}{\prod_{i=1}^{n} a^{\hat{a}_i}} = \frac{b}{a^{\hat{a}_1+\ldots+\hat{a}_n}} = v. \quad (4)$$

Finally, the total number of votes v can be revealed by computing the DL assumption. Any voter can check if their vote is counted and included in the final tally by decrypting and checking the final result on BB. The protocol provides cast-as-intended verifiability without completely trusting the voter's device. It ensures *individual verifiability*. Additionally, any eligible voter can check whether the election is fair or not, which satisfy the requirement of *universal verifiability*.

VI. 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 verification of voters will help satisfy the 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 `token` as shown in Table 3 and 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.

- *Robustness* and *fairness*. GIBI-HE scheme ensures that it can withstand various attacks under RO model, 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 Equation 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 `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 requirements of the 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. Below, we briefly outline the different potential security threats to our scheme and discuss how our scheme addresses them. A full security proof is provided in Section VII.

A. 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.

1) TP ACTS AS REGISTRY MANAGER

A malicious TP may try to impersonate RM_i to allow registration without checking the 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.

2) TP ACT AS VOTER

A malicious TP may try to impersonate a voter $\mathsf{v}_{i,j}$ by forming his 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.

3) TP ACT AS VOTING AUTHORITY

A malicious TP may try to impersonate a VA_i to allow anyone to give the right 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 the respective $\mathsf{vsk}_{i,j}$. It is impossible for VA_i to eliminate ZK in this process and proceed with voting.

4) TP ACT AS TALLYING AUTHORITY

A malicious TP may try to impersonate a Ta_i to tally all encrypted votes. However, this impersonation is infeasible unless he possesses the capability to decrypt or manipulate the original votes. Furthermore, he cannot execute any operations on the encrypted votes without access to the corresponding partial share from the voters.

B. MALICIOUS VOTER

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

1) 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 the next step using ZK.

2) VOTER ACTS AS ANOTHER REGISTRY VOTER

A 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 the randomness introduced in KeyGen.

C. MALICIOUS REGISTRY MANAGER RM_I

1) 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 voters to obtain cpk for the next ZK proof.

2) 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 sets of voters.

D. MALICIOUS VOTING AUTHORITY VA_I

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

1) 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.

2) VA_I ACTS AS TALLYING MANAGER TA_I

VA_i acts as Ta_i by generating a tallying of encrypted votes for the voter and running the partial decryption phase.

TABLE 3. Key distribution in GIBI-HE E-voting scheme.

	Trusted Participants						Malicious Participants					
Keys	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
token			✓	✓								Validate
(a, b)			✓		✓						✓	Tally
v						✓			✓	✓		Decrypt

Impersonator VA_i playing role of Ta_i should be collected cipher text and satisfy correctness proof before displaying result on BB.

E. 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.

1) Ta_i ACTS AS VA_i

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

VII. SECURITY PROOF

The proposed GIBI-HE e-voting scheme satisfies the security requirements of *eligibility, privacy, anonymity, unreusability, fairness, individual* and *universal verifiability*, and protection against attack under the RO model, provided that at least one of the authorities is honest. Following is the security proof for the security model presented in Section II described in Table 1.

A. 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:

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

Proof 2: In this security game, the impersonator $\mathbb{I}$ who $\hat{\text{A}}$ (t, q, ϵ) breaks the GIBI-HE scheme, where t is the 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 quantifies the level of security or the maximum allowable advantage that an adversary can have in breaking the GIBI-HE scheme. Challenger $\mathbb{C}$ acts as a simulator, which helps to find the value of the DL assumption. $\mathbb{C}$ will simulate $\mathbb{I}$ as follows 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 $(\mathsf{q}_0, \ldots, \mathsf{q}_i, \ldots, \mathsf{q}_m)$ 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; $\mathbb{C}$ will run the further transcript using vsk. 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.
- KeyGen. $\mathbb{C}$ obtains $\mathsf{mpk} = (\mathbb{G}, q, g_1, g_2, y_1, y_2, H)$ by giving input 1^λ and passes mpk to $\mathbb{I}$.
- **Training Phase**. $\mathbb{I}$ can issue multiple sets of queries $(\mathsf{q}_0, \ldots, \mathsf{q}_i, \ldots, \mathsf{q}_m)$ where q_i for $\mathsf{v}_{i,j}$ and there are m queries in total. In the training phase, $\mathbb{I}$ attempts to learn from $\mathbb{C}$ and will try to forge vsk; $\mathbb{C}$ will run the further transcript using vsk. GIBI-HE is considered an advanced version of GIBI in combination with HE without the use of pairing in it for a set of voters.

  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 $\mathbb{I}$ is 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 the challenge, P calculates

the response $(Y_{i,1}, Y_{i,2})$ and sends it 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 helps to check the *eligibility* of the voter and all to proceed further to cast the 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.

Validation Query. Using ZK proof, in a simulator, P as $\mathsf{v}_{i,j}$ and V as VA communicate, and based on a random self-challenge by $\mathsf{v}_{i,j}$, it generates a 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 $\mathbb{I}$.

Decrypt Query. $\mathbb{I}$ will provide (a, b) from tallying, and it forwards to $\mathsf{v}_{i,j}$ to calculate the partial share $a^{\hat{a}_i}$ and send it to $\mathbb{C}$. Only valid voters can decrypt the cast 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 R_i where all RM_i are defined as parent and voters in RM_i child node according to hierarchy under that respective R_i. vsk helps to generate vsk of child-eligible voter. A child node's vsk is generated only if it has vsk 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 a transcript is created, even if it has not yet been queried before as an Extract query. $\mathsf{v}^*_{i,j}$ as P participates in the transcript and adds to the set. VA^*_i will not be able to issue a transcript for the already corrupted voter. $\mathsf{v}^*_{i,j}$ is the targeted ID of the voter, and VA^*_i needs to verify it. $\mathsf{v}_{i,j} = \mathsf{v}^*_{i,j}$, $\mathbb{I}$ acts as the cheater VA^*_i and $\mathbb{C}$ does not have the user secret key of $\mathsf{v}^*_{i,j}$, however, it needs to be created 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 transcripts as many times as the 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 a random challenge. $c_i^* \in \mathbb{Z}_q^*$ where corresponds to $\mathsf{v}^*_{i,j}$. On the basis of the 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 provides *integrity* for valid voters.

Encrypt Query. When $\mathbb{I}$ casts vote $v^*_{i,j}$ and encrypts $ev^*_{i,j}$ to pass to $\mathbb{C}$.

Validate Query. Running a validation query using voter-generated $\hat{c}^*_{i,j}$ aborts the condition and fails to receive RES. This stage requires $\mathsf{v}^*_{i,j}$ to verify the correctness of their vote without revealing the contents of the vote to anyone else, which justifies the property of *individual verifiability*.

Decrypt Query. $\mathsf{v}^*_{i,j}$ fails to run the decrypt query since $\mathsf{v}^*_{i,j}$ does not hold any combined ciphertext and will fail to compute the partial share for $\hat{a}_i^*$. Anyone can independently verify that the votes were accurately cast, collected, tallied, and the results were correctly computed at this stage, which means they satisfy the property of *universal verifiability* allow for public scrutiny, and help 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**. $\mathbb{C}$ receives $[r_{i,1}, c_1, E_i, Y_{i,1}]$ and $[r_{i,2}, c_2, E_i, Y_{i,2}]$ from $\mathbb{I}$, it is given that $c_1 \neq c_2$, meaning the two sets of information are distinct. From the received information, the $\mathbb{C}$ calculates two values, $\widetilde{\hat{s}}_i$ and $\widetilde{\beta}_i$, by subtracting the Y values and dividing by the difference of the c values. If the calculated values $\widetilde{\hat{s}}_i$ and $\widetilde{\beta}_i$ are equal to the original values $\hat{s}_i$ and β_i, then the $\mathbb{C}$ party aborts the protocol.
  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.

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

For probability distribution to prove *zero-knowledgeness* for $\mathbb{C}$, it is winning the game after solving the DL assumption. We shall successfully extract from 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 calculate with the probability $\epsilon_{\text{GIBI-HE}} = (-\frac{1}{2^k} - \frac{1}{2^k})^l$ where the value of l determines how many times the expression is repeated and affects the final value of $\epsilon_{\text{GIBI-HE}}$. Assume $\mathbb{C}$ solves the DL assumption, which computes the correct value of an event as A where it accepts $ev_{i,j}$ and the not aborting event as B. Winning probability can be given considering the 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.

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

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

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

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

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

B. 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 DL for 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 assumption for GIBI-HE scheme holds.

Proof 3: 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 the proof of Theorem 2.
- **Training Phase**. $\mathbb{I}$ tries to send queries to check fragile nature of simulator. Query set contains $(\mathsf{q}_0, \mathsf{q}_1, \ldots, \mathsf{q}_m)$ 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 proof 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 proof 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 *unreusability* of the votes. Validate Query. Using ZK, simulator holds the communication script for $\mathsf{v}_{i,j}$ by generating self challenge to generate RES and passes to $\mathbb{I}$.
  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 $\mathbb{C}$.
  Case 2. $\mathsf{RM}_i = \mathsf{RM}^*_i$ and $\mathsf{v}_{i,j} = \mathsf{v}^*_{i,j}$.
  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 the proof of Theorem 2.
  Verificaton Query. For *eligibility*, ZK provide authorization for eligible voter $\mathsf{v}^*_{i,j}$ right to cast ballot. ZK can be proven using a 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.
  Encrypt Query. Similar to Case 2 Encrypt oracle in the proof of Theorem 2.
  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 the final tally and outcome and fails to generate RES^*. This satisfy the property of *individual verifiability* where $\mathsf{v}^*_{i,j}$ to verify that vote is recorded or not by later checking on BB.
  Decrypt Query. $\mathsf{v}^*_{i,j}$ discard from Decrypt oracle since it will not have (a, b) to satisfy DE assumption and eventually can not calculate its ballot $v_{i,j}$. In case of authorized voters, the case is opposite and so *universal verifiability* to be satisfied by any valid $\mathsf{v}^*_{i,j}$ under RM^*_i verify the accuracy of the election outcome.
- **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 win in the security game. We now analyze the probability of aborts during the whole simulation process for malicious RM^*_i.

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

  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.

These proofs will show that our scheme satisfies all of the required security properties, including *eligibility, privacy, anonymity, unreusability, fairness, individual* and

[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.

TABLE 4. Comparison with other similar schemes.

Work	Assumption	Scheme	Encryption	Efficiency	Security
Yang *et al.* [3]	DL	DS	DE	$\mathcal{O}(2t \times k)$	Standard
Malina *et al.* [2]	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 is number of candidates, n is time of group operation, and $\mathcal{O}$ is complexity.

universal verifiability, 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 models VI-A and VI-B, which means that an attacker who has access to the public parameters and the Encrypt oracle cannot distinguish between encryptions of two different votes, making it *robust* in nature. Theorem 3 proves the security of the scheme for security model VI-C in the presence of malicious authorities using the DL assumption, meaning that even if some authorities behave maliciously by modifying encrypted votes or sending incorrect votes, the scheme remains secure and maintains its *privacy* on all levels.

To prove the existence of simulators for malicious VA_i VI-D and Ta_i VI-E, 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 the proposed GIBI-HE e-voting scheme under RO and that simulators can be constructed for such attacks.

VIII. EFFICIENCY ANALYSIS

In this section, we evaluate the efficiency of previous related e-voting schemes 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: the GS scheme for e-voting.

Yang et al. [3] 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 proof, and the signature scheme is $2t \times k \times L_P \times 5t \times n_c \times L_P + 2t + t$, where t is the time of one exponentiation, L_P is the total available points, and k is the candidate.

Next, we consider the e-voting GS scheme by Malina et al. [2], which use ElGamal encryption and GS for verification. Bilinear Pairing (BP) is used for this scheme, which is 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 increase the scheme's efficiency and decrease its computation overhead. In Table 4, we present a comprehensive analysis of both efficiency and security, comparing the existing scheme with our proposed solution.In Table 5, we calculate the computational cost for our new GIBI-HE e-voting scheme. We consider k candidates and n voters in the KeyGen to Validate phase run $n \times k$ times. Tally is run n times for each VA. Decrypt is distributed and run once, involving n voters. According to computational cost analysis, the GIBI-HE scheme is more efficient and secure than the other group DS and IBI schemes.

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: k is number of candidates, n is number of 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.

IX. FUTURE WORK

E-voting schemes are an actively researched area, and the underlying techniques and their use and combination constantly improve over time. Several potential future avenues are possible to further enhance and improve the e-voting scheme presented in this 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, it could be of interest to explore 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.

X. 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 the ZK proof. The proposed scheme is secure under various scenarios and *robust* under the RO model. The GIBI-HE scheme maintains 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* 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 the partial tally. Thus, it satisfied the requirement of *fairness* in our e-voting GIBI-HE scheme. Therefore, the proposed GIBI-HE e-voting scheme is a novel and secure method for conducting elections electronically, bringing widespread use of e-voting technology even closer. Although the computational cost is somewhat higher due to the distributed system, we plan to reduce this cost in the future and enhance the scheme's resilience to vulnerabilities. Moreover, we will evaluate whether the adoption of post-quantum cryptography necessitates re-examining all underlying assumptions to ensure long-term security for e-voting.

REFERENCES

[1] *International Institute for Democracy and Electoral Assistance*. Accessed: Sep. 30, 2010. [Online]. Available: https://www.idea.int/data-tools/question-view/742

[2] L. Malina, J. Smrz, J. Hajny, and K. Vrba, "Secure electronic voting based on group signatures," in *Proc. 38th Int. Conf. Telecommun. Signal Process.*, Prague, Czech Republic, Jul. 2015, pp. 6–10.

[3] X. Yang, X. Yi, S. Nepal, A. Kelarev, and F. Han, "A secure verifiable ranked choice online voting system based on homomorphic encryption," *IEEE Access*, vol. 6, pp. 20506–20519, 2018.

[4] T. Haines, S. J. Lewis, O. Pereira, and V. Teague, "How not to prove your election outcome," in *Proc. IEEE Symp. Secur. Privacy*, San Francisco, CA, USA, May 2020, pp. 644–660.

[5] D. Boneh and M. K. Franklin, "Identity-based encryption from the Weil pairing," *SIAM J. Comput.*, vol. 32, no. 3, pp. 586–615, 2003, doi: 10.1137/S0097539701398521.

[6] M. Bellare, C. Namprempre, and G. Neven, "Security proofs for identity-based identification and signature schemes," *J. Cryptol.*, vol. 22, no. 1, pp. 1–61, Jan. 2009.

[7] X. Yi, R. Paulet, E. Bertino, X. Yi, R. Paulet, and E. Bertino, *Homomorphic Encryption*. Berlin, Germany: Springer, 2014.

[8] A. McCarthy, B. Smyth, and E. A. Quaglia, "Hawk and aucitas: E-auction schemes from the helios and civitas e-voting schemes," in *Proc. 18th Int. Conf.*, Christ Church, Barbados. Berlin, Germany: Springer, Mar. 2014, pp. 51–63.

[9] I. Damgård, N. Fazio, and A. Nicolosi, "Non-interactive zero-knowledge from homomorphic encryption," in *Proc. 3rd Theory Cryptography Conf.*, New York, NY, USA. Berlin, Germany: Springer, Mar. 2006, pp. 41–59.

[10] G. Ateniese, S. Kamara, and J. Katz, "Proofs of storage from homomorphic identification protocols," in *Advances in Cryptology—ASIACRYPT*, vol. 9. Berlin, Germany: Springer, 2009, pp. 319–333.

[11] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 4, pp. 469–472, Jul. 1985.

[12] D. Chaum, E. van Heyst, and B. Pfitzmann, "Group signatures," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 1991, pp. 257–265.

[13] D. Boneh and M. K. Franklin, "Predicate encryption and simple definitions of secrecy," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 1997, pp. 319–335.

[14] C. C. Cocks, "Covert security of key-agreement protocols," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2001, pp. 260–275.

[15] S. Canard, B. Schoenmakers, M. Stam, and J. Traoré, "List signature schemes," *Discrete Appl. Math.*, vol. 154, no. 2, pp. 189–201, Feb. 2006.

[16] A. K. Vangujar, T.-S. Ng, J.-J. Chin, and S.-C. Yip, "Group identity-based identification: Definitions and construction," in *Proc. Cryptol. Inf. Secur. Conf.*, 2020, p. 51.

[17] Y. Wang and G.-J. Chen, "Secure and efficient e-voting system based on distributed ELGamal," in *Proc. Int. Conf. Comput. Sci. Softw. Eng.*, 2006, pp. 399–404.

[18] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981.

[19] D. L. Chaum, "Elections with unconditionally-secret ballots and disruption equivalent to breaking rsa," in *Proc. Workshop Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1988, pp. 177–182.

[20] A. Juels, D. Catalano, and M. Jakobsson, "Coercion-resistant electronic elections," in *Proc. ACM Workshop Privacy Electron. Soc.*, Nov. 2005, pp. 61–70.

[21] D. Chang, A. K. Chauhan, and J. Kang, "Apollo: End-to-end verifiable voting protocol using mixnet and hidden tweaks," in *Information Security and Cryptology—ICISC*. Cham, Switzerland: Springer, 2015, pp. 194–209.

[22] B. Adida, "Helios: Web-based open-audit voting," in *Proc. USENIX Secur. Symp.*, vol. 17, 2008, pp. 335–348.

[23] D. Chaum, "Blind signatures for untraceable payments," in *Advances in Cryptology*. Boston, MA, USA: Springer, 1983, pp. 199–203.

[24] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2001, pp. 552–565.

[25] J. D. C. Benaloh, *Verifiable Secret-Ballot Elections*. New Haven, CT, USA: Yale Univ. Press, 1987.

[26] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 1994, pp. 174–187.

[27] D. Zhang, M. Liu, and Z. Yang, "Zero-knowledge proofs of identity based on ElGamal on conic," in *Proc. IEEE Int. Conf. E-Commerce Technol. Dyn. E-Business*, Aug. 2004, pp. 216–223.

[28] S. Steffen, B. Bichsel, R. Baumgartner, and M. Vechev, "ZeeStar: Private smart contracts by homomorphic encryption and zero-knowledge proofs," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2022, pp. 179–197.

[29] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. Workshop Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 1985, pp. 47–53.

[30] F. Zhang and K. Kim, "ID-based blind signature and ring signature from pairings," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Berlin, Germany: Springer, 2002, pp. 533–547.

[31] J. C. Choon and J. Hee Cheon, "An identity-based signature from gap Diffie–Hellman groups," in *Proc. Int. Workshop Public Key Cryptography*. Berlin, Germany: Springer, 2002, pp. 18–30.

[32] K. Kurosawa and S.-H. Heng, "From digital signature to ID-based identification/signature," in *Proc. Int. Workshop Public Key Cryptography*. Berlin, Germany: Springer, 2004, pp. 248–261.

[33] A. K. Vangujar, T.-S. Ng, J. Chia, J. J. Chin, and S.-C. Yip, "Group identity-based identification: Definitions, construction and implementation," *Malaysian J. Math. Sci.*, vol. 15, pp. 123–139, Dec. 2021.

[34] S.-Y. Tan, S.-H. Heng, R. C.-W. Phan, and B.-M. Goi, "A variant of Schnorr identity-based identification scheme with tight reduction," in *Proc. Int. Conf. Future Gener. Inf. Technol.* Berlin, Germany: Springer, 2011, pp. 361–370.

[35] J. Katz and N. Wang, "Efficiency improvements for signature schemes with tight security reductions," in *Proc. 10th ACM Conf. Comput. Commun. Secur.*, 2003, pp. 155–164.

[36] I. Sertkaya, P. Roenne, and P. Y. A. Ryan, "Estonian internet voting with anonymous credentials," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 30, no. 2, pp. 420–435, Feb. 2022.

[37] V. Cortier, D. Galindo, S. Glondu, and M. Izabachène, "Distributed ElGamal á la pedersen: Application to helios," in *Proc. 12th ACM Workshop Workshop Privacy Electron. Soc.*, vol. 6879, Nov. 2013, pp. 131–142.

[38] T. Haines, R. Goré, and B. Sharma, "Did you mix me? Formally verifying verifiable mix nets in electronic voting," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2021, pp. 1748–1765.
[39] B. Yu, J. K. Liu, A. Sakzad, S. Nepal, R. Steinfeld, P. Rimba, and M. H. Au, "Platform-independent secure blockchain-based voting system," in *Proc. 21st Int. Conf.* Guildford, U.K.: Springer, Sep. 2018, pp. 369–386.
[40] J. I. A. Keromytis and M. Yung, *Applied Cryptography and Network Security*. Piscataway, NJ, USA: IEEE Press, 2019.
[41] S. H. Heng, "Design and analysis of some cryptographic primitives," Ph.D. dissertation, Tokyo Inst. Technol., Meguro, Japan, 2004.
[42] A. K. Vangujar, J. J. Chin, S. Y. Tan, and T.-S. Ng, "A hierarchical identity-based identification scheme without pairing," *Malaysian J. Math. Sci.*, vol. 13, pp. 93–109, 2019.
[43] C.-P. Schnorr, "Efficient identification and signatures for smart cards," in *Advances in Cryptology—CRYPTO*. New York, NY, USA: Springer, 1990, pp. 239–252.
[44] X. Yi, R. Paulet, and E. Bertino, *Homomorphic Encryption and Applications* (Springer Briefs in Computer Science). Springer, 2014, doi: 10.1007/978-3-319-12229-8.

APURVA K. VANGUJAR received the B.E. degree in computer science from MGM University, India, in 2015, and the master's degree by research in cryptography from Multimedia University, Cyberjaya, Malaysia, in 2019. She is currently pursuing the Ph.D. degree in cryptography from University College Cork, Ireland. She is employed with Jaguar Land Rover, Shannon, Ireland. Her research interests include identity-based cryptography, zero-knowledge proofs, and lattice-based cryptography. She is particularly keen on exploring ways to enhance the security of smart city applications, such as e-voting, e-auctions, and VANETs, through the application of cryptographic techniques.

BUVANA GANESH received the bachelor's and master's degrees in mathematics. Her master's thesis focused on post quantum cryptography and her Ph.D. thesis focused on homomorphic encryption and its applications with University College Cork. Currently, she is a Hardware Security Engineer with Qualcomm Inc.

ALIA UMRANI received the B.E. degree in telecommunications engineering from the NED University of Engineering and Technology (NEDUET), Karachi, Pakistan, in 2017, and the M.S. degree in information security from the National University of Sciences and Technology (NUST), Islamabad, Pakistan, in 2020. Currently, she is pursuing the Ph.D. degree in computer science with University College Cork (UCC), Cork, Ireland. She is also a Lecturer with the Department of Computing, Griffith College Cork, Cork. Her primary research interests include applied cryptography and security within smart broadcast environments. Specifically, she focuses on addressing security challenges in vehicular ad hoc networks by designing classical and post-quantum cryptographic protocols, including signature and key exchange protocols.

PAOLO PALMIERI (Member, IEEE) received the Ph.D. degree in cryptography from the Crypto Group, Université catholique de Louvain, Belgium, in January 2013. He then moved to the Delft University of Technology, The Netherlands, where he was a Postdoctoral Researcher for two years, working on privacy-enhancing technologies. Before joining UCC, he was a Lecturer in computing with Bournemouth University, from 2015 to 2016, and a Lecturer in cyber security with Cranfield University, U.K., from 2016 to 2017. He is currently a Lecturer in cyber security with University College Cork, Ireland. Most of his research work in cyber security focuses on cryptography, privacy, and anonymity. His research interests include secure computation, privacy-enhancing technologies, anonymity protocols, location privacy, and smart cities.