


## Research Article

# Improved Cloud-Assisted Privacy-Preserving Profile-Matching Scheme in Mobile Social Networks

Ying Zou,<sup>1,2</sup> Yanting Chai,<sup>3</sup> Sha Shi,<sup>4</sup> Lei Wang,<sup>1</sup> Yunfeng Peng,<sup>5</sup> Yuan Ping,<sup>6</sup> and Baocang Wang<sup>3</sup> 

<sup>1</sup>Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

<sup>2</sup>Department of Mathematics Teaching and Research, Shanghai Business School, Shanghai, China

<sup>3</sup>State Key Laboratory of Integrated Service Networks, Cryptographic Research Center, Xidian University, Xi'an 710071, China

<sup>4</sup>Engineering Research Center of Molecular and Neuro Imaging of Ministry of Education of China and School of Life Science and Technology, Xidian University, Xi'an 710071, China

<sup>5</sup>PBC School of Finance, Tsinghua University, Beijing, China

<sup>6</sup>School of Information Engineering, Xuchang University, Xuchang 461000, China

Correspondence should be addressed to Baocang Wang; [bcwang79@aliyun.com](mailto:bcwang79@aliyun.com)

Received 24 September 2019; Revised 3 May 2020; Accepted 1 August 2020; Published 21 September 2020

Academic Editor: Emanuele Maiorana

Copyright © 2020 Ying Zou et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Due to the transparency of the wireless channel, users in multiple-key environment are vulnerable to eavesdropping during the process of uploading personal data and re-encryption keys. Besides, there is additional burden of key management arising from multiple keys of users. In addition, profile matching using inner product between vectors cannot effectively filter out users with ulterior motives. To tackle the above challenges, we first improve a homomorphic re-encryption system (HRES) to support a single homomorphic multiplication and arbitrarily many homomorphic additions. The public key negotiated by the clouds is used to encrypt the users' data, thereby avoiding the issues of key leakage and key management, and the privacy of users' data is also protected. Furthermore, our scheme utilizes the homomorphic multiplication property of the improved HRES algorithm to compute the cosine result between the normalized vectors as the standard for measuring the users' proximity. Thus, we can effectively improve the social experience of users.

## 1. Introduction

With the rapid development of Internet technology, mobile devices such as mobile phones and tablets have gradually become popular in people's daily life in recent years. Some social networks such as WeChat, Twitter, and Facebook are gradually integrated into people's life, and people would like to share some opinions, pictures, and videos with others. Therefore, people are more willing to find potential friends with similar interests in mobile social networks.

Profile-matching is the most effective way to measure the proximity between users' personal profiles. The user's personal profile is often defined as a vector in practical applications, and each dimension of the vector represents an

attribute corresponding to a hobby, such as football, photography, and religion. Each attribute value is represented by an integer between 0 and 10 or a larger range. The attribute value indicates the degree of the interest. The value 0 means that the user has no interest in the item, and value 10 represents that the user is particularly fond of it. Moreover, social proximity is often defined as the inner product of two users' vectors [1–3]. Since the inner product is the sum of the products of the corresponding attributes between the two vectors, it cannot accurately show the proximity between users. For example, the vector of user  $A$  is  $\mathbf{u}_A = (3, 4, 7, 8)$ , user  $B$  is  $\mathbf{u}_B = (10, 10, 10, 10)$ , and user  $C$  is  $\mathbf{u}_C = (4, 3, 8, 7)$ . User  $A$  wants to find a person with a higher proximity between user  $B$  and user  $C$ . If the inner product of two

vectors is chosen as a measurement, it is surprising that user  $B$  meets higher requirements than user  $C$ . To get more accurate social proximity results, it is preferred to adopt cosine proximity result between the two normalized vectors as the standard.

In addition, users' profiles often contain sensitive data and they do not want to expose their personal information. One way to protect the users' privacy is to encrypt the data by cryptographic technologies, but the structure of raw data will be essentially damaged after encryption, causing difficulties in reprocessing the data. The homomorphic encryption technology [4–7] has its unique advantages in encrypted data processing. In particular, partial homomorphic encryption technique [8] is more suitable for many realistic applications [9].

In many existing profile-matching works [1, 10, 11], a large amount of interactions are often required between user mobile terminals to obtain matching results, which will bring heavy computational costs and communication overhead to users. In addition, users also need to stay online during the matching process. Fortunately, with the development of cloud computing technology [12, 13], the cloud platform can provide users with huge storage space and abundant computing resources. Outsourcing computing and storage to the cloud can effectively reduce the burden on the users' mobile terminals. Gao et al. [2] transfer users' work to the cloud with the help of two cooperative but non-collusive clouds, and users can go offline after uploading their profiles to the cloud. Gao et al.'s scheme utilizes an ElGamal-like proxy re-encryption [14] algorithm with additive homomorphic property, which leads to the issues of key management and the leakage of re-encryption keys. Furthermore, the ElGamal-like algorithm requires that the size of the plaintext cannot exceed 40 bits in order to ensure the decryption efficiency; thus it cannot be applied to the scenarios requiring high data accuracy.

The main contributions of our work are as follows:

- (i) We improve the HRES algorithm [15] in order to avoid the drawbacks in [2]. The improved algorithm supports one homomorphic multiplication and arbitrarily many homomorphic additions, which can effectively avoid the key leakage and the key management issues caused during users' uploading re-encryption keys.
- (ii) This paper utilizes the homomorphic multiplication property of the improved HRES algorithm to compute the cosine result between the normalized vectors as the standard for measuring proximity. Our proposal can effectively ensure the accuracy of the matching results and improve the social experience of the users. Furthermore, the improved HRES algorithm can prove to be semantically secure, and the profile-matching protocol is also secure in the sense that both clouds cannot get useful information about users' data under the non-collusion security model.

**1.1. Related Work.** The research on privacy-preserving profile matching can be mainly divided into two categories, coarse-grained [16–19] and fine-grained [1, 2, 10] profile-matching schemes.

**1.1.1. Coarse-Grained Scheme.** In the coarse-grained profile-matching schemes, the matching proximity is often defined as a set intersection or the cardinality of intersections of user's attribute sets, but this solution cannot further distinguish the specific relevance between users. In 2011, Li et al. [16] proposed two distributed privacy-preserving profile-matching protocols, which deploy the homomorphic property of Shamir secret sharing scheme [20] to calculate the intersections of users' private sets without relying on a trusted third party. However, the coarse-grained profile matching cannot accurately measure the proximity of users.

**1.1.2. Fine-Grained Scheme.** For the fine-grained profile-matching schemes [1, 2, 10], user's preference or behavior pattern is usually regarded as a multidimensional vector, and the social proximity is usually measured by the inner product between two users' vectors. For example, Zhang et al. [1] designed a fine-grained profile-matching protocol with three security levels. A user can initiate a matching query for their encrypted data with other users and finally obtains the proximity result by utilizing the homomorphic addition property of the Paillier cryptosystem [21]. However, it is required that both users stay online to perform multiple interactions during the execution of protocols, thus imposing a heavy communication and computational burden on mobile terminals. To tackle the above issues, Gao et al. [2] introduced a novel cloud-assisted profile-matching scheme under multiple keys. The cloud environment is composed of two cooperative but non-collusive clouds, and users in the social application could go offline after uploading their encrypted data. A friend finder can designate a target and initiate a matching query to the cloud; then, the two cloud servers return the matching result to the user through interactions. In their scheme, the clouds perform most of the computations, which effectively reduces the burden of users. And the data providers do not have to stay online all the time. However, the scheme [2] utilizes a secure ElGamal-like [14] proxy re-encryption algorithm to encrypt data, so each user needs to generate their secret keys and re-encryption keys. Although schemes in a multiple-key environment could benefit from some existing technologies [22] that could create secure communicating groups within a secure group with exchange of small information, multiple keys increase additional burden of key management in our consideration. Moreover, it seems impossible to guarantee users' privacy once the user's re-encryption key is leaked in the process of uploading to the cloud. What is worse, Gao et al.'s scheme uses the inner product of two vectors to measure the matching degree. However, the ElGamal-like algorithm requires the size of the plaintext to be less than 40 bits to make sure that the ciphertext can be efficiently decrypted using Pollard's kangaroo algorithm [23] to solve the discrete logarithm problem for a relatively smaller

integer. Therefore, the scheme fails to provide high precision computations on the users' data.

**1.2. Organization.** The rest of the paper is organized as follows. Section 2 gives some notations and the main part of the HRES algorithm [15]. Section 3 introduces the system and adversary models. Section 4 provides the construction of the improved algorithm and the profile-matching scheme. Section 5 presents the security proof of the improved HRES algorithm and the security analysis of the profile-matching protocol. Section 6 draws conclusions.

## 2. Preliminaries

In this section, we briefly introduce some notations appearing in this paper. We also introduce a fundamental HRES algorithm with two cooperative and non-collusive clouds and a data normalization method.

**2.1. Notation.** In this paper, we write  $x \leftarrow X$  for assigning  $x$  a value  $X$  and denote the integer set  $\{1, \dots, n\}$  as  $[n]$ . In addition,  $a \rightarrow b$  indicates that  $b$  is calculated by the algorithm  $a$ . We use  $(\cdot)$  and  $(\cdot)$  to denote the Euler and the Carmichael function, respectively. The ring  $\mathbb{Z}_n$  means the residue class ring of integers modulo  $n$ , i.e.,  $\{0, \dots, n-1\}$ . We use bold lowercase letters to represent vectors, and the Euclidean norm for a vector  $\mathbf{u}$  is denoted as  $|\mathbf{u}|$ . In addition, the bit length of an integer  $a$  is denoted as  $|a|$ . We use the symbol  $\lceil \cdot \rceil$  to denote the ceiling function. Besides,  $\mathbf{u} \cdot \mathbf{v}$  means the inner product of the two vectors  $\mathbf{u}$  and  $\mathbf{v}$ , and we use  $E_{PK}(\cdot)$  to denote the encryption function with the public key  $PK$  of the improved HRES algorithm that will be introduced in the next section.

**2.2. Homomorphic Re-Encryption.** Re-encryption technology supports a proxy transferring decryption authority without decrypting ciphertext [24]. On this basis, homomorphic re-encryption supports homomorphic operation on ciphertext, which is more suitable for data dissemination in networks with special needs. In 2017, Ding et al. proposed a HRES homomorphic re-encryption scheme [15], which includes two non-collusive cloud servers, CA and CB, that jointly manage the ciphertext data. The data owner encrypts his data with the public key generated by negotiation between the two cloud servers, and the ciphertexts can be correctly decrypted only if the two cloud servers cooperate with each other. The HRES consists of the following algorithms.

- (i) **Key Generation.** Given a security parameter  $\lambda$ , let  $n = pq$  be a safe RSA modulus, where  $p$  and  $q$  are primes of the form of  $p = 2p' + 1$  and  $q = 2q' + 1$ , and  $p'$  and  $q'$  are primes of equal bit length. Let  $g$  be an element of the maximal order  $(n^2) = \text{lcm}(\phi(p^2), \phi(q^2)) = 2np'q'$  in  $\mathbb{Z}_{n^2}$ . The two cloud servers CA and CB, respectively, generate their own key pairs:  $(sk_{CA} = a \in \mathbb{Z}_{(n^2)}, pk_{CA} = g^a \pmod{n^2})$  and  $(sk_{CB} = b \in \mathbb{Z}_{(n^2)}, pk_{CB} = g^b \pmod{n^2})$ . Therefore,

CA negotiates with CB to generate their Diffie-Hellman key  $PK = pk_{CA}^{sk_{CB}} = pk_{CB}^{sk_{CA}} = g^{ab} \pmod{n^2}$ .

- (ii) **Encryption.** Given the Diffie-Hellman key  $PK$  and a message  $m \in \mathbb{Z}_n$ , output the ciphertext as follows:  $\mathbf{c} = E_{PK}(m) = (c_1, c_2) = ((1 + mn)PK^r \pmod{n^2}, g^r \pmod{n^2})$ , where  $E_{PK}(m)$  indicates the ciphertext encrypted with  $PK$  and  $r$  is a random integer selected from  $\mathbb{Z}_{(n^2)}$ .
- (iii) **Transfer.** Given  $sk_{CA} = a$  and a ciphertext  $\mathbf{c} = E_{PK}(m) \in \mathbb{Z}_{n^2}^2$ , the cloud CA can transfer the above ciphertext into another ciphertext as  $E_{pk_{CB}}(m) = (c_1', c_2') = (c_1, c_2^{a \pmod{n^2}})$  where  $E_{pk_{CB}}(m)$  indicates that the ciphertext can be decrypted with  $sk_{CB}$ .
- (iv) **Decryption.** Given  $sk_{CB} = b$  and a ciphertext  $E_{pk_{CB}}(m)$ , the cloud CB can decrypt the plaintext as follows:

$$\begin{aligned} c_2' &= c_2^b = g^{rab} = PK^r \pmod{n^2}, \\ m &= L_n\left(\left(-\right)\left(-\right)\right) \pmod{n^2} = L_n(1 + mn), \end{aligned} \quad (1)$$

where the function  $L_n(\cdot)$  is defined as  $L_n(x) = x - 1/n$ .

**2.3. Normalization Method.** This section mainly introduces a common data standardization method: Z-score standardization method, which standardizes data based on the mean and variance of the original data. The processed data obeys the normal distribution; that is, the mean value of the data is 0 and the standard deviation is 1. The conversion formula is as follows:

$$x^* = \frac{x - \bar{x}}{\sigma}, \quad (2)$$

where  $x^*$  represents the processed data,  $\bar{x}$  is the mean of the samples and  $\sigma$  denotes the standard deviation of the samples.

In the proposed scheme, the raw data need to be normalized before processing. The reason for this operation is that the values of data from different sources are quite different. In order to eliminate the influence of different numerical range and make the data comparable, data need to be normalized. Hence, Z-score method is adopted to make the data at the same level, which is convenient for analyzing the data.

## 3. System and Threat Model

**3.1. System Model.** As shown in Figure 1, our system model contains three entities: a friend finder (Alice), the cloud environment, and other users. Each entity is described as follows: Alice is marked as a friend finder who wants to find friends that have similar interests with her in the social network. The cloud environment includes two cloud servers, CA and CB, which can provide users with enormous storage space for storing personal profiles and a large amount of

computing resources. Through cooperating, CA and CB can help Alice calculate the proximities with other users, and the privacy of the two users will be not compromised. There are many other users in social networks who prefer to outsource the encrypted data representing their preferences to the cloud CA.

Each user registers a personal account in the mobile social network and then fills in the personal information. The personal information contains user's preferences that can be used as a measurement for profile matching.

(iv) **Partial decryption.** Given  $sk_{CB}$  and a ciphertext  $E_{pk_{CB}}(m) \in \mathbb{Z}_{p^3}^2$ , the cloud CB can output the plaintext as follows:

$$\begin{aligned} &= \frac{b}{2} = ab = g^{rab} = PK^r \pmod{p^3}, \\ m &= L_p \left( \left( \frac{2}{ab} \right) \pmod{p^3} \right) = L_p(1 + mp), \end{aligned} \quad (3)$$

where the function  $L_p(\cdot)$  is defined as  $L_p(x) = x - 1/p$ .

(v) **Aggregation.** We first show that the improved scheme supports  $k + 1$  homomorphic additions. Given any  $k + 1$  ciphertexts, namely, for  $i = 1, \dots, k + 1$ ,  $\mathbf{c}_i = E_{PK}(m_i) = (g^{r_i}, (1 + m_i p) PK^{r_i} \pmod{p^3})$  with the underlying plaintext being  $m_i$  for  $i = 1, \dots, k + 1$ , we firstly show that our improved algorithm supports  $k + 1$  homomorphic additions.

Specifically, we compute  $\mathbf{c} = (c_1, \dots, c_{k+1}) = (\prod_{i=1}^{k+1} g^{r_i} \pmod{p^3}, \prod_{i=1}^{k+1} (1 + m_i p) PK^{r_i} \pmod{p^3})$ . Note that

$$\begin{aligned} &= \prod_{i=1}^{k+1} g^{r_i} = \prod_{i=1}^{k+1} (1 + m_i p) PK^{r_i} = \left( 1 + p \sum_{i=1}^{k+1} m_i + p^2 \sum_{i \neq j} m_i m_j \right) g^{ab \% \sum_{i=1}^{k+1} r_i} \pmod{p^3}, \\ &= \prod_{i=1}^{k+1} g^{r_i} = \prod_{i=1}^{k+1} g^{r_i} = g^{ab \% \sum_{i=1}^{k+1} r_i} \pmod{p^3}. \end{aligned} \quad (4)$$

From the refreshed ciphertext  $\mathbf{c} = (c_1, \dots, c_{k+1})$ , the cloud CA can use his secret key  $sk_{CA} = a$  to partially decrypt the ciphertext as  $\mathbf{c}_2 = (c_2, c_2) = (c_1, c_1^a \pmod{p^3})$ , where

$$\begin{aligned} c_2 &= \left( 1 + p \sum_{i=1}^{k+1} m_i + p^2 \sum_{i \neq j} m_i m_j \right) g^{ab \% \sum_{i=1}^{k+1} r_i} \pmod{p^3}, \\ c_2 &= c_1^a = g^{a \% \sum_{i=1}^{k+1} r_i} \pmod{p^3}. \end{aligned} \quad (5)$$

The cloud CB can further decrypt the ciphertext  $\mathbf{c}_2 = (c_2, c_2)$  using his secret key  $sk_{CB} = b$  as follows. The cloud CB first computes

$$= \frac{b}{2}$$

and  $c_2 = (c_2^{(1)}, c_2^{(2)}) = (g^{r_1+r_2} \pmod{p^3}, (1 + p(m_1 + m_2) + p^2 m_1 m_2) g^{a(r_1+r_2)} \pmod{p^3})$ , where  $c_2^{(1)} = g^{r_1+r_2} \pmod{p^3}$  and  $c_2^{(2)} = (1 + p(m_1 + m_2) + p^2 m_1 m_2) g^{a(r_1+r_2)} \pmod{p^3}$ . The cloud CB first computes

$$c_2^{(2)} = g^{ab(r_1+r_2)} = PK^{r_1+r_2} \pmod{p^3}. \quad (10)$$

Noting that

$$c_2^{(2)} = 1 + p(m_1 + m_2) + p^2 m_1 m_2 \pmod{p^3}. \quad (11)$$

The cloud CB computes

$$\begin{aligned} & L_p \left( c_2^{(2)} \pmod{p^3} \right) \pmod{p} \\ &= L_p \left( 1 + p(m_1 + m_2) + p^2 m_1 m_2 \pmod{p^3} \right) \pmod{p} \\ &= L_p \left( 1 + p(m_1 + m_2) + p^2 m_1 m_2 + tp^3 \pmod{p^3} \right) \pmod{p} \\ &= ((m_1 + m_2) + pm_1 m_2 + tp^2) \pmod{p} \\ &= (m_1 + m_2) \pmod{p}. \end{aligned} \quad (12)$$

CB could obtain  $m_1 + m_2$  when  $|m| + 1 < |p|$ . Then, CB calculates

$$\begin{aligned} & \frac{c_2^{(2)} \pmod{p^3} - 1 - p(m_1 + m_2)}{p^2} \pmod{p} \\ &= \frac{1 + p(m_1 + m_2) + p^2 m_1 m_2 + tp^3 - 1 - p(m_1 + m_2)}{p^2} \pmod{p} \\ &= \frac{p^2 m_1 m_2 + tp^3}{p^2} \pmod{p} \\ &= m_1 m_2 + tp \pmod{p} \\ &= m_1 m_2 \pmod{p}, \end{aligned} \quad (13)$$

where the cloud CB obtains the results  $m_1 m_2 \pmod{p}$ . Thus, our scheme supports a homomorphic multiplication operation on ciphertexts when  $|m| < |p|/2$ .

**4.2. Privacy-Preserving Profile Matching.** In this part, the improved HRES algorithm is adopted to implement our privacy-preserving profile-matching scheme. Suppose that Alice's vector is  $\mathbf{u} = (u_1, \dots, u_n)$  and that Bob's vector is  $\mathbf{v} = (v_1, \dots, v_n)$ . The cosine value of the two vectors can be calculated as

$$\begin{aligned} \mathbf{u} \cdot \mathbf{v} &= u_1 v_1 + \dots + u_n v_n = \sum_{i=1}^n u_i v_i, \\ \cos(\mathbf{u}, \mathbf{v}) &= \frac{\sum_{i=1}^n u_i^2 + \sum_{i=1}^n v_i^2 - \sum_{i=1}^n (u_i - v_i)^2}{2|\mathbf{u}||\mathbf{v}|}. \end{aligned} \quad (14)$$

In particular,  $\cos(\mathbf{u}, \mathbf{v}) \in [0, 1]$  since  $u_i$  and  $v_i$  are nonnegative integers. Users can encrypt their vectors with the Diffie-Hellman key  $PK$  and then outsource their encrypted data to CA. The procedure for the data outsourcing is presented in Algorithm 1 and the privacy-preserving profile-matching scheme is shown as Algorithm 2.

## 5. Correctness and Security

In this section, we firstly give the correctness analysis of our protocol and then prove that the improved HRES algorithm is semantically secure with rigorous method. At last, we prove that our profile-matching scheme is secure under the semihonest model.

**5.1. Correctness.** In our scheme, CB can correctly decrypt and obtain the result of  $\sum_{i=1}^n (u_i' - v_i')$  without revealing any useful information. The demonstration is shown as follows. Firstly, CA computes

$$\begin{aligned} E_{PK_2}^{p-1}(v_i') &\equiv \left\{ (1 + v_i'p)^{p-1} PK^{(p-1)r_i}, g^{(p-1)r_i} \right\} \pmod{p^3} \\ &\equiv \left\{ (1 + v_i'p(p-1) + C_{p-1}^2(v_i'p)^2) PK^{(p-1)r_i}, g^{(p-1)r_i} \right\} \pmod{p^3}. \end{aligned} \quad (15)$$

We can verify that

$$\begin{aligned} & E_{PK_2}(u_i') \cdot E_{PK_2}^{p-1}(v_i') \\ &\equiv \left\{ (1 + u_i'p) PK^{r_i}, g^{r_i} \right\} \cdot \left\{ (1 + v_i'p(p-1) + C_{p-1}^2(v_i'p)^2) PK^{(p-1)r_i}, g^{(p-1)r_i} \right\} \pmod{p^3} \\ &\equiv \left\{ (1 + (u_i' - v_i')p + (v_i' - u_i'v_i' + C_{p-1}^2(v_i')^2)p^2) PK^{(p-1)r_i+r_i'}, g^{(p-1)r_i+r_i'} \right\} \pmod{p^3}. \end{aligned} \quad (16)$$



**Input:** A user Bob wants to outsource his personal profile to CA and holds a private vector  $\mathbf{u} = \langle u_1, \dots, u_n \rangle$ .

**Output:** The encrypted result  $E_{PK}(\mathbf{u}')$ ,  $E_{PK}(\frac{1}{B})$  and  $E_{PK}(s_B)$  are sent to CA.

- (1) CA executes the  $\mathcal{E}$  algorithm of the improved HRES algorithm with CB to generate their respective key pairs  $\langle pk_{CA}, sk_{CA} \rangle, \langle pk_{CB}, sk_{CB} \rangle$  and their Diffie-Hellman key  $PK_2$ . Thereafter,  $PK_2$  is issued to the users in social networks.
- (2) Firstly, Bob normalizes his vector with Z-score method and then converts each element of the normalized vector to an integer by multiplying it with a pretreatment public integer  $l_1$  and ceiling. Finally, Bob gets his matching vector  $\mathbf{u}' = \langle u'_1, u'_2, \dots, u'_n \rangle$  and calculates  $s_B = \sum_{i=1}^n u'^2_i$ ,  $B = \lceil |\mathbf{u}'| \rceil$ .
- (3) Bob negotiates with the CA to run the Diffie-Hellman key exchange protocol to generate a secret random integer  $r_1$ .
- (4) After encrypting the following values:  $E_{PK_2}(\mathbf{u}') \leftarrow \langle E_{PK_2}(u'_1), E_{PK_2}(u'_2), \dots, E_{PK_2}(u'_n) \rangle$ ,  $E_{PK_2}(s_B)$ , and  $E_{PK_2}(\frac{1}{B})$ , Bob uploads these encrypted values to CA.

#### ALGORITHM 1: Data outsourcing.

**Input:** Alice's private vector  $\mathbf{v} = \langle v_1, \dots, v_n \rangle$ , and the secret keys of CA and CB.

**Output:** Alice gets the cosine result  $\cos(\mathbf{u}', \mathbf{v}')$ .

- (1) Alice executes the  $\mathcal{E}$  algorithm of the improved HRES algorithm with CA to generate their respective key pairs  $\langle pk_A, sk_A \rangle$ ,  $\langle pk'_{CA}, sk'_{CA} \rangle$  and their Diffie-Hellman key  $PK_1$ . Then,  $PK_1$  is assigned to the cloud CB.
- (2) Alice sends her encrypted profile to CA for querying the proximity with Bob in the social networks.
  - (i) Alice also needs to process her normalized vector with the public number  $l_1$  to get the result  $\mathbf{v}' = \langle v'_1, \dots, v'_n \rangle$ , and calculates  $s_A = \sum_{i=1}^n v'^2_i$ ,  $A = \lceil |\mathbf{v}'| \rceil$ .
  - (ii) Alice negotiates with CA to run the Diffie-Hellman key exchange protocol to generate a secret random integer  $r_2$ .
  - (iii) Alice uploads the encrypted values  $E_{PK_2}(\mathbf{v}')$ ,  $E_{PK_2}(\frac{1}{A})$ , and  $E_{PK_2}(s_A)$  to CA.
- (3) CA generates random integers  $r_i$  and works out the following formulas:
  - (i)  $E_{PK_2}(u'_i) \cdot E_{PK_2}^{p-1}(v'_i) \rightarrow E_{PK_2}(u'_i - v'_i)$ .
  - (ii)  $E_{PK_2}(u'_i - v'_i) \rightarrow E_{PK_2}(r_i(u'_i - v'_i)_i)$ .
 Next, CA computes  $E_{PK_2}(\frac{1}{B}) \cdot E_{PK_2}(\frac{1}{A})$ ; executes the  $\mathcal{D}$  algorithm to process  $E_{PK_2}(r_i(u'_i - v'_i)_i)$  and  $E_{PK_2}(\frac{1}{B}) \cdot E_{PK_2}(\frac{1}{A})$  with its secret key  $sk_{CA}$ ; and then sends the processed results to CB.
- (4) CB runs the  $\mathcal{D}$  algorithm with its secret key  $sk_{CB}$  to decrypt  $E_{PK_2}(\frac{1}{B}) \cdot E_{PK_2}(\frac{1}{A})$  and gets  $\frac{1}{AB}$  where  $\frac{1}{AB} = \frac{1}{A} \cdot \frac{1}{B}$ .
- (5) CB runs the  $\mathcal{D}$  algorithm with its secret key  $sk_{CB}$  to decrypt  $E_{PK_2}(r_i(u'_i - v'_i)_i)$  and compute  $r_i(u'_i - v'_i)_i \cdot \frac{1}{AB}$ . Next, CB sends the encrypted values  $E_{PK_2}(r_i(u'_i - v'_i)_i)$  to CA.
- (6) CA computes
  - (i)  $E_{PK_2}(\frac{1}{AB} \cdot (u'_i - v'_i)^2) \rightarrow E_{PK_2}((u'_i - v'_i)^2)$ .
  - (ii)  $E_{PK_2}(s_A) \cdot E_{PK_2}(s_B) \cdot E_{PK_2}^{p-1}(\sum_{i=1}^n (u'_i - v'_i)^2) \rightarrow E_{PK_2}(2\mathbf{u}' \circ \mathbf{v}')$ .
  - (iii)  $E_{PK_2}^{2-1}(2\mathbf{u}' \circ \mathbf{v}') \rightarrow E_{PK_2}(\mathbf{u}' \circ \mathbf{v}')$ .
 After that, CA generates a random integer  $r_3$  and calculates  $E_{PK_2}(\mathbf{u}' \circ \mathbf{v}')$ . Finally, CA runs the  $\mathcal{D}$  algorithm to decrypt  $E_{PK_2}(\mathbf{u}' \circ \mathbf{v}')$  and sends the results to CB.
- (7) CB computes the obfuscated proximity result and sends it to CA.
  - (i) CB runs the  $\mathcal{D}$  algorithm to decrypt  $E_{PK_2}(\mathbf{u}' \circ \mathbf{v}')$  and obtains  $\mathbf{u}' \circ \mathbf{v}'$ .
  - (ii) CB processes the value  $\frac{1}{AB}$ .

Then, CA generates the random integers  $r_i$  to obfuscate  $E_{PK_2}(u'_i - v'_i)$ , and the final result can be obtained through the following formula:

$$\begin{aligned} & E_{PK_2}^i(u'_i - v'_i) \\ & \equiv \left\{ \left( 1 + (u'_i - v'_i)p + (v'_i - u'_i v'_i + C_{p-1}^2(v'_i)^2)p^2 \right)^{i PK} g^{i r_i (p-1) + i r'_i}, g^{i r_i (p-1) + i r'_i} \right\} \bmod p^3 \\ & \equiv \left\{ \left( 1 + i(u'_i - v'_i)p + X \right) PK^{i r_i (p-1) + i r'_i}, g^{i r_i (p-1) + i r'_i} \right\} \bmod p^3, \end{aligned} \quad (17)$$

where  $X = (C_{p-1}^2(u'_i - v'_i) + i(v'_i - u'_i v'_i + C_{p-1}^2(v'_i)^2))p^2$ . Hence, CB can obtain  $i(u'_i - v'_i) \bmod p$ . Then, CB calculates  $\frac{2}{i}(u'_i - v'_i)^2 \bmod p = \frac{2}{i}(u'_i - v'_i)^2$  when  $2|i| + 2|\max\{u'_i, v'_i\}| < p$ , and it cannot reveal information about  $u'_i$  and  $v'_i$ .

On the other hand, we will prove that Alice can obtain a more accurate cosine result as follows.

Once CA receives  $E_{PK_2}(\frac{2}{i}(u'_i - v'_i)^2)$ , it can remove the blinding values  $\frac{2}{i}$  by computing  $E_{PK_2}^i(\frac{2}{i}(u'_i - v'_i)^2) \longrightarrow E_{PK_2}((u'_i - v'_i)^2)$ , and then it computes the following formulas:

- (i)  $E_{PK_2}((u'_1 - v'_1)^2), \dots, E_{PK_2}((u'_n - v'_n)^2) \longrightarrow E_{PK_2}(\sum_{i=1}^n (u'_i - v'_i)^2)$ .
- (ii)  $E_{PK_2}(s_A) \cdot E_{PK_2}(s_B) \cdot E_{PK_2}^{p-1}(\sum_{i=1}^n (u'_i - v'_i)^2) \longrightarrow E_{PK_2}(2\mathbf{u}' \circ \mathbf{v}')$ .
- (iii)  $E_{PK_2}^{2^{-1}}(2\mathbf{u}' \circ \mathbf{v}') \longrightarrow E_{PK_2}(\mathbf{u}' \circ \mathbf{v}')$ .

Then, CA obtains  $E_{PK_2}(\mathbf{u}' \circ \mathbf{v}')$  with the blinding value  $\cdot$ . CB decrypts the ciphertext, computes  $l_2(\mathbf{u}' \circ \mathbf{v}') / 1 \leq 2 \leq A \leq B$ , and then encrypts the ceiling result with  $PK_1$ . Thus, CA can compute and get  $E_{PK_1}^{1/2^{-1}}(\frac{1}{2} \cdot \mathbf{u}' \circ \mathbf{v}') \longrightarrow E_{PK_1}(\frac{1}{2} \cdot \mathbf{u}' \circ \mathbf{v}')$ . We find that when the pretreatment number  $l_2$  is large,  $(\frac{1}{2} \cdot \mathbf{u}' \circ \mathbf{v}')$  is closer to the cosine result  $\cos(\mathbf{u}', \mathbf{v}')$ .

**5.2. Security Proof of Improved Algorithm.** In this subsection, the semantic security of the improved HRES algorithm is proved through three theorems. We first prove that the DDH problem in  $G$  (the cyclic group of modulo  $p^3$ ) is hard to solve. Based on this conclusion, the improved HRES algorithm could be semantically secure.

**Theorem 1.** *Let  $G$  be the cyclic group of modulo  $p^3$ , and  $g$  be a generator of  $G$ . The discrete logarithm problem (in  $G$ ) is hard to solve.*

*Proof.* For the sake of contradiction, it is assumed that the discrete logarithm problem (in  $G$ ) is not difficult; i.e., there exists an oracle that can solve the discrete logarithm problem (in  $G$ ) in polynomial time. For example, given an input  $A \equiv g^a \bmod p^3$ , the oracle returns the index  $a$  as output.

- (1) Note that, if  $g$  is a generator of  $\mathbb{Z}_p$  and satisfies  $g^{p-1} \neq 1 \bmod p^2$ ,  $g$  is also a generator of  $G$ . Our strategy is as follows. Let  $B \equiv g^b \bmod p$ , we can

assume that there exists an integer  $b'$  that satisfies  $B \equiv g^{b'} \bmod p$ , such that the equation  $b \equiv b' \bmod (p)$  holds. Multiply simultaneously both sides of the equation by  $p^2$  and the equation  $bp^2 \equiv b$



exists an adversary  $A_{DDH}$  who can distinguish the random quadruple  $R$  from DH quadruple  $D$ ,  $A_{DDH}$  can also distinguish  $R$  and  $D$  in  $\mathbb{Z}_p$ . However, the DDH assumption in  $\mathbb{Z}_p$  is difficult, so the original hypothesis does not hold. Thus, we can get a conclusion that DDH assumption (in  $G$ ) is also difficult.

- (2) If  $g$  is a generator of  $\mathbb{Z}_p$  and satisfies  $g^{p-1} \equiv 1 \pmod{p^2}$ , then  $g + p$  is also the generator that belongs to both  $\mathbb{Z}_p$  and  $G$ . The proof process is similar to the above one and we will not describe it here.

**Theorem 3.** *If Decisional Diffie-Hellman assumption in  $\mathbb{Z}_{p^3}^*$  holds, the improved HRES algorithm  $\Pi$  presented in Section 5.1 is semantically secure.*

*Proof.* During the KeyGen phase, the cloud servers CA and CB negotiate with each other to generate their Diffie-Hellman key. Due to the difficulty of discrete logarithm problem in  $\mathbb{Z}_{p^3}^*$ , it is negligible to get any information about  $sk_1$ ,  $sk_2$ , or  $sk_1 \cdot sk_2$  for any adversaries.

For the sake of contradiction, it is assumed that the scheme  $\Pi$  is not semantically secure; i.e., there exists a polynomial time adversary  $A_\Pi$  which can break semantic security with nonnegligible probability.  $A_\Pi$  constructs a distinguisher that can solve the DDH problem in  $\mathbb{Z}_{p^3}^*$ . The construction is as follows.

Given a challenge quadruple  $(g, g^a, g^b, T)$ , where  $a, b \in \mathbb{Z}_{p^3}^*$ , the goal of the distinguisher is to determine  $T = g^{ab}$  or  $T = R$ , where  $R$  is a random integer in  $\mathbb{Z}_{p^3}^*$ . The challenger sends the public key  $PK = g^a$  to the adversary  $A_\Pi$ . Then, the adversary  $A_\Pi$  submits two challenge messages  $(m_0, m_1)$  of equal length to the challenger based on his prior knowledge, and then the challenger returns  $C = (g^b, T(1 + m_d p)) \pmod{p^3}$  as a challenge ciphertext to the adversary  $A_\Pi$ , in which  $d \in \{0, 1\}$ . Finally, the adversary outputs  $d'$  as the guess result. If  $d = d'$ , the challenger outputs  $T = g^{ab}$ ; otherwise, it outputs  $T = R$ . The discussion is as follows:

- (1) If  $T = g^{ab}$ , then  $C$  is a valid ciphertext and the probability of adversary guessing correctly is equal to  $1/2 + \epsilon$ .
- (2) If  $T = R$ , then  $R(1 + mp)$  is independent of the encrypted message because the random value  $R$  is uniformly and randomly distributed among  $\mathbb{Z}_{p^3}^*$ . Therefore, the probability that the adversary guesses correctly is  $1/2$ .

As a result, if the distinguisher can break the scheme  $\Pi$  with a nonnegligible probability, the adversary  $A_\Pi$  can attack the DDH assumption in  $\mathbb{Z}_{p^3}^*$  with the same advantage. For the reason that the DDH assumption in  $\mathbb{Z}_{p^3}^*$  is difficult, our improved scheme  $\Pi$  is semantically secure.

**5.3. Security Analysis of Our Protocol.** The security analysis of our privacy-preserving profile-matching scheme under the semihonest model will be presented in this subsection

with a real and ideal paradigm [2, 26]. For any adversaries who attack a real protocol execution, there exists an adversary who attacks an ideal execution, such that the input and output distributions of the adversary and participants in both the real and the ideal executions are fundamentally the same.

**Theorem 4.** *Our profile-matching scheme described in Section 5 can securely obtain the matching result through the calculations on ciphertexts under the semihonest and non-collusive adversaries.*

*Proof.* In this scheme, there are mainly four parties: Alice, Bob, CA, and CB. We can construct four simulators  $\text{Sim} = \langle \text{Sim}_A, \text{Sim}_B, \text{Sim}_{CA}, \text{Sim}_{CB} \rangle$  against four types of adversaries  $\langle A_A, A_B, A_{CA}, A_{CB} \rangle$  that will corrupt the privacy of Alice, Bob, CA, and CB, respectively.

$\text{Sim}_A$  simulates  $A_A$  as follows: After receiving the normalized vector  $\mathbf{v}' = \langle v'_1, v'_2, \dots, v'_n \rangle$ ,  $\text{Sim}_A$  encrypts  $\mathbf{v}'$  and  $\sum_{i=1}^n v_i^2$ , respectively, to get  $E_{PK_2}(\mathbf{v}')$  and  $E_{PK_2}(s_A)$ . Then,  $\text{Sim}_A$  chooses a random integer  $r_2$  and encrypts  $r_2 \left\lceil \sqrt{\sum_{i=1}^n v_i^2} \right\rceil$  into  $E_{PK_2}(r_2 \cdot s_A)$ .  $\text{Sim}_A$  randomly picks a vector  $\mathbf{U}' = \langle \hat{u}_1, \hat{u}_2, \dots, \hat{u}_n \rangle$ , computes  $E_{PK_2}(r_1 r_2 \cdot \mathbf{U}' \circ \mathbf{v}')$ , and sends the partial decryption result  $E_{PK_2}(r_1 r_2 \cdot \mathbf{U}' \circ \mathbf{v}')$  to  $A_B$ .

$\text{Sim}_{\text{CB}}$  simulates  $\text{A}_{\text{CB}}$  as follows:  $\text{Sim}_{\text{CB}}$  chooses random integers  $r_i, m_i, t_i$ , where  $i \in [n]$ , and then encrypts them as  $(E_{PK_2}(m_i) \cdot E_{PK_2}^{-1}(t_i))^{r_i}, E_{PK_2}((r_i m_i)^2)$ . Then, it picks a random number  $s$  and encrypts it as  $E_{PK_1}(s), E_{PK_2}(s)$ .  $\text{Sim}_{\text{CB}}$  re-encrypts them with the secret key of CA and sends the values  $(E_{PK_2}(m_i) \cdot E_{PK_2}^{-1}(t_i))^{r_i}, E_{PK_2}((r_i m_i)^2), E_{PK_1}(s), E_{PK_2}(s), r_i(m_i - t_i)$ , and  $(r_i m_i)^2$  to  $\text{A}_{\text{CB}}$ . The view of  $\text{A}_{\text{CB}}$  is the above encrypted values and some obfuscated data. Although  $\text{A}_{\text{CB}}$  can decrypt and obtain the obfuscated data, the random numbers selected by the simulator are uniformly and randomly distributed in the message space, so they are the obfuscated messages. The view of  $\text{A}_{\text{CB}}$  in real and ideal executions is indistinguishable owing to the semantic security of the improved HRES scheme mentioned above.

ultimately get the matching result. Most computations are

## 6. Evaluation

**6.1. Comparison.** In this subsection, we mainly discuss the advantages of our scheme compared with the existing privacy-preserving profile-matching schemes [1, 2, 11, 27] in Table 1. These schemes [1, 11, 27] require users to stay online simultaneously to obtain matching results through multiple interactions, resulting in additional computational costs and communication overheads on mobile devices of users. In our scheme, users only need to encrypt their personal profiles and upload them to the cloud, and then they can go offline. For a Friend finder, he can designate a target to initiate a matching query and

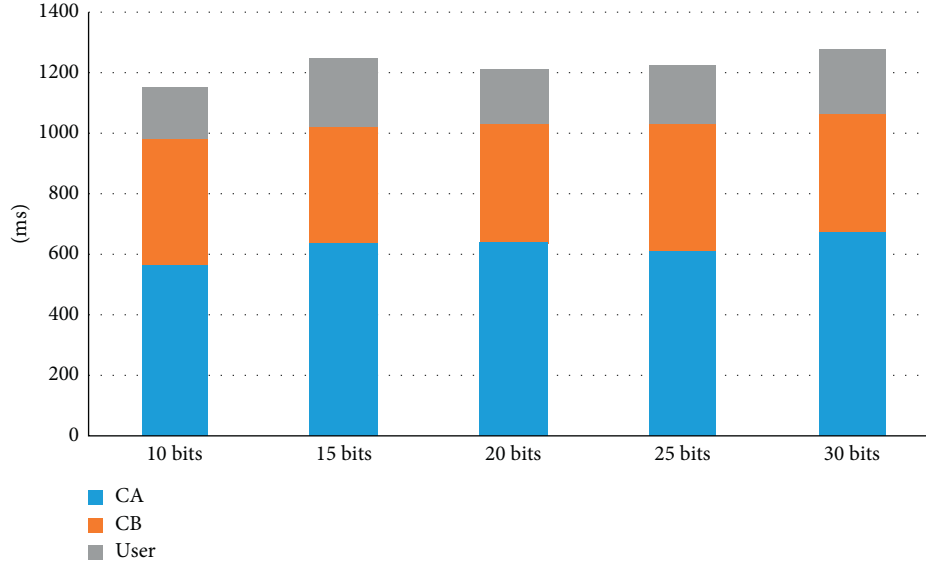


FIGURE 3: Computational costs of the proposed scheme with the increase in data bits.

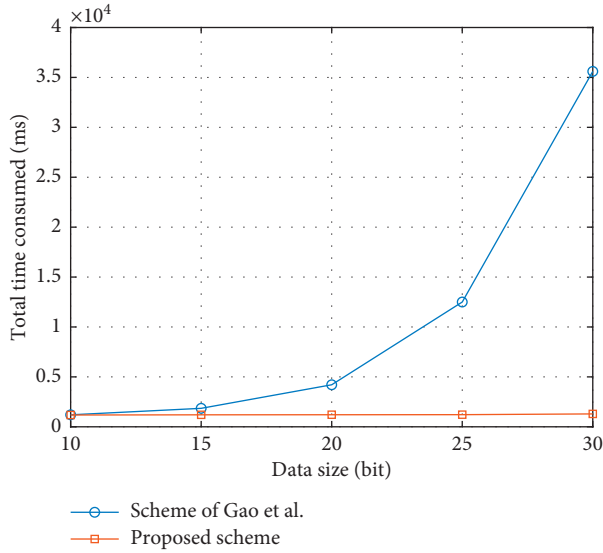


FIGURE 4: Efficiency comparison between the proposed scheme and Gao's scheme.

**6.3. Complexity Analysis.** In this subsection, we review some existing conclusions before analyzing the computational complexity of our scheme. The complexity of calculating modular multiplication and modular exponentiation is  $O(\log^2 p)$  and  $O(\log^3 p)$ , respectively, where  $p$  is the modulus.

First, we analyze the computational complexity of the improved HRES algorithm. The  $E$  involves two modular multiplications, a modular addition and two modular exponentiations. Hence, the computational complexity of this part is  $O(\log^3 p)$ . The  $P_{A \rightarrow A}^1$  only needs a modular exponentiation, so the computational complexity is  $O(\log^3 p)$ . Similarly, the  $P_{A \rightarrow A}^2$  involves a modular exponentiation, a modular inversion, a modular multiplication, a subtraction, and a division. Hence, the

corresponding computational complexity is  $O(\log^3 p)$ . On this basis, we could deduce that the computational complexity of the procedure executed by Alice is  $O(\log^3 p)$ , and the same for Bob, CA, and CB.

## 7. Conclusion

In this paper, we propose a privacy-preserving profile-matching scheme over improved HRES algorithm in mobile social networks. The improved algorithm can support one-time homomorphic multiplication and arbitrarily many homomorphic additions. Compared with the original scheme [2], the key management burden can be reduced, and the privacy problem of users caused by the re-encryption keys leakage can be effectively solved. In addition, our scheme utilizes the cosine result between two normalized vectors as the standard for measuring the users' proximity, which can effectively improve the social experience of the users. Even if users with ulterior motives collude with one of the clouds, the personal data of other users will not be revealed. At last, we prove that our scheme is secure under the semihonest model through strict security analysis.

## Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

## Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This work was funded by the National Key R&D Program of China under grant no. 2017YFB0802000, the National

Natural Science Foundation of China under grant nos. U19B2021 and U1736111, the National Cryptography Development Fund under grant no. MMJJ20180111, the Key Research and Development Program of Shaanxi under grant no. 2020ZDLGY08-04, and the Program for Science and Technology Innovation Talents in the Universities of Henan Province under grant no. 18HASTIT022.

## References

- [1] R. Zhang, J. Zhang, Y. Zhang, J. Sun, and G. Yan, "Privacy-preserving profile matching for proximity-based mobile social networking," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 656–668, 2013.
- [2] C. Gao, Q. Cheng, X. Li, and S. Xia, "Cloud-assisted privacy-preserving profile-matching scheme under multiple keys in mobile social network," *Cluster Computing*, vol. 22, no. 1, pp. 1655–1663, 2019.
- [3] I. Ioannidis, A. Grama, and M. Atallah, "A secure protocol for computing dot-products in clustered and distributed environments," in *Proceedings of the International Conference on Parallel Processing*, IEEE, Washington, DC, USA, pp. 379–384, September 2002.
- [4] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–35, 2018.
- [5] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, pp. 169–178, Bethesda, MD, USA, May 2009.
- [6] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: conceptually-simpler, asymptotically-faster, attribute-based," in *Proceedings of the Annual International Cryptology Conference*, pp. 78–92, Santa Barbara, CA, USA, August 2013.
- [7] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 831–871, 2014.
- [8] D. Boneh, C. Gentry, S. Halevi, F. Wang, and D. J. Wu, "Private database queries using somewhat homomorphic encryption," *Applied Cryptography and Network Security*, in *Proceedings of the International Conference on Applied Cryptography and Network Security*, pp. 102–118, Banff, AB, Canada, June 2013.
- [9] L. Morris, *Analysis of Partially and Fully Homomorphic Encryption*, Rochester Institute of Technology, Rochester, NY, USA, 2013.
- [10] L. Zhang, X. Y. Li, Y. Liu, and T. Jung, "Verifiable private multiparty computation: ranging and ranking," in *Proceedings of the 2013 IEEE INFOCOM*, IEEE, Turin, Italy, pp. 605–609, 2013.
- [11] W. Dong, V. Dave, L. Qiu, and Y. Zhang, "Secure friend discovery in mobile social networks," in *Proceedings of the 2011 IEEE INFOCOM*, IEEE, Shanghai, China, pp. 1647–1655, 2011.
- [12] M. Armbrust, A. Fox, R. Griffith et al., "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.
- [13] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Future Generation Computer Systems*, vol. 28, no. 3, pp. 583–592, 2012.
- [14] B. K. Samanthula, Y. Elmehdwi, G. Howser, and S. Madria, "A secure data sharing and query processing framework via federation of cloud computing," *Information Systems*, vol. 48, pp. 196–212, 2015.
- [15] W. Din, Z. Yan, and R. H. Deng, "Encrypted data processing with homomorphic re-encryption," *Information Sciences*, vol. 409, pp. 35–55, 2017.
- [16] M. Li, N. Cao, S. Yu, and W. Lou, "Findu: privacy-preserving personal profile matching in mobile social networks," in *Proceedings of the 2011 IEEE INFOCOM*, pp. 2435–2443, Shanghai, China, 2011.
- [17] M. Von Arb, M. Bader, M. Kuhn, and R. Wattenhofer, "Veneta: serverless friend-of-friend detection in mobile social networking," in *Proceedings of the 2008 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, IEEE, Avignon, France, pp. 184–189, 2008.
- [18] M. Li, S. Yu, N. Cao, and W. Lou, "Privacy-preserving distributed profile matching in proximity-based mobile social networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 5, pp. 2024–2033, 2013.
- [19] L. Zhang, X. Y. Li, K. Liu, T. Jung, and Y. Liu, "Message in a sealed bottle: privacy preserving friending in mobile social networks," *IEEE Transactions on Mobile Computing*, vol. 14, no. 9, pp. 1888–1902, 2014.
- [20] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.
- [21] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 223–238, Prague, Czech Republic, May 1999.
- [22] M. Bilal and S.-G. Kang, "A secure key agreement protocol for dynamic group," *Cluster Computing*, vol. 20, no. 3, pp. 2779–2792, 2017.
- [23] J. M. Pollard, "Monte Carlo methods for index computation  $\text{mod } p$ ," *Mathematics of computation*, vol. 32, no. 143, p. 918, 1978.
- [24] M. Bilal and S. Pack, "Secure distribution of protected content in information-centric networking," *IEEE Systems Journal*, vol. 14, no. 2, pp. 1921–1932, 2020.
- [25] B. Krishnamurthy and C. E. Wills, "Privacy leakage in mobile online social networks," in *Proceedings of the 3rd Conference on Online Social Networks*, p. 4, USENIX Association, Boston, MA, USA, 2010.
- [26] Y. Lindell, "Secure multiparty computation for privacy preserving data mining," in *Encyclopedia of Data Warehousing and Mining*, pp. 1005–1009, IGI Global, PA, USA, 2005.
- [27] Y. Wang, X. Chen, Q. Jin, and J. Ma, "LIP3: a lightweighted fine-grained privacy-preserving profile matching mechanism for mobile social networks in proximity," in *Proceedings of the International Conference on Algorithms and Architectures for Parallel Processing*, vol. 99, pp. 166–176, Zhangjiajie, China, November 2015.