A Recommendation-based Matchmaking Scheme for Multiple Mobile Social Networks against Private Data Leakage

Yong WANG, Jie HOU, Yuan-wei Tan, Xiao NIE

Department of Computer Science and Communication Engineering
University of Electronic and Science Technology of China, 611731
Chengdu, China

Abstract

Mobile social networks (MSNs) enable users to discover and interact with existing and potential friends both in the cyberspace and in the real world. Although mobile social network applications bring us much convenience, privacy concerns become the key security issue hindering their wide adoptions. In this paper, we propose a recommendation-based matchmaking scheme in multiple MSNs, which can help users to find their potential friends without disclosing their private information. The correctness and security analyzing results show that our scheme can resist both semi-honest and malicious attacks while providing matchmaking functionality against private data leakage.

© 2013 The Authors. Published by Elsevier B.V.
Selection and peer-review under responsibility of the organizers of the 2013 International Conference on Information Technology and Quantitative Management

Keywords: Matchmaking, Mobile Social Network (MSN), Privacy preserving, Homomorphic

1. Introduction

Mobile social networks (MSNs) applications have become popular in our daily lives with the wide usage of personal hand-held mobile devices. MSNs not only enable people to use their existing online social networks at anywhere and anytime, but also introduce a myriad of mobility-oriented applications, e.g., location-based services.

As an important application in the multiple MSNs, matchmaking can help users to find potential friends who are sharing the common attributes (e.g., interests). However, such application also raises a number of privacy concerns. Generally, the two parties involved in the matchmaking don’t wish to reveal any additional information other than what is necessary. If users’ private information is directly exchanged with each other, the sensitive data can be easily collected by the malicious users. Malicious users may also lie or even collude to get others’ private information. Moreover, the wireless medium makes it easy for a malicious user to spoof and inject traffic into the mobile social networks. Therefore, preserving users’ privacy from leakage during the matchmaking becomes an important issue.

In this paper, we propose an recommendation-based matchmaking scheme in multiple MSNs. The scheme includes three phases: (1) Registration phase: users in MSNs publish their public attributes and public relations for
potential friends finding; (2) Recommendation phase: when a user (called initiator) launching a friends finding request, the scheme will provide the candidates according to the similarity of their public attributes and public relations. By this method, the scheme can improve the efficiency of friends finding process; (3) Matchmaking phase: the initiator performs our protocols with the candidates to determine if to make friends. The correctness and security analysis results show that our scheme can resist malicious and semi-honest adversaries while providing privacy-preserving matchmaking services in MSNs.

2. Related Work

2.1. Friend Recommendation

Nowell et al.[1] made recommendation of friends by considering only the local features of graph and compared several local similarity metrics, such as common neighbors, Jaccard’s coefficient, shortest path, and Katz coefficient. Ruturaj et al.[2] found prospective connections for a specific user using the friends-of-friends idea to make recommendation in federated networks. Symeonidis et al.[3] incorporated transitive node similarity into global graph features that captures adequately the missing local graph characteristics and enhance its performance.

Heng et al.[4] proposed a collaboration framework based on local user similarity and global similarity. Vinti et al.[5] proposed a collaborative filtering framework for friends recommendation based on the interaction intensity and the adaptive user similarity.

2.2. Matchmaking protocol

Matchmaking protocol is the core component of a matchmaking system. Matchmaking can be described as a private set intersection (PSI) problem or a private cardinality of set intersection (PCSI) problem. As several solutions adopted by PSI problems correspond to a PCSI scheme, we focus on the related research about PSI protocols, which are classified into three categories as follows:

Agrawal et al.[6] introduced a two party one-way Private Set Intersection (PSI) protocol based on commutative encryption. The protocol is based on the Decisional Diffie-Hellman hypothesis (DDH) assumption and has linear complexity, but it doesn’t take the defense to malicious attacks into consideration. Later, Xie et al.[7] extended[6]’s idea. Except public-key cryptography, there are also some attempts to solve the PSI problem by using simple symmetric-key approaches. However, these works are often flawed. For example, Shundong et al.[8] proposed to use the XOR operation as the symmetric commutative function, which sharply reduces the computational overhead.

Freedman et al.[9] proposed a PSI protocol which is based on polynomial evaluation and additively homomorphic encryption. Since the protocol is one way, it cannot be used in a distributed environment. There are also several researches sketching privacy-preserving set intersection protocols employing the mathematical properties of polynomials. e.g., Kissner and Song[10] used polynomials to represent multi-sets.

There is another PSI construct which is based on oblivious pseudo-random functions. This approach dates backs to Freedmane et al.[11]. Revisiting their idea, Hazay and Lindell[12] utilized specific properties of the Naor-Reingold PRF in order to achieve high efficiency in the presence of both malicious and semi-honest models. Recently, Jarecki and Liu[13] presented a very efficient protocol for computing a pseudo random function with a committed key (informally, this means that the same key is used in all invocations), leading to an efficient PSI protocol. Afterwards, Jarecki et al.[14] proposed an adaptive set intersection computation protocol based on oblivious pseudo-random function. WANG et al.[17] introduced a privacy preserving matchmaking scheme for MSN based on privacy levels, which can help users to find their friends without leaking their privacy information.

3. Preliminaries

3.1. Additive Homomorphic Encryption

Let $e_{pk}()$ denote the encryption with a key pair $(pk, sk)$. An additive homomorphic encryption scheme $e_{pk}$ supports the following operations that can be performed without knowledge of the private key:

- Given two encryptions $e_{pk}(m_1)$ and $e_{pk}(m_2)$, we can efficiently compute the encryption of $(m_1 + m_2)$, denoted by $e_{pk}(m_1 + m_2) = e_{pk}(m_1) + e_{pk}(m_2)$
• Given some constant \( c \) and an encryption \( \varepsilon_{pk}(m) \), we can also efficiently obtain \( \varepsilon_{pk}(cm) = c^h \varepsilon_{pk}(m) \)

This property is satisfied by the Paillier encryption or the ElGamal encryption. Our protocol utilizes the Paillier encryption.

**Threshold Decryption.** Based on Shoup’s threshold version of RSA [15], Damgard et al. [16] proposed a threshold version of Paillier’s encryption scheme. Threshold encryption requires a pre-determined number of players to collaborate on fully decrypting a message. Any collaboration between fewer than the specified number of contributors does not result in a complete decryption.

### 3.2. Our Attributes Representation Technique

We represent the individual elements of a set as prime numbers which is called prime representation. We utilize the prime number table \( P_\eta \) that contains \( \eta \)-bit primes. Then denote by \( \varphi \) a function to a prime table \( \varphi : \{0, 1\}^* \rightarrow P_\eta \) and denote by \( H \) a hash function \( H : \{0, 1\}^* \rightarrow \{1, \cdots, l\} \) where \( l \) is a constant. However, we can see that the function \( \varphi \) is not collision-free and must be accommodated in some way. For that, we define a process that throws prime numbers into \( l \) buckets, such that each bucket contains at most \( m \) elements. Our algorithm is described as:

1. Access to a prime table \( P_\eta \) consisting of \( \eta \)-bit primes;
2. For each \( a_i \in X \), \( \alpha_i = \varphi(a_i) \), add \( \alpha_i \) to a bucket \( B_j \), where \( j = H(a_i) \) for some \( j \in \{1, \cdots, l\} \);
3. Return \( \{B_j\}_l \).

**Brief Analysis.** When it is assumed that the function \( \varphi \) is uniformly random, the probability that collision does not occur in \( m \) results of \( \varphi \) from distinct elements is

\[
\left( 1 - \frac{1}{|P_\eta|} \right) \times \cdots \times \left( 1 - \frac{m}{|P_\eta|} \right) \geq \left( 1 - \frac{m}{|P_\eta|} \right)^m
\]

In our protocol, because the average number of elements in each bucket is small (e.g., \( m \approx 10 \)), the probability that collision by \( \varphi \) occurs in a given bucket is negligible if the size of \( P_\eta \) is sufficiently large (e.g., \( |P_\eta| = 220 \)). Moreover, the size of \( P_\eta \) does not depend on the cardinality of datasets since the problem of large datasets can be addressed by adding the number of buckets. The set of all 20-bit primes is a good example of \( P_\eta \).

### 4. System Design

Our system is designed to help a user (the initiator) to find friends in multiple mobile social networks. As there exist various kinds of relations in the network, each relation can be treated as a single relational network. Every user has a special status in each single relational network. We utilize a cloud based service to receive and process the matchmaking requests sent by the initiator based on the friends-of-friends linkage and the public attributes similarity, the service also recommends the candidates to the initiator to perform the matchmaking protocol.

Each user has two basic categories of attributes: public attributes and private attributes. The public attributes are sent to the registration service to be published while the private attributes are kept by themselves. In our scheme, every two users determine the private attributes types they can access from each other with the corresponding PAS before performing the matchmaking protocol. Besides, we ensure that users can’t reveal any additional information other than what is necessary when interacting with others in the whole matchmaking process.

#### 4.1. System Architecture

Our matchmaking system consists of three components illustrated in figure 1:

A mobile user is an ordinary hand-held device user in multi-relational social networks. Each user has a set of attributes. The Policy-Authorize Service (PAS) is used to authorize elements for a given party. The matcher and the combiner which are assumed to be non-colluding. The matcher and the combiner work together to perform recommending without either of them learning about the association between the users and their attributes or connecting relations through a privacy preserving protocol.
4.2. Protocols design

Our matchmaking scheme consists of three phases: registration, recommendation, and matching:

**Phase 1**: Every user registers his network status and public attributes on the cloud service with a specialized mechanism. After registration, the matcher and combiner will obtain some tables;

**Phase 2**: Once the matcher receives a matchmaking request from the initiator, it communicates with the combiner to perform the recommendation protocol. The initiator will obtain the candidate users to execute the actual matchmaking protocol in phase 3;

**Phase 3**: The initiator performs the matchmaking protocol with all the candidates to find his potential friends.

4.2.1. Phase 1

Each user in our scheme has a unique ID, a public attributes set, a private attributes set, and a set of friends. Assuming Alice’s ID is AID, her public attributes set is: \{P\_1^A, P\_2^A, \cdots, P\_n^A\}, her friend set is: \{F\_1^A, F\_2^A, \cdots, F\_n^A\}. Before registration, she calculates the union of her public attributes sets and friend sets respectively: \( P^A = P\_1^A \cup P\_2^A \cup \cdots \cup P\_n^A = \{p\_a1, p\_a2, \cdots, p\_amu\} \), \( F^A = F\_1^A \cup F\_2^A \cup \cdots \cup F\_n^A = \{f\_a1, f\_a2, \cdots, f\_amu\} \). Now she can register on the cloud service:

**Step 1**: Alice encrypts her public attributes (\( p\_ai \)) and friends (\( f\_ai \)) with the matcher’s public-key (\( pk_M \)), and then with the combiner’s public key (\( pk_C \)). Probabilistic encryption (e.g., by adding randomized padding) is used to defend against dictionary attacks. Then she sends all the double-encrypted attributes and friends to the matcher;

**Step 2**: The matcher picks a random registration ID \( rid_A \) for her registration, and forwards the double-encrypted attributes and friends along with the \( rid_A \) to the combiner. The matcher stores in a table R2U a mapping from \( rid_A \) to the ID of Alice AID.

**Step 3**: Upon receiving the double-encrypted attributes and friends, the combiner decrypts them to reveal \( m\_a \) attributes and \( l\_i \) friends for that registration encrypted by the matcher’s public-key. The combiner picks a random attribute ID \( aid\_i \) for each of the \( i \) encrypted attributes and a random friend ID \( fid\_ai \) for each of the \( i \) encrypted friends. It sends \( aid\_i \), and the \( i\_th \) encrypted attribute, \( fid\_ai \), and the \( i\_th \) encrypted friends to the matcher (one at a time). These messages are mixed with aid/encrypted-attribute pairs, fid/encrypted-friend pairs from other ongoing registrations so the matcher can’t link them back to which registration (\( rid_A \)) they are associated with. If enough natural cover traffic doesn’t exist, the combiner can generate cover traffic. The combiner stores the set of \( aid\_i \) associated with the registration (\( rid_A \)) in the table R2A, and a reverse mapping from the \( aid\_i \) to the \( rid_A \) in table A2R. Similarly, it stores the set of \( fid\_ai \) associated with the registration (\( rid_A \)) in the table R2F, and a reverse mapping from the \( fid\_ai \) to the \( rid_A \) in table F2R.

**Step 4**: The matcher, upon receiving each aid/encrypted attribute pair, decrypts the encrypted attribute to reveal the plain-text attribute \( p\_ai \). At this point, if an equivalent attribute had been stored in table T2A, the matcher puts this attribute and \( p\_ai \) together to construct a set and add \( rid_A \) in the registration ID item associated with the set; otherwise, the matcher stores \( p\_ai \) associated with \( rid_A \). It also stores a reverse-mapping from aid to the plain-text set of equivalent attributes in table A2T. With the same method, the matcher updates table T2F and F2T.
The recommendation protocol

**Step 1:** When the matcher receives Alice’s request, it searches its R2U to obtain her registration ID $r_{id_A}$, then sends $r_{id_A}$ to the combiner;

**Step 2:** The combiner looks up in its R2F table the associate set $\{f_{id_{Ai}}\}$ of the received $r_{id_A}$, then for each $f_{id_{Ai}}$ it executes the following behaviors:

(a) The combiner sends $f_{id_{Ai}}$ to the matcher, the matcher looks up its associate friend $f_{ai}$ in table T2F;

(b) The matcher searches $f_{ai}$’s associated registration ID in R2U table, we assume the search result is $r_{id}$, then the matcher sends $r_{id}$ to the combiner;

(c) Given the registration ID $r_{id}$, the combiner can obtain the associated friend ID set by looking up R2F, then it sends each element of the set to the matcher (one at a time);

(d) The matcher checks its T2F table to find out the friends associated with each element in the friend ID set.

Because each friend is symbolized by its user ID, we obtain the friends of Alice’s $i_{th}$ friend $f_{ai}$ now.

**Step 3:** The matcher obtains friends of Alice’s all friends, combining them with some users selected randomly, we form a user set to join the next filtering;

**Step 4:** For each user $i$ in the set formed in step 3, the matcher finds out its corresponding registration ID $r_{id_i}$ and sends it to the combiner;

**Step 5:** The combiner sets a variable $n_i = 0$ corresponding to $r_{id_i}$ to request his equivalent public attribute numbers with Alice;

**Step 6:** The combiner looks up the R2A table to obtain the attribute ID sets corresponding to $r_{id_A}:\{a_{id_{Ai}}\}$, for each $a_{id_{Ai}}$, it executes the following behaviors:

(a) The combiner sends the $a_{id_{Ai}}$ to the matcher. Upon receiving the message, the matcher looks up in table A2T the plain-text equivalent attribute set associated with that aid. For each attribute in the equivalence set, the matcher looks up table T2A for its corresponding attribute ID. The matcher unions together these attribute IDs to construct the response.

(b) The combiner finds out the registration IDs associated with attribute IDs in the response, if $r_{id_i}$ is included in the registration IDs, $n_i = n_i + 1$.

**Step 7:** The combiner ranks the registration IDs received in step 4 based on their equivalent public attribute numbers with Alice: $n_i$ and sends the registration IDs in the top of the ranked list to the matcher;

**Step 8:** The matcher looks up the user IDs associated with the registration IDs received in step 7 in table R2U.

Now the matcher obtains the candidate users. It sends the recommendation result to Alice.

| Table 1: Recommendation protocol |

After each user finishes registration of public attributes on the registration, the matcher can obtain integrated tables: R2U, T2A, A2T, T2F and F2T, the combiner can obtain integrated tables: R2A, A2R, R2F, and F2R.

4.2.2. Phase 2

When receiving a matchmaking request from the initiator (e.g. Alice), the cloud service will perform the recommendation protocol shown in Table 1 to provide candidate users for her. Our recommendation scheme takes both the social relations and similarity of users’ public attributes into account; it first utilizes the friends-of-friends linkages to filter numerous unrelated users, then ranks the remaining users according to their similarity of public attributes with Alice, the top $n$ users in the ranked list are considered as candidates. We also select users randomly as candidates to avoid the locality.

4.2.3. Phase 3

The initiator (Alice) performs the matchmaking protocol with the candidate users. Our protocol contains two parts where part 1 (Table 2) accomplishes the access authorization of two participants’ private attributes and part 2 (Table 3) realizes the computation of set intersection between Alice and a random candidate user (e.g., Bob). We take the results of part 1 as inputs of part 2.
Part 1 The identification of accessible attributes

Setup: Alice’s private attributes sets of each single relational network are \( \{P^1_A, P^2_A, \cdots, P^m_A\} \) where \( P^i_A = \{p^i_{A1}, p^i_{A2}, \cdots, p^i_{Am}\} \), similarly, Bob’s sets are \( \{P^1_B, P^2_B, \cdots, P^m_B\} \) where \( P^i_B = \{p^i_{B1}, p^i_{B2}, \cdots, p^i_{Bm}\} \).

Step 1: For \( i = 1, 2, \cdots, n \), PAS, issues a BLS signature on each element of \( P^i_A \) and \( P^i_B \). Then Alice and Bob separately obtain \( \{\sigma^i_{A1}, \sigma^i_{A2}, \cdots, \sigma^i_{Am}\} \) and \( \{\sigma^i_{B1}, \sigma^i_{B2}, \cdots, \sigma^i_{Bm}\} \) where \( \sigma^i_{Aj} = H(p^i_{Aj}, P_A)^s \cdot R_A \) and \( \sigma^i_{Bj} = H(p^i_{Bj}, P_B)^s \cdot R_B \), where public known names of Alice and Bob;

Step 2: For each single relational network, Alice and Bob execute the following operations:
(a) Alice and Bob separately select a random value: \( r_A \in_R Z_p \), \( r_B \in_R Z_p \) and let \( R_A = g^{r_A}, R_B = g^{r_B} \). Then Alice sends \( R_A \) to Bob, Bob sends \( R_B \) to Alice;
(b) For each element \( p^i_{Aj} \in P^i_A \), Alice selects a \( c = 1 \in G_p \), then she calculates
\( c = c \cdot e(\sigma^i_{Aj}, R_B) \cdot e(H(p^i_{Aj}, P_A)^s, v_{kB}) \). Only if Bob has a correct signature on \( x \) from authority PAS, \( c \) can be calculated successfully. As a result, Alice obtains \( C^i_A \) which composed of \( c \). With the same method, \( P_B \) can obtain \( C^i_B \).

Step 3: Alice computes \( C_A = \bigcup_{i=1}^n C^i_A \), Bob computes \( C_B = \bigcup_{i=1}^n C^i_B \).

Table 2: ACCESSIBLE ATTRIBUTE AUTHORIZATION

| Accessible Attribute Authority | Description |
|-------------------------------|-------------|
| **Part 2 The computation of set intersection** |

Setup: We use \( C_A, C_B \) obtained in part 1 as the inputs of Alice and Bob and assume the two attributes sets have equal size: \( C_A = \{a_1, a_2, \cdots, a_l\} \), \( C_B = \{b_1, b_2, \cdots, b_l\} \). There is a prime table \( P_q \) consisting of \( q \)-bit primes and \( l \) buckets containing at most \( m \) elements. \( H \) is a random hash function: \( \{0, 1\}^* \rightarrow \{1, 2, \cdots, l\} \), and \( \psi \) represents a Map-To-Prime function. \( \epsilon^{pk}(\cdot) \) denotes a threshold additive homomorphic encryption scheme.

Offline:
Step 1: Alice (resp., Bob) computes \( H(a_i) \) and \( \psi(a_i) \) for each \( a_i \in C_A \) (resp., \( H(b_i) \) and \( \psi(b_i) \) for each \( b_i \in C_B \));
Step 2: For each \( j = 1, 2, \cdots, l \), Alice (resp., Bob) computes \( A_j = \prod_{j=1}^{H(a_i)} \psi(a_i) \) (resp., \( B_j = \prod_{j=1}^{H(b_i)} \psi(b_i) \));
Step 3: Alice (resp., Bob) chooses random elements \( r_1, r_2 \) (resp., \( s_1, s_2 \)) in \( Z_p \) for each bucket.

Online:
Step 1: For each \( j = 1, 2, \cdots, l \), Alice (resp., Bob) computes \( e_{pk}(r_2), e_{pk}(A_j), e_{pk}(r_1A_j^2), (\text{resp, } e_{pk}(s_2), e_{pk}(B_j), e_{pk}(s_1B_j^2)) \) and sends them to Bob (resp., Alice);
Step 2: For each \( j = 1, 2, \cdots, l \), each player computes \( e_{pk}\left((r_1+s_1)A_j^2+(r_2+s_2)B_j^2\right) \) using additive homomorphic property;
Step 3: For each \( j = 1, 2, \cdots, l \), Alice and Bob perform a threshold decryption to obtain \( (r_1+s_1)A_j^2+(r_2+s_2)B_j^2 \);
Step 4: For each \( j = 1, 2, \cdots, l \), each player checks whether \( \psi(a)^2\left((r_1+s_1)A_j^2+(r_2+s_2)B_j^2\right) \) or not for all own private input a whose bucket index is \( j \). If \( \psi(a)^2 \) divides \( (r_1+s_1)A_j^2+(r_2+s_2)B_j^2 \), then \( a \) is included in the intersection \( C_A \cap C_B \);
Step 5: Alice compares the intersection size \( |C_A \cap C_B| \) with its threshold \( e \). If \( |C_A \cap C_B| > e \), she decides to make friend with Bob and sends a notification message to him.

Table 3: Computation of \( C_A \cap C_B \)
5. Security Analysis

In this section, we consider the security of our matchmaking process. We first prove the correctness. That is, when all participants are honest. Alice and Bob can correctly compute $C_A \cap C_B$ in the matchmaking protocol.

**Lemma 1. (Correctness)** If all participants faithfully follow the protocol, then Alice and Bob can correctly compute $C_A \cap C_B$ with overwhelming property in the matchmaking protocol.

**Proof.** When $a$ is an element in the intersection $C_A \cap C_B$, $\varphi(a)$ divides $A_j$ and $B_j$ for the bucket $j = H(a)$. Hence $\varphi(a)^2$ divides $A_j^2$, $B_j^2$ and $(r_1 + s_1) A_j^2 + (r_2 + s_2) B_j^2$. Therefore, each player learns that $a$ is an element in the intersection.

Assume that $a$ is not an element in the intersection $C_A \cap C_B$. We do not consider $a$ is not in $C_A$ and not in $C_B$, since no players try to check the divisibility of $\varphi(a)^2$. Without loss of generality, suppose $a$ is in $C_A$, but not in $C_B$. Then, $\varphi(a)$ divides $A_j$, but does not divide $B_j$. Hence $\varphi(a)^2$ divides $A_j^2$, but does not divide $B_j^2$.

In order that $\varphi(a)^2$ does not divide $(r_1 + s_1) A_j^2 + (r_2 + s_2) B_j^2$, $\varphi(a)^2$ should not divide $r_2 + s_2$. Since $r_2$ and $s_2$ are chosen randomly in $Z_{\sqrt{|d|}}$, the probability that $\varphi(a)^2$ divides $r_2 + s_2$ is $1/\varphi(a)^2$. It is the probability that Alice misunderstands that $a$ belongs to the intersection. When the bit size of primes in $P_\eta$ is 20-bit, the probability becomes about $1/2^{40}$.

5.1. Security in the matchmaking protocols

We can find out that the statuses of two participants in the part 2 of matchmaking protocol are equivalent. We don’t consider the collusion of the two participants. So we only need to analyze the security in the scenario that one participant is an adversary (e.g., Bob) and the other is an honest user (e.g., Alice).

**Adversary model.** The attacker can corrupt some users and eavesdropping any messages sent over channel. We show that no attribute information for a particular Alice is leaked more than that in the protocol (i.e., $C_A \cap C_B$).

**Lemma 2.** Assume that an additive homomorphic threshold encryption $e_{pk}(\cdot)$ is semantically secure, with overwhelming probability, any adversary learns no more information than what would be obtained by using the same private inputs in the ideal model with a trusted third party.

**Proof.** The view of Bob in the part 2 of matchmaking protocol is $C_B$, $e_{pk}(r_2)$, $e_{pk}(A_j^2)$, $e_{pk}(r_1 A_j^2)$, $s_1$, $s_2$, $B_j$, $e_{pk}((r_1 + s_1) A_j^2 + (r_2 + s_2) B_j^2)$ (denoted by DATA-1). Because we have assumed that the additive homomorphic threshold encryption $e_{pk}(\cdot)$ is semantically secure, Bob can’t deduce $r_2$, $r_1$, $A_j$ from the encrypted values: $e_{pk}(r_2)$, $e_{pk}(A_j^2)$, $e_{pk}(r_1 A_j^2)$, we can remove them from DATA-1. After engaging in a threshold decryption, Bob can learns $(r_1 + s_1) A_j^2 + (r_2 + s_2) B_j^2$. So the remain information is $C_B$, $s_1$, $s_2$, $B_j$, $(r_1 + s_1) A_j^2 + (r_2 + s_2) B_j^2$.

In order that Bob learns any information of Alice, he has to find the factor of $A_j$ in the equation $(r_1 + s_1) A_j^2 + (r_2 + s_2) B_j^2 = d$. Since $s_1$, $s_2$, $d$ and $B_j$ are known values to the adversary, it is equivalent to find the factor of an appropriate value of $x$ in the equation $xy + c_1x + c_2z = c_3$ (1) for variables $x$, $y$ and $z$ and constant $c_1$, $c_2$ and $c_3$. The equation (1) can be substituted by the equation $xy + c_1z = c_2(2)$.

As far as we know, Equation (2) has finitely many positive integer solutions but there is no efficient algorithm to find solutions. Hence we believe the following conjecture is true.

For variables $x$, $y$, $z$ and given constant $c_1$, $c_2$, there is no efficient algorithm to find all solutions for equation (2).

Moreover, since $z$ is chosen at random in $Z_{\sqrt{|d|}}$, the number of possible values of $z$ is about $2510$. Hence one has to factor about $2^{510}$ $(c_2 - c_1 z)$’s to solve equation (2).

Hence, by this conjecture, with the remain information $C_B$, $s_1$, $s_2$, $B_j$, $(r_1 + s_1) A_j^2 + (r_2 + s_2) B_j^2$, Bob can’t conclude any information about the private inputs of Alice, with overwhelming probability, except for that given by computing the intersection of their attributes sets.

The execution result of our matchmaking protocol determines that every two participants performing the protocol can learn their intersection mutually no matter whether they make friends with each other. Thus there exists a potential attack that several users collude to conclude the initiator’s private attributes set. Assuming the threshold set by the initiator is $\varepsilon$ and the size of the initiator’s private attributes set is $k$, it needs at least $k/(\varepsilon - 1)$ adversaries colluding to obtain his private attributes set.
5.2. Complexity Analysis

We used integer multiplications, modular exponentiations (ME), and integer divisions in our protocol, in which, ME is the most expensive operation. Hence, we analyze and compare the computational complexity based on the number of MEs.

Each player sends three ciphertexts per bucket. Also each player sends one element to perform a threshold decryption per bucket. Hence, the total number of sent ciphertexts is \( 8l \approx 8k/m \), where \( l \) is the number of buckets, \( k \) is the cardinality of private input sets, and \( m \) is a pre-fixed number which is the bound of the number of elements in a bucket. Therefore, the communication complexity is \( O(k) \).

In case of the computational complexity, it is assumed that the threshold Paillier encryption [12] is utilized, which requires 2 MEs for one encryption and 3 MEs (1 ME for share decryption and 2 MEs for share combining) for a threshold decryption for each player. Hence, for each bucket, each player requires 6 MEs for encryptions, 2 MEs for \( e_{pk} \left( (r_1 + s_1) A_j^2 + (r_2 + s_2) B_j^2 \right) \) computation using additive homomorphic property and 3 MEs for a threshold decryption. Therefore, the total operations are \( 22l \approx 22k/m \) MEs, hence, the computational complexity is \( O(k) \).

6. Conclusion

Applications of matchmaking can help users to find their potential friends, but raise serious privacy leakage issues. In this paper, we present a privacy preserving matchmaking scheme for multiple MSNs utilizing recommendation mechanism, which can help users to find their potential friends without leakage of their unnecessary private data. In the future, we plan to create fully distributed matchmaking protocols to extend the application scenarios. Furthermore, we plan to consider the users’ attributes relations in the development of privacy preserving matchmaking protocols.

References

[1] Nowell LD, Kleinberg J. The link prediction problem for social networks. In: Proceedings of the twelfth international conference on information and knowledge management (CIKM). November 2003, pp. 556-559.
[2] Dhekaner R, Vibber B. Talash: friend finding in federated social networks. In: Proceedings of the LDOV11, March 29, 2011.
[3] Symeonidis P, Tiakas E, Manolopoulos Y. Transitive node similarity for link prediction in social networks with positive and negative links. In: Proceedings of the fourth ACM conference on Recommender systems. RecSys’10, 2010, pp. 183-190.
[4] Heng Luo, Changyong Niu, Ruimin Shen, Carsten Ullrich. A collaborative filtering framework based on both local user similarity and global user similarity. In: Mach Learn, 2008, pp. 231-245.
[5] Agarwal V, Bharadwaj KK. A collaborative filtering framework for friends recommendation in social networks based on interaction intensity and adaptive user similarity. In: Social Network Analysis and Mining, September 2012.
[6] R. Agrawal, A. Evfimievski, and R. Srikant. Information Sharing Across Private Databases. In: Proc. of SIGMOD 2003, pages 86-97.
[7] Q. Xie, U. Hengartner. Privacy-Preserving Matchmaking for Mobile Social Networking Secure Against Malicious Users. In Proc. 9th Intl.Conf. on Privacy, Security, and Trust (PST) 2011, pp. 252-259.
[8] L. Shundong, W. Daoshun, D. Yiqi, and L. Ping. Symmetric Cryptographic Solution to Yao’s Millionaires’ Problem and an Evaluation of Secure Multiparty Computations. Information Sciences, 178(1): 244-255, 2008.
[9] M.J. Freedman, K. Nissim and B. Pinkas. Efficient Private Matching and Set Intersection. In: EUROCRYPT 2004, Springer-Verlag (LNCS 3027), pages 1-19, 2004.
[10] L. Kissner and D.X. Song. Privacy-Preserving Set Operations. In: CRYPTO 2005, Springer-Verlag (LNCS 3621), pp. 241-257, 2005.
[11] N. Freedman, M.J., Ishaï, Y., Pinkas, B., Reingold, O. Keyword search and oblivious pseudorandom functions. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 303-324. Springer, Heidelberg (2005).
[12] Hazay, C., Lindell, Y. Efficient Protocols for Set Intersection and Pattern Matching with Security Against Malicious and Covert Adversaries. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 155-175. Springer, Heidelberg (2008).
[13] Jarecki, S., Liu, X. Efficient Oblivious Pseudorandom Function with Applications to Adaptive OT and Secure Computation of Set Intersection. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 577-594. Springer, Heidelberg (2009).
[14] S. Jarecki and Liu. X. Fast secure computation of set intersection. In: Proc. of Security and Cryptography for Networks (SCN’10), volume 6280 of LNCS, pp. 418-435, 2010.
[15] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In: J. Stern, editor, Advances in Cryptology-EuroCrypt, LNCS 1592, pp. 223-238, 1999.
[16] I. Damgård and M. Jurik. A generalisation, a simplification and some applications of paillier’s probabilistic public-key system. In: K. Kim, editor, Public Key Cryptography, LNCS 1992, pp. 119-136, 2001.
[17] Y. Wang, T. T. Zhang, H. Z. Li, L. P. He, J. PENG. Efficient Privacy Preserving Matchmaking for Mobile Social Networking against Malicious Users. In: Proc. of TrustCom 2012.