Message in a Sealed Bottle: Privacy Preserving Friending in Social Networks

Lan Zhang*, Xiang-Yang Li†
* Department of Software Engineering, Tsinghua University
† Department of Computer Science, Illinois Institute of Technology, and TNLIST, Tsinghua University

***Abstract*—Many proximity-based mobile social networks are developed to facilitate connections between any two people, or to help a user to find people with matched profile within a certain distance. A challenging task in these applications is to protect the privacy the participants' profiles and personal interests.**

In this paper, we design novel mechanisms, when given a preference-profile submitted by a user, that search a person with matching-profile in decentralized multi-hop mobile social networks. Our mechanisms are privacy-preserving: no participants' profile and the submitted preference-profile are exposed. Our mechanisms establish a secure communication channel between the initiator and matching users at the time when the matching user is found. Our rigorous analysis shows that our mechanism is secure, privacy-preserving, verifiable, and efficient both in communication and computation. Extensive evaluations using real social network data, and actual system implementation on smart phones show that our mechanisms are significantly more efficient then existing solutions.

***Index Terms*—Private Profile Matching, Secure Communication, Decentralized Mobile Social Networks.**

I. Introduction

A boom in mobile hand-held devices greatly enriches the social networking applications. Many social networking services are available on mobile phones (*e.g.*, JuiceCaster and MocoSpace) and majority of them are location-aware (*e.g.*, FourSquare, BrightKite and Loopt). However, most of them are designed for facilitating people connection based on their real life social relationship. There is an increasing difficulty of befriending new people or communicating with strangers while protecting the privacy of real personal information.

Friending and communication are two important basic functions of social networks. When people join social networks, they usually begin by creating a profile, then interact with other users. The content of profile could be very broad, such as personal background, hobbies, contacts, places they have been to. Profile matching is a common and helpful way to make new friends with common interest or experience, find lost connections, or search for experts. Some applications help a user automatically find users with similar profile within a certain distance. For example, in the proximity-based mobile social network "*Color*", using the GPS and Bluetooth capabilities on phones, people in close proximity (within 50 meters) can share photo automatically based on their similarity. MagnetU [1] matches one with nearby people for dating, friend-making. Small-talks [24] connects proximate users based on common interests. These applications use profiles to facilitate friending between strangers and also enable privacy preserving people searching to some extent. Observe that in practice the mobile Internet connection may not always be available and it may incur high expense. Thus, the availability of short-range wireless technology such as WiFi and Bluetooth makes it possible to build a new class of proximity-based decentralized (or ad hoc) social networks with mobile phones [21], [27].

However the increasing privacy concern becomes a barrier for mobile ad hoc social networking. People are unwilling to disclose personal profiles to arbitrary persons in physical proximity before deciding to interact with them. The insecure wireless communication channel and potentially malicious service provider increase the risk of revealing private information. Friending based on *private profile matching* allows two users to match their personal profiles without disclosing them to each other. There are two mainstreams of approaches to solve the privacy-preserving profile-based friending problem. The first category treats a user's profile as a set of attributes and provide well-designed protocols to privately match users' profiles based on *private set intersection* (PSI) and *private cardinality of set intersection* (PCSI), [14], [23]. The second category considers a user's profile as a vector and measures the social proximity by *private vector dot product* [9], [12], [28]. They rely on public-key cryptosystem and homomorphic encryption, which results in expensive computation cost and usually requires a trusted third party. Multiple rounds of interactions are required to perform the public key exchange and private matching between each pair of parties, which causes high communication cost in mobile social networks. Presetting (*e.g.* exchange public keys) is often required by these approaches before matching. In the final step of these protocols, only one party learns the matching result, which makes them *unverifiable*. And there lack efficient methods to verify the result. Moreover, in these approaches, matched users and unmatched users all get involved in the expensive computation and learn the matching result (*e.g.* profile intersection) with the initiator despite different similarities between them. Most of them are vulnerable to active attacks like dishonesty and colluding adversary. These limitations hinder the adoption of the SMC-related private matching methods in mobile social networks.

Furthermore, a secure communication channel is also equally important in mobile social networks. Although the matching process is private, the following chatting may still be disclosed to adversary and more privacy may be leaked. Most work assume that there is a secure communication

channel established by using public key encryption system. This involves a trusted third party and key management, which is difficult to manage in decentralized mobile social networks.

Facing these challenges, we first formally define the privacy preserving verifiable computation problem (Section II). We then propose several protocols (Section III) to address the privacy preserving profile matching and secure communication channel establishment in decentralized social networks without any presetting or trusted third party. We take advantage of the common attributes between matching users, and use it to encrypt a message with a secret key in it. In our mechanisms, only a matching user can decrypt the message, and unmatched users learn nothing. A privacy-preserving profile matching and secure channel construction are completed simultaneously with one round communication. The secure channel construction resists the man in the middle attack. A sequence of well-designed schemes make our protocol practical, flexible and lightweight, *e.g.*, a remainder vector is designed to significantly reduce the computation and communication cost of unmatched users. Our profile matching mechanism is also *verifiable* which thwarts cheating about matching result. Both precise and fuzzy matching/search in a flexible form are supported. The initiator can define a similarity threshold, the participants whose similarity is below the threshold learns nothing. We also design mechanism for location privacy preserved vicinity search based on our basic scheme. Compared to most existing work (Section VI) relying on the asymmetric cryptosystem and trust third party, our protocols require no presetting and much less computation. To the best of our knowledge, these are the first privacy-preserving verifiable profile matching protocols based on symmetric cryptosystem.

We rigorously analyze the security and performance of our mechanism (Section IV). We then conduct extensive evaluations on the performances of our mechanisms using large scale social network data, Tecent Weibo. Our evaluation results (Section V) show that our mechanisms outperform existing solutions significantly. We also implement our protocols in laptop and mobile phones and measure the computation and communication cost in real systems. In our mobile-phone implementation, a user only needs about 1.3ms to generate a friending request. For non-candidate users, on average, it takes about 0.63ms to process this friending request, while for candidate users, on average it takes about 7ms to process this request.

II. System Model and Problem Definition

A. System Model

We consider a mobile ad hoc social networking system. When a person joins a social network, he/she usually begins by creating his/her own profile with a set of attributes. The attribute can be anything generated by system or input by users, including his/her location, places he/she has been to, his social groups, experience, interests, contacts, keywords of his/her blogs, etc. According to our analysis of two well-known social networking systems (Facebook and Tecent Weibo), more than 90% users have unique profiles. Thus for most users, the complete profile can be his/her fingerprint in social networks. Then, in most social networks, friending usually takes two typical steps: profile matching and communication. These applications cause a number of privacy concerns.

1) **Profile Privacy**: The profiles of all the participants, including the initiator, intermediate relay users and the matching target, should not be exposed without their consent. For the initiator, the required profile could be his/her own profile or his/her desired person' profile. For other participants, *e.g.*, the unmatched relay users and the matching users, protecting their profiles is necessary and can reduce the barrier to participate in the mobile social networks (MSN). Note that, the exact location information is also a part of the user's profile privacy.
2) **Communication Security**: The messages between a pair of users should be transmitted through a secure communication channel. We emphasize that the secure communication channel establishment has been ignored in most previous work which address the private profile matching in decentralized mobile social networks. In practice, after profile matching, more privacy, even profile information, may be exposed to adversaries via communication through an insecure channel.

In this paper, we address the privacy preserving profile matching and secure communication channel establishment in decentralized social networks without any presetting or trusted third party. We define the problem formally before presenting our mechanism. Each user v_k in a social network has a profile which is a set consisting of m_k attributes, $A_k = \{a_k^1, a_k^2, ..., a_k^{m_k}\}$. The number of attributes is not necessary the same for different users. Each attribute consists of a header indicating its category name and a value field with a single value or multiple values. An initiator v_i represents his/her desired user by a request attribute set with m_t attributes as $A_t = \{a_t^1, a_t^2, ..., a_t^{m_t}\}$. Our mechanism allows the initiator to search a matching user in a flexible way by constructing the request attribute set in the form of $A_t = (N_t, O_t)$. Here

- N_t consists of α *necessary* attributes. All of them are required to be owned by a matching user;
- O_t consists of the rest $m_t - \alpha$ *optional* attributes. At least β of them should be owned by a matching user.

The acceptable *similarity threshold* of a matching user is $\theta = \frac{\alpha+\beta}{m_t}$. Let $\gamma = m_t - \alpha - \beta$. When $\gamma = 0$, an perfect match is required. A *matching user* v_m with a profile A_m satisfies that

$$N_t \subset A_m \text{ and } |O_t \cap A_m| > \beta. \quad (1)$$

When $A_t \subset A_m$, v_m is a perfect matching user. In a decentralized mobile social network, a request will be spread by relays until hitting a matching user or meeting a stop condition, *e.g.* expiration time. When a matching user is found, the initiator v_i and the match user v_m decide whether to connect each other.

B. Adversary Model

In the profile matching phase, if a party obtains one or more users (partial or full) attribute sets without the explicit

consents from those users, it is said to conduct *user profiling* [14]. Two types of user profiling are taken into consideration. In the *honest-but-curious* (HBC) model, a user tries to learn more profile information than allowed by inferring from the information he/she receives and relays but honestly follow the mechanism. In a *malicious* model, an attacker deliberately deviates from the mechanism to learn more profile information. In this work we consider several powerful malicious attacks.

Definition 1 (Dictionary profiling): A powerful attacker who has obtained the dictionary of all possible attributes tries to determine a specific user's attribute set by enumerating or guessing all likely attribute sets.

Most existing private profile matching approaches are vulnerable to the dictionary profiling attack.

Definition 2 (Cheating): Cheating is another threat for most existing private profile matching approaches. In the process of profile matching, a participant may cheat by deviating from the agreed protocol.

In the communication phase, an adversary can learn the content of messages by eavesdropping. The construction of a secure channel may suffer the Man-in-the-Middle (MITM) attack. There are other saboteur attacks. In the deny of service (DoS) attack, an adversary keeps sending profile matching requests. The DoS attack can be prevented by restricting the frequency of relay and reply requests from the same user. Note that, some saboteur behaviors are precluded, such as alter or drop the requests or replies.

C. Design Goal

The main goal and great challenge of our mechanism is to conduct efficient matching against the user profiling and cheating, as well as establish a secure communication channel thwarting the MITM attack and eavesdropping in a decentralized manner without a trust third party. In our mechanism, a user can define a similarity threshold to protect his/her privacy from the user whose similarity is not up to the threshold. Specifically, We define different privacy protection levels $PPL(A_k, v_j)$ of a profile A_k of v_k against a user v_j.

Definition 3 (Privacy Protection Level): Four different privacy protection levels (PPL0, PPL1, PPL2, PPL3) will be discussed in this work. Specifically,

PPL0: If $PPL(A_k, v_j) = 0$, v_j can learn the profile A_k.

PPL1: If $PPL(A_k, v_j) = 1$, v_j can learn the intersection of A_k and A_j.

PPL2: If $PPL(A_k, v_j) = 2$, v_j can learn the α necessary attributes of A_k and the fact that at least β optional attributes are satisfied. Specially, when $\alpha = 0$, v_j learns the fact that the cardinality of $A_k \bigcap A_j$ exceeds the threshold.

PPL3: If $PPL(A_k, v_j) = 3$, v_j learns nothing about A_k.

We design our mechanism to achieve PPL2 against matching user and PPL3 against unmatching user in both HBC and malicious model and thwart cheating. We also optimize the mechanism to reduce the computation cost for unmatched users. Furthermore, in our mechanism human interactions are needed only to decide whether to connect their matching users.

III. Privacy Preserving Profile Matching and Secure Communication

Here we present our lightweight privacy preserving flexible profile matching in decentralized mobile social networks without any presetting or trust third party. A secure communication channel is constructed between matching users.

A. Basic Mechanism

Observe that the intersection of the request profile and the match profile is a nature common secret shared by the initiator and the match user. The main idea of our mechanism is to use the request profile as a key to encrypt a message. Only a matching user, who shares the secret, can decrypt the message with his/her profile efficiently.

Fig. 1 illustrates our basic privacy preserving search and secure channel establishment mechanism. Here, we first draw an outline of how the initiator creates a request package and how a relay user handles the request package.

The initiator starts the process by creating a *request profile* characterizing the matching user and a secret message for him/her. The request profile is a set of *sorted* attributes. Then he/she produces a *request profile vector* by hashing the attributes of the request profile one by one. A *profile key* is generated based on the request profile vector using some publicly known hashing function. The initiator encrypts the secret message with the profile key. A *remainder vector* of the profile vector is yield for fast exclusion by a large portion of unmatched persons. To support a flexible fuzzy search requiring no perfect match, the initiator can define the *necessary attributes*, *optional attributes* and the *similarity threshold* of the matching profile. And a *hint matrix* is constructed from the request vector according to the similarity definition, which enables the matching person to recover the profile key. In the end, the initiator packs the encrypted *message*, the *remainder vector* and the *hint matrix* into a *request package* and sends it out. Note that the required profile vector will not be sent out.

When a *relay user* receives a request from another user, he/she first processes a fast check of his/her own profile vector with the remainder vector. If no sub-vector of his/her profile vector fits the remainder vector, he/she knows that he/she is not a matching user and will forward the request to other relay users immediately. Otherwise, he/she is a *candidate* target and will generate a *candidate profile vector set* by some linear computation with his/her profile and the hint matrix. Then a *candidate profile key set* is obtained. In the basic mechanism, If any key of his/her candidate key set can decrypt the message correctly, he/she is an eligible person and the searching and private messaging complete. Otherwise, he/she just forwards the request to other relay users.

B. Profile Vector and Key Generation

To protect the profile privacy and support a fuzzy search, a cryptographic hash (*e.g.* SHA-256) of the attribute is adopted as the attribute equivalence criterion in this mechanism. However, due to the avalanche effect, even a small change in the input will result in a mostly different hash. Although

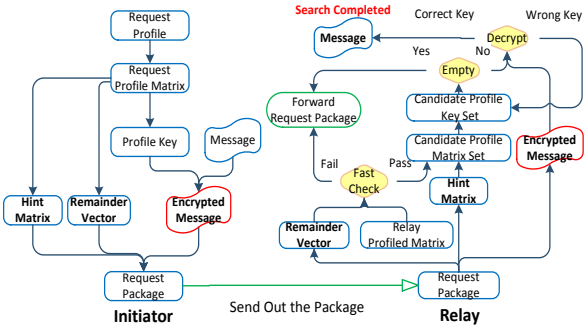


Fig. 1. The procedure to create a request package by the initiator and handle a request package by some forwarder.

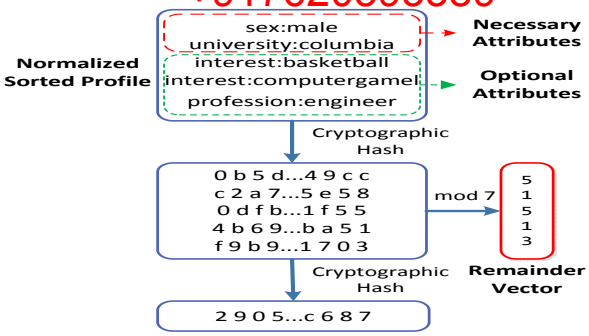


Fig. 2. The procedure to generate the *profile key* and remainder vector with a sample user profile.

consistent attribute name can be provided by the social networking service, the attribute fields or the tags are created by users. So a profile normalization is necessary before the cryptographic hashing to ensure two attributes which are considered equivalent to yield the same hash value. In our mechanism, we use some common techniques to normalize the users profile, including removing whitespace, punctuation, accent marks and diacritics, converting all letters to lower case, converting numbers (dates, currencies, etc.) into words, text canonicalization, expanding abbreviations, converting the plural words to singular form. After the profile normalization, most inconsistences caused by spelling and typing are eliminated. But the sematic equivalence between two different words are not in this paper's consideration.

Assume the cryptographic hash function is $\mathbf{H}$ which yields n-bit length hash value. With an *sorted* normalized profile vector $A_k = [a_k^1, a_k^2, ..., a_k^m]^T$, a *profile vector* H_k is

$$H_k = \mathbf{H}(A_k) = [h_k^1, h_k^2, ..., h_k^m]^T. \quad (2)$$

Here $h_k^i = \mathbf{H}(a_k^i)$, which is n bit long.

A *profile key* is created with H_k,

$$K_k = \mathbf{H}(H_k). \quad (3)$$

Fig. 2 shows the profile vector and key generation of an example profile. With the key of the required profile, the initiator encrypts the secret message using a symmetric encryption technique like Advanced Encryption Standard (AES).

C. Remainder Vector and Hint Matrix

So far, with the profile key, we have realized a naive private profile matching and secure channel establishment. The initiator sends out a message encrypted by the key of the required profile. Any person who receives it tries to decrypt the message with his/her own profile key. Only the exactly matching person will decrypt the message correctly, and he/she can construct a secure communication channel protected by the profile key with the initiator. However, the naive mechanism has some flaws making it unpractical.

1) The search is not *flexible*. The initiator cannot query any subset of other's profile. For example, he/she need to find a "student" studying "computer science" regardless of the "college".
2) An perfect matching is required and *no fuzzy* search is supported. In most case, the initiator need only find some person with profile exceeding the required similarity threshold to the requested profile.
3) All participants must decrypt the message, although most of them hold wrong keys. It wastes the computation resource and increases the search delay.

Improving the naive basic mechanism, our new mechanism allows the initiator to search a person with m_t required attributes in a flexible way $A_t = (N_t, O_t)$, as described in Section II-A. In our mechanism we use a *Remainder Vector* for fast excluding most unmatched users And a *hint matrix* is designed to work with remainder vector to achieve efficient free-form fuzzy search.

1) Remainder Vector: Assume that there are m_t attributes in the required profile, p is a prime integer larger than m_t. A *remainder vector* R_k consists of the remainders of all the hashed attributes in the input profile vector divided by p, as illustrated in Fig. 2.

$$R_k = [h_k^1 \mod p, h_k^2 \mod p, ..., h_k^m \mod p]^T. \quad (4)$$

Then the following theorem is straightforward.

Theorem 1: Given two attributes' hashes $h^i = \mathbf{H}(a^i)$ and $h^j = \mathbf{H}(a^j)$. Remainder $r^i \equiv h^i \mod p$. Remainder $r^j \equiv h^j \mod p$. If $r^i \neq r^j$, then $h^i \neq h^j$.

Assuming the required profile $A_t = [a_t^1, a_t^2, ..., a_t^{m_t}]^T$ by an initiator, the required profile vector is $H_t = [h_t^1, h_t^2, ..., h_t^{m_t}]^T$. Then the remainder vector of the required profile is $R_t = [r_t^1, r_t^2, ..., r_t^{m_t}]^T$. Assume that a relay user's profile vector is H_k. With remainder vector, a relay user simply calculates m_t *candidate attribute subsets* $H_k(r_t^i)$ fitting each r_t^i. Here

$$\forall h_k^x \in H_k(r_t^i): \ h_k^x \mod p = r_t^i. \quad (5)$$

i.e., attributes in $H_k(r_t^i)$ yields the same remainder r_t^i when divided by p.

A combination of one element from each candidate attribute subset forms a profile vector of the relay user. If the candidate attribute subset $H_k(r_t^i)$ is $\emptyset$, the corresponding element in the combination is *unknown* and the relay user fails to meet the required attribute a_t^i according to Theorem 1. The relay user is a *candidate matching user* of the request if there exist at

least one combination which is an ordered set, denoted by H_c, that satisfies the following:

1) The α necessary attributes are all known, *i.e.*

$$H_k(r_t^i) \neq \emptyset, \forall i \leq \alpha; \tag{6}$$

2) The number of unknown elements don't exceed γ, *i.e.*

$$|\{H_k(r_t^i)|\alpha < i < m_t,\ H_k(r_t^i) = \emptyset\}| \leq \gamma; \tag{7}$$

3) Since H_t and H_k are both sorted, the elements of H_c should still keep the order consistent with the relay user's profile vector H_k, *i.e.*

$$\begin{aligned} &\forall h_k^x \in H_c, h_k^y \in H_c. \\ &h_k^x \in H_k(r_t^i), h_k^y \in H_k(r_t^j), i < j \Rightarrow x < y. \end{aligned} \tag{8}$$

We call H_c a *candidate profile vector*.

In a fuzzy search, during the fast checking procedure, if there is no candidate profile vector that can be constructed by the relay user's profile vector, then he/she is not a matching user and she/he can forward the package. An ordinary relay user doesn't have many attributes, commonly dozens of attributes, so there won't be many candidate profile vectors. Using the remainder vector, quick exclusions of a portion of unmatched users can be made.

2) Hint Matrix: A *hint matrix* is constructed to support a flexible fuzzy search. It describes the linear constrain relationship among the $\beta+\gamma$ optional attributes to help calculating γ unknown attributes from β known attributes. The hint matrix helps a matching user exceeding the similarity threshold to recover the required profile vector, so as to generate the correct profile key. Note that when a perfect matching user is required, no hint matrix is needed.

The *constrain matrix* with γ rows and $\gamma + \beta$ columns is defined as:

$$C_{\gamma\times(\gamma+\beta)} = [I_{\gamma\times\gamma}, R_{\gamma\times\beta}]. \tag{9}$$

Here matrix I is a γ dimensional identity matrix, R is a matrix of size $\gamma\times\beta$, each of its elements is a random nonzero integer.

Multiplying the constrain matrix to the optional attributes of the required profile vector, the initiator gets a matrix B:

$$B = C \times [h_t^{\alpha+1}, h_t^{\alpha+2}, ..., h_t^r]^T \tag{10}$$

Then the *hint matrix* M is defined as matrix C, followed by matrix B, *i.e.*,

$$M = [C, B]. \tag{11}$$

When $\gamma > 0$, the initiator generates the hint matrix and sends it with the encrypt message and the remainder vector. In a fuzzy search, after the fast checking procedure by the remainder vector, if the relay user is a candidate matching user, he/she constructs a set of candidate profile vector H_c with unknowns.

Guaranteed by the definition of the candidate profile vector, each H_c has no more than γ unknowns, and any unknown h_c^i, which is the i-th element of H_c, has $i > \alpha$. Now, the unknowns of a candidate profile vector can be calculated by solving a system of linear equations:

$$C \times [h_c^{\alpha+1}, h_c^{\alpha+2}, ..., h_c^r]^T = B \tag{12}$$

Equivalently, we have

$$[I, R] \times [h_c^{\alpha+1}, h_c^{\alpha+2}, ..., h_c^r]^T = B \tag{13}$$

This system of equations has equal to or less than γ unknowns. It can be easily proved that, it has a unique solution. With the solution, a complete candidate profile vector H_c' is recovered. For each H_c', a candidate key $K_c = \mathbf{H}(H_c')$ can be generated. If any of the relay user's candidate keys decrypts the message correctly, he/she is a matching user and gets the encrypted secret. Else, he/she forwards the request to the next person.

D. Location Attribute and Its Privacy Protection

In localization enabled mobile social networks, a user usually searches matching users in vicinity. In the existing systems, a user is required to provide his/her own current location information and desired search range. However the user's accurate location will be exposed. In our mechanism, location is considered as a dynamic attribute that will be updated while the user moves. Accurate location is also the user's privacy. To conduct location privacy preserved vicinity search, we design a scheme for location attribute matching based on lattice. Our location attribute matching scheme is compatible with our privacy preserved profile matching mechanism. When generating the profile vector, we use lattice based hashing to hash a user's location and his/her vicinity region. A private vicinity search can be easily conducted via our fuzzy search scheme with the help of hint matrix.

1) Lattice based Location Hashing: We map the two-dimensional location to the hexagonal lattice. The lattice point is a discrete set of the centers of all regular hexagons, as the dots shown in Fig. 3. The lattice is formally defined as:

$$\{x = u_1 a_1 + u_2 a_2 | (u_1, u_2) \in \mathbb{Z}^2\} \tag{14}$$

Here a_1, a_2 are linearly independent primitive vectors which span the lattice. Given the primitive vectors a_1, a_2, a point of the lattice is uniquely identified by the integer vector $u = (u_1, u_2)$. There are infinite choices for a_1, a_2. Let d denote the shortest distance between lattice points, for simplicity, we choose the primitive vectors as presented in Figure 3:

$$a_1 = (d, 0);\ a_2 = (\frac{1}{2}d, \frac{\sqrt{3}}{2}d). \tag{15}$$

Defining a geography location as the origin point O and the scale of the lattice cell d, with the lattice definition, a location can be hashed to its nearest lattice point. Any two locations hashed to the same lattice point are inside a single hexagonal lattice cell, and they are separated by a bounded distance d.

Given a user v_k's current location $l_k(t)$ at time t and his/her vicinity range D, his/her vicinity region can be hashed to a *lattice point set*, $V_k(O, d, l_k(t), D)$, consisting of the center lattice point, i.e. the hash of $l_k(t)$, and other lattices points whose distances to the center lattice point are less than D.

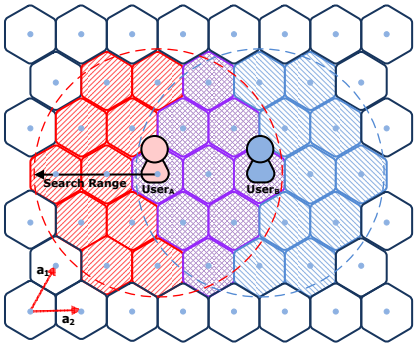


Fig. 3. Lattice-based location hash.

2) Location Privacy Preserved Vicinity Search: Intuitively, given the distance bound to define vicinity, if two users are within each other's vicinity, the intersection of their vicinity regions will have a proportion no less than a threshold Θ. The initiator calculates his/her vicinity lattice point set $V_i(O, d, l_i(t), D)$ by defining O, d and D, here D indicates his/her search range. Therefore, if a user v_k is in his/her vicinity, then v_k's vicinity lattice point set $V_k(O, d, l_k(t), D)$ should satisfy the requirement:

$$\theta_k = \frac{|V_i(O, d, l_i(t), D) \bigcap V_k(O, d, l_k(t), D)|}{|V_k(O, d, l_k(t), D)|} \geq \Theta \qquad (16)$$

In the example in Fig. 3, user v_A's vicinity range $D = 3d$. The lattice points within the red slashed hexagons constitute $V_A(O, d, l_A(t), D)$ and the lattice points within the blue back slashed hexagons constitute $V_B(O, d, l_B(t), D)$. The purple grid region is the intersection of vicinity regions of v_A and v_B. In this case, the threshold Θ should be $\frac{9}{19}$. In other words, if $V_B(O, d, l_B(t), D)$ contains no less than 9 common lattice points of $V_A(O, d, l_A(t), D)$, he/she is in the vicinity of v_A. In practice, the initiator can adjust the parameter d to change the cardinality of the vicinity lattice point set to a suitable size.

To conduct location privacy preserved vicinity search, the initiator won't send his/her vicinity lattice point set directly. Using the sorted lattice points, he/she generates a dynamic profile key, a dynamic remainder vector and a dynamic hint matrix in the same way as processing other attributes. So a vicinity search works as a fuzzy search with similarity threshold Θ. Only participants in his/her vicinity who has a certain amount of common lattice points with him/her can generate the correct dynamic profile key with the help of the dynamic remainder vector and hint matrix.

3) Location Based Profile Matching: Compared to static attributes like identity information, location is usually a temporal privacy. With the *dynamic key* generated from location information, we can improve the protection of a user's static attributes. When constructing profile vector of a user, we can hash the concatenation of each static attribute and his/her current dynamic key instead of directly hash static attributes. Since the dynamic key changes when the user updates his/her location, the hash values of the same static attribute are completely different for users at different locations. It will greatly increase the difficulty for the malicious adversary to conduct dictionary profiling. However, it won't bring much more computation for ordinary user because of quite limited lattice points in his/her vicinity range.

E. Profile Matching Protocols

We are now ready to present our privacy preserving profile matching protocols. Here we present three different protocols that achieve different levels of privacy protection.

Protocol 1: Privacy Preserving Profile Matching

1) The initiator encrypts a random number x and a public predefined confirmation information in the secret message by the required profile key, say $E_{K_t}(confirmation, x)$. And he/she sends the request package out.
2) A candidate relay user can verify whether he/she decrypts the message correctly by the confirmation information. If he/she does not match, he/she just forwards the request to the next user. If he/she is matching, he/she can reply the request by encrypting the predefined ack information and a random number y along with any other message (e.g. the intersection cardinality) by x, say $E_x(ack, y)$, and sends it back to the initiator.

1) Protocol 1: The scheme of Protocol 1 is based on the basic mechanism. Under Protocol 1, a unmatched relay user doesn't know anything about the request. The matching user knows the intersection of required profile and his/her own profile after step 1 in the HBC model. A matching user can decide whether to reply the request according to the profile intersection. The initiator doesn't know anything about any participant until he/she gets a reply. With replies, he/she knows the matching users and even the most similar replier by the cardinality information. Then he/she can start secure communication with a matching user protected by $x + y$ or with a group of matching user protected by x. However, in malicious model, if the matching user has a dictionary of the attributes and the size of dictionary is not large enough unfortunately, he/she can learn the whole request profile by the recovered profile vector.

2) Protocol 2: To prevent malicious participants, we design Protocol 2, which is similar to Protocol 2, but it excludes the confirmation information from the encrypted message.

Under Protocol 2, after the first round of communication, the participants won't known anything about the request in both HBC model and malicious model. The initiator knows who are the matching users and even the most similar one according to the replies. Then the initiator can start secure communication with a matching user protected by the key $x + y$ or with a group of matching users protected by x. In malicious model, if a participant has a dictionary of the attributes, he/she may construct a large candidate profile key set and send it to the initiator. The main difference between an ordinary user and a malicious user with a dictionary is the size of their attribute space. An ordinary user with about dozens of attributes can make a quick reaction and reply a small size acknowledge set.

Protocol 2: Privacy Preserving Profile Matching

1) The initiator encrypts a random number x in the secret message without any confirmation information by the required profile key, say $E_{K_t}(x)$.
2) A candidate matching user cannot verify whether he/she decrypts the message correctly. Let the candidate profile key set of a candidate user be $\{K_c^1, K_c^2, \ldots, K_c^z\}$. He/she decrypts the message in the request with each candidate profile key to get a set of numbers, say $X = \{x_j | x_j = D_{K_c^j}(E_{K_t}(x))\}$. Then he/she encrypts the predefined ack information and a random number y by each x^j as the key, and sends back the acknowledge set $\{E_{x^j}(ack, y)\}$ back to the initiator, for a public ack.
3) The initiator excludes the malicious replier whose response time exceeds the time window or the cardinality of reply set exceed the threshold.

While it takes much longer for a malicious user due to a large number of candidate attribute combinations. So the initiator can identify the malicious repliers by response time and the cardinality of reply set.

Using our profile matching mechanism, it is impossible to build an attribute dictionary in this social networking system. If an adversary constructs the attribute dictionary from other sources, *e.g.*, other similar social networking systems, the space of attributes are mostly very large, which makes the dictionary profiling infeasible. Especially in the localizable social mobile social networks, the vast dynamic location attribute will greatly increase the difficulty of dictionary profiling. However, there still may exist a special case that the attribute space is not large enough in some social networks. In this case, in Protocol 1, the request profile may be exposed via dictionary profiling by malicious participants. Although protocol 2 protects the request profile from any participants, a malicious initiator may learn the profile of unmatching repliers.

3) Protocol 3: To prevent the dictionary profiling by malicious initiator, we improve Protocol 2 to Protocol 3 which provides a user personal defined privacy protection.

Definition 4 (Attribute Entropy): For an attribute a^i with t^i values $\{x_j : j = 1, \ldots, t^i\}$. $P(a^i = x_j)$ is the probability that the attribute a^i of a user equals x_j. The entropy of the attribute a^i is $S(a^i) = -\sum_{j=1}^{t^i} P(a^i = x_j) \log P(a^i = x_j)$.

Definition 5 (Profile Entropy): The entropy of a profile A_k is $S(A_k) = \sum_{i=1}^{m_k} S(a^i)$.

Intuitively, the larger the entropy of a profile, the more privacy information is contained in the profile. A participant can determine his/her personal privacy protection level by given a acceptable profile entropy leakage upper limit φ. Based on the user defined protection level, Protocol 3 is φ-entropy private for each user.

Definition 6 (φ-Entropy Private): A protocol is φ-entropy private when the entropy of possible privacy leakage is not greater than the upper limit φ: $S(Leak(A_k)) \leq \varphi$.

The parameter φ is decided by each user. Here we suggest two options to determine φ.

(1) *K-anonymity based*. To prevent from being identified by disclosed attributes, the user will only send out messages protected by the profile key generated from the subset of attributes which at least k users own the same subset. Let a subset of A_k be $A_k^s = a^1, \ldots, a^l$. Suppose that the t^i values of a^i have equal probability, then there are $\frac{n}{\prod_{i=1}^{l} t^i}$ users are expected to own the same subset. If the user require that $\frac{n}{\prod_{i=1}^{l} t^i} \geq k$, then $\log \prod_{i=1}^{l} t^i \leq \log \frac{n}{k}$, ie $S(A_k^s) \leq \log \frac{n}{k}$. So the parameter φ could be $\log \frac{n}{k}$.

(2) *Sensitive attributes based*. The user can determine the sensitive attributes which must not be disclosed according to the current context. Let the set of sensitive attributes defined by user v_k is A_k^s, then $\varphi = \min(S(a^i))$, where $a^i \in A_k^s$.

Protocol 3: Privacy Preserving Profile Matching

1) The initiator encrypts a random number x in the secret message without any confirmation information by the required profile key, say $E_{K_t}(x)$.
2) A candidate matching user cannot verify whether he/she decrypts the message correctly. He/she selects a set of candidate profile $\{A_c^1, A_c^2, \ldots, A_c^z\}$ which satisfies that $S(\bigcup_{i=1}^{z} A_c^i) \leq \varphi$. And he/she generates the corresponding candidate profile keys $\{K_c^1, K_c^2, \ldots, K_c^z\}$. He/she decrypts the message in the request with each candidate profile key to get a set of numbers, say $X = \{x_j | x_j = D_{K_c^j}(E_{K_t}(x))\}$. Then he/she encrypts the predefined ack information and a random number y by each x^j, and sends back the acknowledge set $\{E_{x^j}(ack, y)\}$ back to the initiator.
3) The initiator excludes the malicious replier whose response time exceeds the time window or the cardinality of reply set exceed the threshold and decrypts the replies with x. If he/she gets a correct ack information from a reply, the corresponding replier is matching.

Protocol 3 is private when the initiator is not malicious. and it is φ-private for each participant even when the initiator can conduct a dictionary profiling.

Note that, for all three protocols, each request has a valid time. An expired request will be dropped. And each user has a request frequency limit, all participants won't reply the request from the same user within a short time interval.

F. Establishing Secure Communication Channel

As presented in the profile matching protocols, the random number x generated by the initiator and y generated by a matching user have been exchanged secretly between them. x and y is the communication key shared by a pair of matching users. So the secure peer to peer communication channel can be constructed. Our mechanism realizes key exchange between matching users which is resistant to the man in the middle attack.

Furthermore, our mechanism can be easily adopted to discover the community consisting of users with similar profile as the initiator and establish the group key x for secure intra-community communication.

IV. Security and Efficiency Analysis

In this section, we analyze the security and performance of our profile matching and secure communication channel establishing mechanism.

A. Security and Privacy Analysis

1) Profile Privacy: During the generation of the profile key, we don't use the hash value of the static attribute directly. Instead, we use the hash value (with hash function SHA-256) of the combination of the static attribute and the dynamic attribute (*i.e.* location), which greatly increases the protection of the static attribute. We use AES with 256 bit key as the encryption method. The 256-bit profile key is used as the secret key to encrypt the message by AES. Only the encrypted message will be transmitted, and no attribute hash value will be transmitted. No one can obtain other user's attribute hash. Therefore no one can build a dictionary of hash values of attributes via this social networking system. To acquire the profile information of the initiator or other participants the attacker need to decrypt the request/reply message correctly and confirm the correctness. This is extremely difficult due to the choice of SHA-256 and 256-bit-key AES.

TABLE I
THE PRIVACY PROTECTION LEVEL OF OUR PROTOCOLS IN HBC MODEL. v_I IS THE INITIATOR, v_M IS A MATCHING USER AND v_U IS A UNMATCHING USER. A_I, A_M AND A_U ARE THEIR CORRESPONDING PROFILES.

PPL	(A_I, v_M)	(A_I, v_U)	(A_M, v_I)	(A_U, v_I)
Protocol 1	1	3	2	3
Protocol 2	3	3	2	3
Protocol 3	3	3	2	3
PSI	3	3	1	1
PCSI	3	3	$\lvert A_I \cap v_U \rvert$	$\lvert A_I \cap v_U \rvert$

In the honest-but-curious (HBC) model each user only knows his/her own attributes. Therefore, only users owning matching attributes can generate the same profile key and decrypt each other's messages correctly. Unmatched user cannot obtain any information from the encrypted message. Table I presents the protection level of our three protocols during the matching procedure in HBC model. Compared to the existing PSI and PCSI approaches, our protocols provide Level 2 privacy protection for matching users and don't leak any information to unmatching users.. After profile matching, under Protocol 1, the initiator knows nothing until the matching user replies; under Protocol 2 and 3, a user won't learn he/she is matching until the initiator contacts him/her.

In the malicious model, for the adversary who intends to learn the attributes hash to build a dictionary, it is impossible in our social networking system, as a result of that no attribute information is transmitted in any data packets. But we still consider the unlikely case that an adversary constructs an attribute dictionary from other sources, *e.g.*, other similar social networking systems. We suppose that the adversary can eavesdrop all communication.

In most cases, the cardinality m of the dictionary is very large, which makes the dictionary profiling difficult. With a remainder vector, it takes an adversary $(\frac{m}{p})^{m_t}$ guesses to compromise a user's profile with m_t attributes. Here p is a small prime number like 11. In tencent weibo, we found that $m \simeq 2^{20}$ and the average attribute number of each user is 6. When the adversary tries to guess a user's profile by brute force, it will take about 2^{100} guesses. If considering keywords of a user, m is even larger. Especially in localizable mobile social networks, the vast dynamic location attribute will greatly increase the attribute set. The dictionary profiling gets more infeasible.

However, the worst case may still exist that the adversary obtains the attribute dictionary and the dictionary size is not large enough. This kind of adversary also compromises other PSI based approaches. Table II shows the protection level of our protocols in this worst case. In this case, our Protocol 1 cannot protect the initiator's privacy from the dictionary profiling. But it provides Level 2 privacy protection for replying matching users and unconditional Level 3 privacy for unmatching users. Protocol 2 provides unconditional Level 3 privacy for the initiator. It also provides unconditional Level 3 privacy for all participants against any other person except the initiator. Only if the initiator is an adversary with the dictionary, he/she may compromise the candidate user's privacy. Although candidate users are just a small potion of users, to improve Protocol 2 to protect their privacy, we design Protocol 3 which still provides unconditional Level 3 privacy for the initiator and incardinate users, and Level 3 privacy for the candidate users against any other person except the initiator. Moreover, it provides personal defined φ-entropy private for all candidate users against an malicious initiator.

2) Communication Security: Our protocols realize secure key exchange between matching users based on their common attributes. The shared secret key is protected by the profile key, only the user who owns the matching attributes can generate the same profile keys. As the security analysis of our protocols, Protocol 2 and 3 can construct a secure communication between a matching user and the initiator against any other adversary in both HBC and malicious model. So our secure communication channel establishment between matching users thwarts the MITM attack.

3) Verifiability: In majority of existing profiling matching approaches, only one party learns the result and tells the other party. Hence, one party can lie about the result. There lacks a feasible way for the other party to verify the result. Our protocols are verifiable and resists cheating. In Protocol 1, after the matching, only the matching user learns the result and he/she contacts the initiator by relying $E_x(ack, y)$. In the HBC model, an unmatching user cannot obtain the correct x encrypted by the request profile key, so he/she cannot cheat the initiator to pretend to be matching. In Protocol 2 and 3, similarly, only the matching user can get the correct x

TABLE II

THE PRIVACY PROTECTION LEVEL OF OUR PROTOCOLS IN MALICIOUS MODEL WITH SMALL DICTIONARY. v_I IS THE INITIATOR, v_M IS A MATCHING USER AND v_U IS A UNMATCHING USER. A_I, A_M AND A_U ARE THEIR CORRESPONDING PROFILES. v'_I IS A MALICIOUS INITIATOR WITH A PROFILE DICTIONARY. v'_P IS A MALICIOUS PARTICIPANT WITH A PROFILE DICTIONARY EAVESDROPPING THE COMMUNICATION.

PPL	(A_I, v'_P)	(A_M, v'_I)	(A_M, v'_P)	(A_U, v'_I)	(A_U, v'_P)
Protocol 1	0	2	2	3	3
Protocol 2	3	2	3	3 (noncandidate) A_c (candidate)	3
Protocol 3	3	φ-entropy	3	3 (noncandidate) φ-entropy (candidate)	3

Consequently, the initiator can only obtain the correct y of matching user. So both the initiator and participants cannot cheat each other. In the malicious model with a dictionary, it takes an adversary much longer time to guess the correct key. Hence a nonmalicious user can distinguish the adversary by his/her reply delay. So no cheating can be conducted successfully with our protocols.

B. Performance Analysis

1) Computational Cost: For an initiator, it takes $m_t \log m_t$ operations for sorting attributes, $m_t + 1$ hashing operations for profile key generation and m_t modulo operations for remainder vector generation. $\gamma(\gamma+\beta)$ operations are needed to calculate the hint matrix if the required similarity $\theta < 100\%$. In addition, one symmetric encryption is needed with the profile key.

For a participant v_k, it takes $m_k \log m_k$ operations for sorting its attributes, m_k hashing for profile vector generation and m_k modulo operations for remainder calculation. After fast checking by remainder vector, if a participant v_k is not a candidate user, no more computation is need. If v_k is a candidate user, let the number of his candidate profile vector be κ_k. It takes this user κ_k hashing to generate candidate profile keys. If there is a hint matrix, user v_k need to solve κ_k $\gamma(\gamma+\beta)$-dimension linear equation systems. The computation cost is $\kappa_k m_t^3$. In the end, κ_k symmetric decryptions with the profile key. Note that the expected κ_k is $\varepsilon(\kappa_k) = \binom{m_k}{\alpha+\beta} \times \left(\frac{1}{p}\right)^{\alpha+\beta}$. For example, in Tencent Weibo the average attribute number is 6 and the maximum number is 20. Then if $\alpha + \beta = 6$, even a large $m_k = 20$ and small prime number $p = 11$ result in a very small $\varepsilon(\kappa_k) = 0.02$. So it takes small computation cost even for a candidate user. We can show that the expected candidate users is only a small portion of all users and the portion decreases greatly with the increase of m_t and p. It may be considered that larger p will weaken the security due to the decreased difficulty of dictionary profiling. Our testing and analysis show that even a small p, *e.g.*, $p = 11$, can significantly reduce the number of candidate users. So an initiator can choose a proper p to efficiently control the amount of candidate users as well as achieve the secure protection of profile privacy.

In implementation, we use SHA-256 as the hashing method and AES as the symmetric encryption method. The data processing performance of SHA-256 is 111MB/s for a single-threaded implementation on an Intel 1.83GHz processor in 32-bit operation system. Furthermore, for all users the sorting and hashing results are calculated once and used repetitively until the attributes are updated, *e.g.*, the location changes. AES performs well on a wide variety of hardware, from 8-bit smart cards to high-performance computers. The throughput of AES is about 60MB/s on a 1.7GHz processor and about 400MB/s on Intel i5/i7 CPUs. Compared to the existing approaches, we don't use an asymmetric-key cryptosystem and the remainder vector can significantly reduce the computation of unmatching users. In all, our protocol are computationally efficient.

2) Communication Cost: In our protocols there isn't any pre-operation for key exchange or secure channel construction. To conduct profile matching with all users, the initiator only need one broadcast to send the request to all participants. The request consists of a $32m_t$ bit remainder vector and a 256 bit encrypted message. If the required similarity $\theta < 100\%$, there is also a $32\gamma(\gamma+\beta)+256\gamma$ bit hint matrix. So the size of the request message is at most $(1-\theta)32m_t^2+(288-256\theta)m_t+256$ bits. For example, the user of Tencent Weibo has 6 tags in average and 20 tags at most. To search a 60% similar user, the request is about 190B in average and 1KB at most. In Protocol 1, only the matching user will reply the request. So the transmission cost of Protocol 1 is one broadcast and $O(1)$ unicasts. In Protocol 2, only the candidate matching user will reply the request. So the transmission cost of Protocol 2 is one broadcast and $O(n*(\frac{1}{p})^{m_t\theta})$ unicasts. For example, when $p = 11$, $m_t = 6$, $\theta = 0.6$, there are only about $\frac{1}{5610}$ fraction of users will reply. In Protocol 3, the communication cost of reply is even smaller than Protocol 2 because of the personal privacy setting. Note that the reply in all three protocols is only 32Byte. So the communication cost of our protocols is quite small. This makes our protocols suitable for wireless communication environment, for example the mobile social networks.

3) Further Comparison With Related Work: In this section, we mainly compare the computation cost and communication cost between our protocols and PSI and PCSI based approaches. The computation time comparison with dot product based on approach presented in [9].

First we define the basic operations of other asymmetric cryptosystem based approaches:

- M_1: 24-bit modular multiplication;
- M_2: 1024-bit modular multiplication;
- M_3: 2048-bit modular multiplication;
- E_2: 1024-bit exponentiation;

- E_3: 2048-bit exponentiation.

Then we define the basic operation in our protocol:

- $\mathcal{H}$: one SHA-256 hashing operation of an attribute;
- $\mathcal{M}$: is the operation that the 256-bit hash of an attribute modulo the small prime p;
- $\mathcal{E}$: AES-256 encryption;
- $\mathcal{D}$: AES-256 decryption.

Table III presents the computation and communication cost of related work and our protocol. Because the SHA-256, modulo, and AES-256 operations in our protocol are much cheaper than the 1024-bit and 2048-bit modular multiplication and exponentiation, our protocol costs much less computation than asymmetric-key based schemes. The transmitted data size are significantly reduced because basically only one encrypted 256-bit message and one $32 * m_t$ bit remainder vector need to be transmitted. Furthermore, the remainder vector eliminates the reply from most unmatching users. So the total transmission time is also reduced.

V. Evaluation Using Real Data

A. Real Social Networking System Analysis

Our evaluations are based on the profile data of Tencent Weibo [2]. Tencent Weibo is one of the largest micro-blogging websites in China, which is a platform for building friendship and sharing interests online. This dataset has 2.32 million users personal profiles, including the following information: the year of birth, the gender, the tags and keywords. Tags are selected by users to represent their interests. Keywords are extracted from the tweet/retweet/comment of a user, and can be used as features to better represent the user.

In the Tencent Weibo dataset, the cardinality of the tag set is 560419 and the cardinality of the keywords set if 713747. Each user has 6 tags in average and 20 tags at most to represent his/her interest. On average, 7 keywords extracted for a user. The maximum number of keywords of a user is 129. So when the adversary tries to guess a user's profile with 6 tags by brute force, it will take him/her about 10^{30} guesses. Moreover, the large attribute space makes the vector-based matching approaches impractical.

There is a question that, would many users own the same profile? Each profile is a combination of several attributes. When more than one users have the same profile, we say there are collisions for this profile. Figure 4 shows that both in Tencent and Facebook more than 90% users have unique profiles.

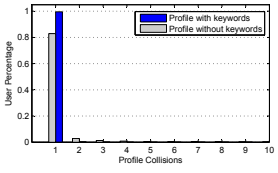


Fig. 4. Profile uniqueness and collisions of users.

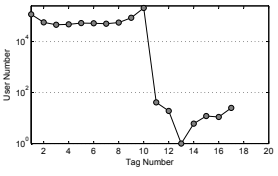


Fig. 5. Users' attribute number distribution.

B. Computation Performance

We then exam the computation performance of our protocols on mobile devices and PC. The mobile phone is HTC G17 with $1228Hz$ CPU, 1GB ROM and 1GB RAM. The laptop is Think Pad X1 with i7 2.7GHz CPU and 4GB RAM. Table IV shows the mean execution time of basic computation. And we evaluate our protocol on the Tencent Weibo dataset, Table VI presents the breakdown of time cost for our protocol. As an example, for a user with 6 attributes, the time need to generate a request is only about 3.9×10^{-2} ms on laptop and 1.3 ms on mobile phone. On average it takes a non-candidate user about 3.9×10^{-2} ms on laptop and 0.63 ms on phone to process the request. A little more extra computation is required for a candidate user, and the computation cost is about 4×10^{-2} ms on laptop and 7 ms on phone for each candidate key.

The time cost of all the operations in our protocols are quite small compared with the computation time of existing asymmetric cryptosystem based approaches, *e.g.* the evaluation result of [9]. Table V-B shows a typical scenario in a mobile social network with 100 users. The numbers of attributes are chosen based on the analysis of Tencent Weibo. With the numerical comparison, it is clearly that our protocol is efficient in both computation and communication.

TABLE IV
Mean computation time for our basic operation.(ms)

	SHA-256	Mod p	AES Enc
Laptop	1.2×10^{-3}	3.1×10^{-4}	8.7×10^{-4}
Phone	4.8×10^{-2}	5.7×10^{-2}	2.1×10^{-2}
	Multiply-256	Compare-256	AES Dec
Laptop	1.4×10^{-4}	1.0×10^{-5}	9.6×10^{-4}
Phone	3.2×10^{-2}	1.0×10^{-3}	2.5×10^{-2}

TABLE V
Mean computation time for basic operations for asymmetric cryptosystem based scheme.(ms)

	1024-exp	2048-exp	1024-mul	2048-mul
Laptop	17	120	2.3×10^{-2}	1×10^{-1}
Phone	34	197	1.5×10^{-1}	2.4×10^{-1}

TABLE VI
Decomposed computation time of our protocols based on the Tencent Weibo dataset.(ms)

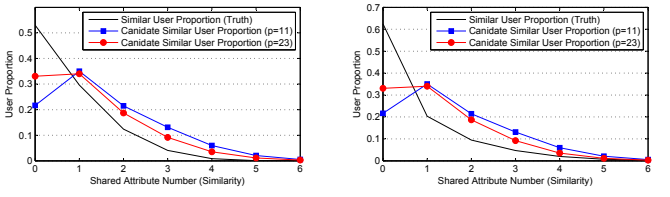
Laptop			
	Mean	Min	Max
MatrixGen	7.2×10^{-3}	1.0×10^{-3}	2.4×10^{-2}
KeyGen	8.1×10^{-3}	2.3×10^{-3}	2.5×10^{-2}
RemainderGen	1.9×10^{-3}	3.1×10^{-4}	6.2×10^{-3}
HintGen	4.7×10^{-3}	2.8×10^{-4}	5.6×10^{-2}
HintSolve	3×10^{-2}	1.1×10^{-3}	1.1
Phone			
	Mean	Min	Max
MatrixGen	2.6×10^{-1}	4.4×10^{-2}	8.9×10^{-1}
KeyGen	6.3×10^{-2}	4.8×10^{-2}	1.4×10^{-1}
RemainderGen	3.4×10^{-1}	5.7×10^{-2}	1.14
HintGen	1.2	1.4×10^{-1}	12
HintSolve	6.9	2.6×10^{-1}	250

TABLE III
COMPARISON OF EFFICIENCY WITH EXISTING SCHEME.$q = 256$

	Party	FNP [10]	FC10 [7]	Advanced [14]	Protocol 1
Computation	P_1 P_k	$(2m_t + m_k n)E_3$ $m_k \log m_t E_3$	$(2.5m_t n)M_2$ $(m_t + m_k)E_2$	$(3m_t n)E_3$ $2m_t E_3$	$(m_t + 1)\mathcal{H} + m_t\mathcal{M} + \mathcal{E}$ $m_k\mathcal{H} + m_k\mathcal{M}$ (noncandidate) $\kappa_k\gamma^2(\gamma + \beta) + (m_k + \kappa_k)\mathcal{H}$ $+m_k\mathcal{M} + \kappa_k\mathcal{D}$ (candidate)
Communication bits	All	$8q(m_t + m_k n)$	$4qn(3m_t + m_k)$	$24[m_t m_k n$ $+tn(8m_t + 2m_k + 12m_t t)]$ $+16qm_t n$	$(1-\theta)32m_t^2 + (288 - q\theta)m_t$ $+q + qn * (\frac{1}{p})^{m_t\theta}$
Communication Transimission	All	1 broadcast n unicasts	$2n$ unicasts	$5n$ unicasts	1 broadcast $n * (\frac{1}{p})^{m_t\theta}$ unicasts

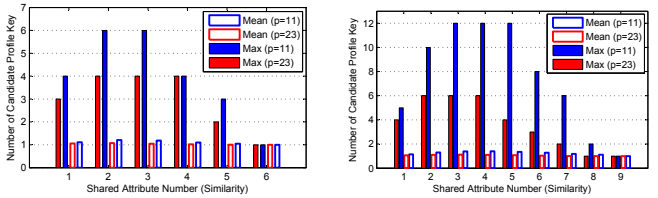
TABLE VII
COMPARISON OF EFFICIENCY WITH EXISTING SCHEME IN TYPICAL SCENARIO. $m_t = m_k = 6, \gamma = \beta = 3, p = 11, n = 100, t = 4$.

	Party	FNP [10]	FC10 [7]	Advanced [14]	Protocol 1
Computation	P_1 P_k	$612E_3$ $5E_3$	$1500M_2$ $12E_2$	$1800E_3$ $12E_3$	$7\mathcal{H} + 6\mathcal{M} + \mathcal{E}(P_1)$ $6\mathcal{H} + 6\mathcal{M}$ (noncandidate) $27\kappa_k + (6 + \kappa_k)\mathcal{H} + 6\mathcal{M} + \kappa_k\mathcal{D}$ (candidate)
Computation(ms)	P_1 P_k	73440 600	34.5 204	216000 1440	$1.1 \times 10^{-2}(P_1)$ 3.1×10^{-3}(noncandidate) $6\kappa_k \times 10^{-3} + 9.1 \times 10^{-3}$ (candidate)
Communication(KB)	All	151	300	704	0.22
Communication Transimission	All	1 broadcast 100 unicasts	200 unicasts	500 unicasts	1 broadcast number of candidates unicasts



(a) users with 6 attributes (b) diverse numb. of attributes

Fig. 6. Candidate user proportion with different similarity and prime number.



(a) users with 6 attributes (b) diverse num. of attributes

Fig. 7. Size of candidate profile key set with different similarities.

C. Performance Evaluations

Based on the user attributes of Tencent Weibo, we evaluate the efficiency of our protocols. Two typical situations are taken into consideration. In the first case, all users have equal size of attributes which is similar to the vector based scheme. We use the attribute data of all 52248 users with 6 attributes. In the second case, we randomly sample 1000 users from all users to conduct profile matching.

In both cases, we exam the similarity between all pairs of users as the ground truth. Then we run our profile matching protocols at different similarity levels. Figure 6 shows the number of candidate users of our protocol change with similarity requirement and the prime number p. The result shows that in both cases, the number of candidate users approaches the number of true matching users with increasing p. And a small p can achieve small size of candidate users and significantly reduce unwanted computation and communication cost for unmatching users.

There is a worry that, the candidate profile key set may be very large for candidate users. Figure 7 presents the number of candidate profile keys during the matching with different similarity level and prime p. The result shows that in real social networking system like Tencent Weibo, the candidate key set is small enough to achieve efficient computation for candidate users.

VI. RELATED WORK

A. Privacy Preserving Friending

Most previous private matching work are based on the secure multi-party computation (SMC) [25]. There are two mainstreams of approaches to solve the private profile-based friending problem. The first category is based on private set intersection (PSI) and private cardinality of set intersection (PCSI). Early work in this category mainly address the private set operation problem in database research, *e.g.* [3], [7], [10], [13], [19], [26]. Work like [17] studied the problem of private k nearest neighbor search. Later on some work treat a user's profile as multiple attributes chosen from a public set of attributes and provide well-designed protocols to privately match users' profiles based on PSI and PCSI, [14], [23]. The second category is based on private vector dot product [12]. [9] considers a user's profile as vector which represents his/her social coordinate, and the social proximity between two uses as the matching metric. It calculates the metric by

private dot product. A trusted central server is requited to precompute users social coordinates and generate certifications and keys. [28] improves these work with a fine-grained private matching by associating a user-specific numerical value with every attribute to indicate the level of interest. However, most these approaches lacks a specific definition of matching user. For example, in the PSI based schemes, any user can learn the profile intersection with any other user despite the similarity between them. The PCSI and dot product based approaches cannot support a precise specific profile matchings.

These protocols often rely on public-key cryptosystem and/or homomorphic encryption, which results in expensive computation cost and usually requires a trusted third party. Even non-matching users involve in the expensive computation. Multiple rounds of interactions are required to perform the private profile matching between each pair of parties. They all need preset procedure, *e.g.* exchanging public keys before matching, precomputing vector [9], establishing secure communication channel and share secret [14]. Furthermore, these protocols are unverifiable.

B. Establishing Secure Channel

Secure communication channel construction is very important in practical private friending system but is often ignored. Secure communication channels are usually set up by authenticated key agreement protocols. This can be performed by relying on a public-key infrastructure, e.g., based on RSA or the Diffie-Hellman protocol [8]. The public-key based methods allow parties to share authenticated information about each other, however they need a trusted third party. Although Diffie-Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key, it needs multiple interactions between two parties and is known to be vulnerable to the Man-in-the-Middle attack.

Device pairing is a another technique to generate a common secret between two devices that shared no prior secrets with minimum or without additional hardware. Examples include the ”resurrecting duckling” [20], ”talking-to-strangers” [4], ”seeing-is-believing” [15] and Short Authenticated Strings (SAS) based key agreement [22] and [16]. However, they employ some out-of-band secure channel to exchange authenticated information or leverage the ability of users to authenticate each other by visual and verbal contact. In addition, the interaction cost is still not well suited to decentralized mobile social networks where secure connections are needed between all parties at once. With these existing schemes, it is more complicated to establish a group key.

C. Attribute Based Encryption

Attribute based encryption is designed for access control of shared encrypted data stored in a server. Only the user possessing a certain set of credentials or attributes is able to access data. It was introduced by Sahai and Waters [18], and then later improved in [5], [6], [11]. All the ABE schemes rely on asymmetric-key cryptosystem, which cost expensive computation.

VII. Discussion and Conclusion

In this paper, we design a novel symmetric-encryption based privacy-preserving profile matching and secure communication channel establishment mechanism in decentralized social networks without any presetting or trusted third party. We take advantage of the common attributes between matching users to encrypt a secret message with a channel key in it. Several protocols were proposed for achieving different levels of privacy. We rigorously compared the performances of our protocols with existing protocols and conducted extensive evaluations on the performances using a large scale dataset from real social networking.

References

[1] Magnetu. http://magnetu.com.
[2] Tencent weibo. http://t.qq.com/.
[3] Agrawal, R., Evfimievski, A., and Srikant, R. Information sharing across private databases. In *Proc. ACM SIGMOD*, 2003, pp. 86–97.
[4] Balfanz, D., Smetters, D., Stewart, P., and Wong, H. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proc. NDSS*, 2002, pp. 7–19.
[5] Bethencourt, J., Sahai, A., and Waters, B. Ciphertext-policy attribute-based encryption. In *Proc. IEEE S&P*, 2007, pp. 321–334.
[6] Chase, M. Multi-authority attribute based encryption. *Theory of Cryptography*, 2007, pp. 515–534.
[7] De Cristofaro, E., and Tsudik, G. Practical private set intersection protocols with linear complexity. *Financial Cryptography and Data Security*, 2010, pp. 143–159.
[8] Diffie, W., and Hellman, M. New directions in cryptography. *IEEE Transactions on Information Theory*, 1976, pp. 644–654.
[9] Dong, W., Dave, V., Qiu, L., and Zhang, Y. Secure friend discovery in mobile social networks. In *Proc. IEEE INFOCOM*, 2011, pp. 1647–1655.
[10] Freedman, M., Nissim, K., and Pinkas, B. Efficient private matching and set intersection. In *Advances in Cryptology-EUROCRYPT* , 2004, pp. 1–19.
[11] Goyal, V., Pandey, O., Sahai, A., and Waters, B. Attribute-based encryption for fine-grained access control of encrypted data. In *Proc. ACM CCS*, 2006, pp. 89–98.
[12] Ioannidis, I., Grama, A., and Atallah, M. A secure protocol for computing dot-products in clustered and distributed environments. In *Proc. IEEE ICPP*, 2002, pp. 379–384.
[13] Kissner, L., and Song, D. Privacy-preserving set operations. In *Advances in Cryptology–CRYPTO*, 2005, pp. 241–257.
[14] Li, M., Cao, N., Yu, S., and Lou, W. Findu: Privacy-preserving personal profile matching in mobile social networks. In *Proc. IEEE INFOCOM*, 2011, pp. 2435–2443.
[15] McCune, J., Perrig, A., and Reiter, M. Seeing-is-believing: Using camera phones for human-verifiable authentication. In *IEEE S&P*, 2005, pp. 110–124.
[16] Pasini, S., and Vaudenay, S. Sas-based authenticated key agreement. *Public Key Cryptography*, 2006, pp. 395–409.
[17] Qi, Y., and Atallah, M. Efficient privacy-preserving k-nearest neighbor search. In *IEEE ICDCS*, 2008, pp. 311–319.
[18] Sahai, A., and Waters, B. Fuzzy identity-based encryption. *Advances in Cryptology–EUROCRYPT*, 2005, pp. 557–557.
[19] Sathya Narayanan, G., Aishwarya, T., Agrawal, A., Patra, A., Choudhary, A., and Pandu Rangan, C. Multi party distributed private matching, set disjointness and cardinality of set intersection with information theoretic security. *Cryptology and Network Security*, 2009, pp. 21–40.
[20] Stajano, F., and Anderson, R. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Security Protocols*, 2000, pp. 172–182.
[21] Stuedi, P., Riva, O., and Alonso, G. Enabling social networking in ad hoc networks of mobile phones. *Proc. VLDB Endowment*, 2009, pp. 1634–1637.

[22] VAUDENAY, S. Secure communications over insecure channels based on short authenticated strings. In *Advances in cryptology–CRYPTO*, 2005, pp. 309–326.

[23] VON ARB, M., BADER, M., KUHN, M., AND WATTENHOFER, R. Veneta: Serverless friend-of-friend detection in mobile social networking. In *IEEE WIMOB*, 2008, pp. 184–189.

[24] YANG, Z., ZHANG, B., DAI, J., CHAMPION, A., XUAN, D., AND LI, D. E-smalltalker: A distributed mobile system for social networking in physical proximity. In *Proc. IEEE ICDCS*, 2010, pp. 468–477.

[25] YAO, A. Protocols for secure computations. In *Proc. the 23rd Annual Symposium on Foundations of Computer Science*, 1982, pp. 160–164.

[26] YE, Q., WANG, H., AND PIEPRZYK, J. Distributed private matching and set operations. *Information Security Practice and Experience*, 2008, pp. 347–360.

[27] ZHANG, L., DING, X., WAN, Z., GU, M., AND LI, X. WiFace: a secure geosocial networking system using WiFi-based multi-hop MANET. In *Proc. the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*, 2010, pp. 1–8.

[28] ZHANG, R., ZHANG, Y., SUN, J., AND YAN, G. Fine-grained private matching for proximity-based mobile social networking. In *Proc. IEEE INFOCOM*, 2012.