Privacy-Preserving Profile Matching for Proximity-Based Mobile Social Networking

Rui Zhang, *Member, IEEE,* Jinxue Zhang, Yanchao Zhang, *Senior Member, IEEE,*
Jinyuan Sun, *Member, IEEE,* and Guanhua Yan

***Abstract*—Proximity-based mobile social networking (PMSN) refers to the social interaction among physically proximate mobile users. The first step toward effective PMSN is for mobile users to choose whom to interact with. Profile matching refers to two users comparing their personal profiles and is promising for user selection in PMSN. It, however, conflicts with users' growing privacy concerns about disclosing their personal profiles to complete strangers. This paper tackles this open challenge by designing novel fine-grained private matching protocols. Our protocols enable two users to perform profile matching without disclosing any information about their profiles beyond the comparison result. In contrast to existing coarse-grained private matching schemes for PMSN, our protocols allow finer differentiation between PMSN users and can support a wide range of matching metrics at different privacy levels. The performance of our protocols is thoroughly analyzed and evaluated via real smartphone experiments.**

***Index Terms*—Proximity-based mobile social networking; profile matching; privacy.**

I. Introduction

THE EXPLOSIVE popularity of portable mobile devices such as smartphones and tablets are fostering the emergence of proximity-based mobile social networking (PMSN), which refers to adjacent mobile users interacting through the Bluetooth/WiFi interfaces on their mobile devices. In contrast to traditional web-based online social networking, PMSN can enable more tangible face-to-face social interactions in public places such as parks, stadiums, and train stations.

The first step toward effective PMSN is for mobile users to choose whom to interact with. As an example, Alice wants to conduct PMSN with nearby passengers at the airport. Since she can simultaneously interact with only one or a few persons, it is crucial for her to select those who can lead to the most meaningful social interactions. A naive method is to choose those with valid public-key certificates, which nevertheless can only establish the identities of the certificate holders. This method is thus insufficient if Alice has multiple candidate neighbors who are all strangers and each have a valid public-key certificate. A more viable method is for Alice to select those whose social profiles most match hers. Widely known as *profile matching*, this method is rooted in the social fact that people normally prefer to socialize with others having similar interests or background over complete strangers.

A major challenge for profile matching is to ensure the privacy of personal profiles which often contain highly sensitive information related to gender, interests, political tendency, health conditions, and so on. This challenge necessitates *private matching*, in which two users compare personal profiles without disclosing them to each other. Private matching for PMSN has been recently addressed in [2]–[4]. Assuming that each user's personal profile comprises multiple attributes chosen from a public set of attributes such as various interests [2], friends [3], or disease symptoms [4], these schemes could enable two users to find the intersection or intersection cardinality of their profiles without disclosing additional information to either party. These schemes, however, cannot well distinguish the users with the same attribute(s). For instance, there are three users all fond of football (i.e., a common attribute), but they watch two/two/seven games a week, respectively. The first two apparently have a better match, but the existing schemes [2]–[4] will all result in the same level of profile similarity between every two users.

We propose *fine-grained* private matching for PMSN to tackle the above challenge. In contrast to coarse-grained private matching [2]–[4] with binary personal profiles, our solution features fine-grained personal profiles in which each attribute is associated with a user-specific integer value indicating the corresponding user's association with this attribute. For example, an attribute value in [0, 10] for the "football" attribute indicates the level of interest in football from no interest (i.e., 0) to extremely high interest (i.e., 10). Our solution enables two users to measure the similarity between their fine-grained personal profiles according to the same or different matching metrics. The privacy provision and overhead of our solution is theoretically analyzed, and its high efficacy and efficiency are confirmed through real smartphone implementations and experiments.

The rest of the paper is organized as follows. Section II formulates fine-grained private matching for PMSN. Section III presents our fine-grained private-matching protocols. Section V analyzes and evaluates the performance of the

Manuscript received March 1, 2012; revised July 15, 2012. This work was supported in part by the US National Science Foundation under grants CNS-1117462 and CNS-0844972 (CAREER). The preliminary version of this paper appeared in IEEE INFOCOM'12 [1].

R. Zhang is with the Department of Electrical Engineering, University of Hawaii, Honolulu, HI 96822 (e-mail: ruizhang@asu.edu).

J. Zhang and Y. Zhang are with the School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe, AZ 85287 (e-mail: {jxzhang, yczhang}@asu.edu).

J. Sun is with the Department of Electrical Engineering and Computer Science, University of Tennessee-Knoxville, Knoxville, TN 37996 (e-mail: jysun@eecs.utk.edu).

G. Yan is with the Los Alamos National Laboratory, Los Alamos, NM 87545 (e-mail: ghyan@lanl.gov).

Digital Object Identifier 10.1109/JSAC.2013.SUP.0513057

proposed protocols. Section VI discusses the related work. Section VII concludes this paper.

II. Problem Formulation and Cryptographic Tool

A. Proximity-based Mobile Social Networking (PMSN)

We assume that each user carries a mobile device such as a smartphone or tablet with the same PMSN application installed. The PMSN application can be developed by small independent developers or offered by online social network providers as a function module of their applications built for mobile devices. For convenience only, we shall not differentiate a user from his mobile device later.

A PMSN session involves two users and consists of three phases. First, two users need discover each other in the neighbor-discovery phase. Second, they need compare their personal profiles in the matching phase. Last, two matching users enter the interaction phase for real information exchange. Our work is concerned with the first and second phases.

The PMSN application uses fine-grained personal profiles. In particular, the application developer defines a public attribute set consisting of d attributes $\{A_1, \dots, A_d\}$, where d may range from several tens to several hundreds depending on specific PMSN applications. The attributes may have different meanings in different contexts, such as interests [2], disease symptoms [4], or friends [3]. For easier illustration, we hereafter assume that that each attribute corresponds to a personal interest such as movie, sports, and cooking. To create a personal profile, every user selects an integer $u_i \in [0, \gamma - 1]$ to indicate his level of interest in A_i (for all $i \in [1, d]$) the first time he uses the PMSN application. As a fixed system parameter, γ could be a small integer, say 5 or 10, which may be sufficient to differentiate user's interest level. The higher u_i, the more interest the user has in A_i, and vice versa. Every personal profile is then defined as a vector $\langle u_1, \dots, u_d \rangle$. The user can also modify his profile later on as needed.

B. Problem Statement: Fine-Grained Private Matching

We consider Alice with profile $\mathbf{u} = \langle u_1, \dots, u_d \rangle$ and Bob with profile $\mathbf{v} = \langle v_1, \dots, v_d \rangle$ as two exemplary users of the same PMSN application as shown in Fig. 1. Let $\mathcal{F}$ denote a set of candidate matching metrics defined by the PMSN application developer, where each $f \in \mathcal{F}$ is a function over two personal profiles that measures their similarity. Our private-matching protocols allow Alice and Bob to either negotiate one common metric from $\mathcal{F}$ or choose different metrics from $\mathcal{F}$ according to their individual needs. We shall focus on the latter more general case henceforth, in which private matching can be viewed as two independent protocol executions, with each user initiating the protocol once according to her/his chosen matching metric. Assume that Alice chooses a matching metric $f \in \mathcal{F}$ and runs the privacy-matching protocol with Bob to compute $f(\mathbf{u}, \mathbf{v})$. According to the amount of information disclosed during the protocol execution, we define the following three privacy levels from Alice's viewpoint, which can also be equivalently defined from Bob's viewpoint for his chosen matching metric.

***Definition** 1:* **Level-I privacy**: *When the protocol ends, Alice only learns $f(\mathbf{u}, \mathbf{v})$, and Bob only learns f.*

***Definition** 2:* **Level-II privacy**: *When the protocol ends, Alice only learns $f(\mathbf{u}, \mathbf{v})$, and Bob learns nothing.*

***Definition** 3:* **Level-III privacy**: *When the protocol ends, Alice learns if $f(\mathbf{u}, \mathbf{v}) < \tau_A$ holds for her personal threshold τ_A without learning $f(\mathbf{u}, \mathbf{v})$, and Bob learns nothing.*

If Alice and Bob both faithfully follow the protocol execution, which corresponds to the *honest-but-curious* (HBC) model [8], neither of them can learn the other's personal profile for all three privacy levels. In addition, with level-I privacy, although Bob cannot learn $f(\mathbf{u}, \mathbf{v})$, he learns the matching metric f chosen by Alice. In contrast to level-I privacy, level-II privacy additionally requires that Bob learn nothing other than $f \in \mathcal{F}$. Finally, level-III privacy discloses the least amount of information by also hiding $f(\mathbf{u}, \mathbf{v})$ from Alice.

Alice or Bob may actively deviate from the protocol execution, which corresponds to the *malicious* model [8]. For example, Bob may manipulate the protocol output by using an arbitrary profile and/or not faithfully following the protocol operations (e.g., by changing intermediate computation results). In Section III-D, we will discuss possible active attacks, their impact, and corresponding countermeasures.

There might also be some *external* attackers (other than Alice and Bob) trying to infer users profile or disrupt PMSN operations. For example, an external attacker may eavesdrop on the messages between Alice and Bob. All our protocols can ensure that the eavesdroppers are completely blind to the profiles of Alice and Bob and the matching metric(s) chosen by them, which will all be encrypted during the protocol execution. For simplicity, we will neglect external eavesdroppers in subsequent protocol illustrations and analysis.

It is beyond the scope of this paper to consider other possible external attacks due to tight space constraints. For example, attackers may launch a jamming attack by purposefully transmitting radio signals to prevent Alice and Bob from exchanging messages. Moreover, an intelligent attackers may launch the man-in-the-middle (MiM) attack by surreptitiously relay messages between Alice and Bob who are not physically proximate. These attacks are not unique to our private-matching scenario, and similar ones can apply to any wireless protocol involving message exchanges between multiple parties. The jamming attack can be thwarted by spread-spectrum techniques [11], [12] and the MiM attack can be tackled by using the device pairing protocol in [13].

C. Cryptographic Tool: Paillier Cryptosystem

Our protocols rely on the Paillier cryptosystem [10], and we assume that every PMSN user has a unique Paillier public/private key pair which can be generated via a function module of the PMSN application. How the keys are generated and used for encryption and decryption are briefed as follows to help illustrate and understand our protocols.

- **Key generation.** An entity chooses two primes p and q and compute $N = pq$ and $\lambda = \mathrm{lcm}(p-1, q-1)$. It then selects a random $g \in \mathbb{Z}^*_{N^2}$ such that $\gcd(\mathsf{L}(g^\lambda \bmod N^2), N) = 1$, where $\mathsf{L}(x) = (x-1)/N$. The entity's Paillier public and private keys are $\langle N, g \rangle$ and λ, respectively.

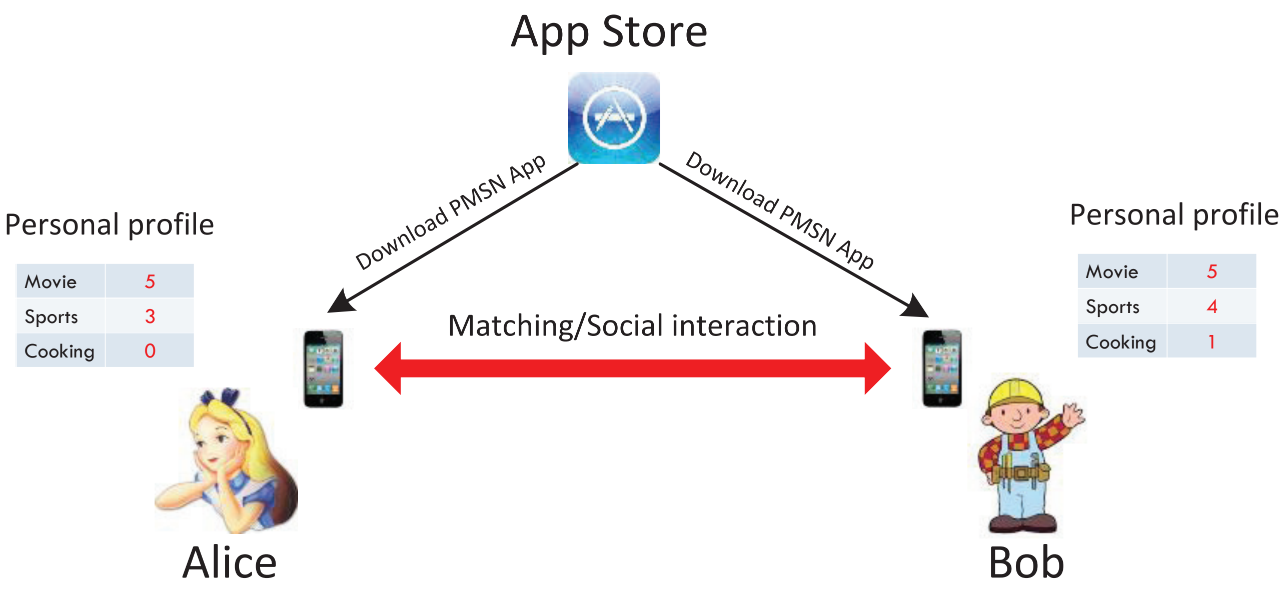


Fig. 1. Illustration of proximity-based mobile social networking and profile matching.

- **Encryption.** Let $m \in \mathbb{Z}_N$ be a plaintext to be encrypted and $r \in \mathbb{Z}_N$ be a random number. The ciphertext is given by

$$\mathsf{E}(m \mod N, r \mod N) = g^m r^N \mod N^2 , \quad (1)$$

where $\mathsf{E}(\cdot)$ denotes the Paillier encryption operation using public key $\mathsf{pk} = \langle N, g \rangle$. To simplify our expressions, we shall hereafter omit the modular notation inside $\mathsf{E}(\cdot)$.
- **Decryption.** Given a ciphertext $c \in \mathbb{Z}_{N^2}$, the corresponding plaintext can be derived as

$$\mathsf{D}(c) = \frac{\mathsf{L}(c^\lambda \mod N^2)}{\mathsf{L}(g^\lambda \mod N^2)} \mod N , \quad (2)$$

where $\mathsf{D}(\cdot)$ denotes the Paillier decryption operation using private key $\mathsf{sk} = \lambda$ hereafter.

The Paillier's cryptosystem has two very useful properties.

- **Homomorphic.** For any $m_1, m_2, r_1, r_2 \in \mathbb{Z}_N$, we have

$$\mathsf{E}(m_1, r_1)\mathsf{E}(m_2, r_2) = \mathsf{E}(m_1 + m_2, r_1 r_2) \mod N^2,$$
$$\mathsf{E}^{m_2}(m_1, r_1) = \mathsf{E}(m_1 m_2, r_1^{m_2}) \mod N^2 .$$

- **Self-blinding.**

$$\mathsf{E}(m_1, r_1) r_2^N \mod N^2 = \mathsf{E}(m_1, r_1 r_2) ,$$

which implies that any ciphertext can be changed to another without affecting the plaintext.

The Paillier cryptosystem is suitable for our scenario for a number of reasons. First, our private-matching protocols rely on homomorphic cryptographic primitives, while the Paillier cryptosystem is the most commonly used one [5], [14]. Second, unlike other homomorphic cryptosystems such as RSA, the Paillier cryptosystem is semantically secure for sufficiently large N and g, which means that it is infeasible for a computationally bounded adversary to derive significant information about a message (plaintext) from its ciphertext and the corresponding public key. In addition, the Java implementation of Paillier cryptosystem is readily available [15] and can be easily ported to smartphones.

To facilitate our illustrations, we assume that N and g are of 1024 and 160 bits, respectively, for sufficient semantical security of the Paillier cryptosystem [5]. Under this assumption, a public key $\langle N, g \rangle$ is of 1184 bits, a ciphertext is of 2048 bits, a Paillier encryption needs two 1024-bit exponentiations and one 2048-bit multiplication, and a Paillier decryption costs essentially one 2048-bit exponentiation.

III. Fine-Grained Private Matching Protocols

In this section, we present three private-matching protocols to support different matching metrics and offer different levels of privacy. In particular, Protocol 1 is for the ℓ_1-distance matching metric and can offer level-I privacy, Protocol 2 supports a family of additively separable matching metrics and can offer level-II privacy, and Protocol 3 is an enhancement of Protocol 3 for supporting level-III privacy.

A complete matching process involves Alice with profile $\mathbf{u} = \langle u_1, \ldots, u_d \rangle$ and Bob with profile $\mathbf{v} = \langle v_1, \ldots, v_d \rangle$, each running an independent instance of the same or even different private-matching protocol. Let f denote any matching metric supported by Protocols 1 to 3. The larger $f(\mathbf{u}, \mathbf{v})$, the less similar $\mathbf{u}$ and $\mathbf{v}$, and vice versa. We can thus consider $f(\mathbf{u}, \mathbf{v})$ some kind of distance between $\mathbf{u}$ and $\mathbf{v}$. Assume that Alice has a threshold τ_A and will accept Bob if $f(\mathbf{u}, \mathbf{v}) < \tau_A$. Similarly, Bob has a threshold τ_B and will accept Alice if $f(\mathbf{u}, \mathbf{v}) < \tau_B$. If both accept each other, they can start real information exchange. Our subsequent protocol illustrations and analysis will be from Alice's viewpoint, which can be similarly done from Bob's viewpoint. We assume that Alice has a Paillier public key $\langle N, g \rangle$ and the corresponding private key λ, which are generated as in Section II-C. A practical security protocol often involves some routines such as using timestamps to mitigate replay attacks and message authentication codes for integrity protection. To focus on explaining our key ideas, we will neglect such security routines in protocol illustrations.

A. Protocol 1 for Level-I Privacy

Protocol 1 is designed for the ℓ_1 distance as the matching metric. Recall that every personal profile is a vector of dimension d. As probably the most straightforward matching metric, the ℓ_1 distance (also called the Manhattan distance) is computed by summing the absolute value of the element-wise subtraction of two profiles and is a special case of the more general ℓ_α distance defined as

$$\ell_\alpha(\mathbf{u},\mathbf{v}) = (\sum_{i=1}^{d} |v_i - u_i|^{\alpha})^{\frac{1}{\alpha}} \ , \tag{3}$$

where $\alpha \geq 1$. When $\alpha = 1$, we have $\ell_1(\mathbf{u},\mathbf{v}) = \sum_{i=1}^{d} |v_i - u_i|$. The ℓ_1 distance allows a user to evaluate whether the overall absolute difference between his and another user's profiles is above a threshold chosen by himself.

Protocol 1 is designed to offer level-I privacy from Alice's viewpoint with regard to Bob. It is a nontrivial adaptation from the protocol in [14] with significantly lower computation overhead to be shown shortly. Our basic idea is to first convert $\ell_1(\mathbf{u},\mathbf{v})$ into the ℓ_2 distance between the unary representations of $\mathbf{u}$ and $\mathbf{v}$ and then compute the ℓ_2 distance using a secure dot-product protocol.

In particular, for all $x \in [0, \gamma-1]$, we define a binary vector $h(x) = \langle x_1, \ldots, x_{\gamma-1}\rangle$, where x_i is equal to one for $1 \leq i \leq x$ and zero for $x < i \leq \gamma - 1$. We also abuse the notation by defining another binary vector $\hat{\mathbf{u}} = h(\mathbf{u}) = \langle h(u_1), \ldots, h(u_d)\rangle = \langle \hat{u}_1, \ldots, \hat{u}_{(\gamma-1)d}\rangle$ and $\hat{\mathbf{v}} = h(\mathbf{v}) = \langle h(v_1), \ldots, h(v_d)\rangle = \langle \hat{v}_1, \ldots, \hat{v}_{(\gamma-1)d}\rangle$. It follows that

$$\begin{aligned} \ell_1(\mathbf{u},\mathbf{v}) &= \sum_{i=1}^{d} |u_i - v_i| \\ &= \sum_{i=1}^{(\gamma-1)d} |\hat{u}_i - \hat{v}_i| \\ &= \sum_{i=1}^{(\gamma-1)d} |\hat{u}_i - \hat{v}_i|^2 = \ell_2^2(\hat{\mathbf{u}},\hat{\mathbf{v}}) \ . \end{aligned} \tag{4}$$

The correctness of the above equation is straightforward. We can further note that

$$\begin{aligned} \ell_2^2(\hat{\mathbf{u}},\hat{\mathbf{v}}) &= \sum_{i=1}^{(\gamma-1)d} |\hat{u}_i - \hat{v}_i|^2 \\ &= \sum_{i=1}^{(\gamma-1)d} \hat{u}_i^2 - 2\sum_{i=1}^{(\gamma-1)d} \hat{u}_i\hat{v}_i + \sum_{i=1}^{(\gamma-1)d} \hat{v}_i^2 \\ &= \sum_{i=1}^{(\gamma-1)d} \hat{u}_i^2 - 2\hat{\mathbf{u}}\cdot\hat{\mathbf{v}} + \sum_{i=1}^{(\gamma-1)d} \hat{v}_i^2 \ . \end{aligned} \tag{5}$$

Since Alice and Bob know $\sum_{i=1}^{(\gamma-1)d} \hat{u}_i^2$ and $\sum_{i=1}^{(\gamma-1)d} \hat{v}_i^2$, respectively, we just need a secure dot-product protocol for Bob to compute $\hat{\mathbf{u}}\cdot\hat{\mathbf{v}}$ without knowing Alice's profile $\mathbf{u}$ or disclosing his profile $\mathbf{v}$ to Alice. Subsequently, Bob can return $-2\hat{\mathbf{u}}\cdot\hat{\mathbf{v}} + \sum_{i=1}^{(\gamma-1)d} \hat{v}_i^2$ for Alice to finish computing $\ell_2^2(\hat{\mathbf{u}},\hat{\mathbf{v}})$ and thus $\ell_1(\mathbf{u},\mathbf{v})$.

Protocol Details

The detailed operations of Protocol 1 are shown in Fig. 2.

1. Alice does the following in sequence.
   a. Construct a vector $\hat{\mathbf{u}} = h(\mathbf{u}) = (h(u_1), \ldots, h(u_d)) = (\hat{u}_1, \ldots, \hat{u}_{(\gamma-1)d})$, where $\hat{u}_j$ is equal to one for every $j \in \mathcal{J}_{\mathbf{u}} = \{j | (i-1)(\gamma-1) < j \leq (i-1)(\gamma-1) + u_i, 1 \leq i \leq d\}$ and zero otherwise.
   b. Choose a distinct $r_j \in \mathbb{Z}_N$ and compute $\mathsf{E}(\hat{u}_j, r_j)$ for every $j \in [1, (\gamma-1)d]$ using her public key $\langle N, g\rangle$.
   c. Send $\{\mathsf{E}(\hat{u}_j, r_j)\}_{j=1}^{(\gamma-1)d}$ and her public key to Bob.
2. Bob does the following after receiving Alice's message.
   a. Construct a vector $\hat{\mathbf{v}} = h(\mathbf{v}) = (h(v_1), \ldots, h(v_d)) = (\hat{v}_1, \ldots, \hat{v}_{(\gamma-1)d})$, where $\hat{v}_j$ is equal to one for every $j \in \mathcal{J}_{\mathbf{v}} = \{j | (i-1)(\gamma-1) < j \leq (i-1)(\gamma-1) + v_i, 1 \leq i \leq d\}$ and zero otherwise.
   b. Compute

$$\begin{aligned} \mathsf{E}(\hat{\mathbf{u}}\cdot\hat{\mathbf{v}}, s) &= \mathsf{E}(\sum_{j\in\mathcal{J}_{\mathbf{v}}} \hat{u}_j, \prod_{j\in\mathcal{J}_{\mathbf{v}}} r_j) \\ &= \prod_{j\in\mathcal{J}_{\mathbf{v}}} \mathsf{E}(\hat{u}_j, r_j) \mod N^2 \ , \end{aligned} \tag{6}$$

   where $s = \prod_{j\in\mathcal{J}_{\mathbf{v}}} r_j$, the first equality sign is due to the constructions of $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$, and the second is due to the homomorphic property of the Paillier cryptosystem.
   c. Compute

$$\mathsf{E}((N-2)\hat{\mathbf{u}}\cdot\hat{\mathbf{v}}, s') = \mathsf{E}^{N-2}(\hat{\mathbf{u}}\cdot\hat{\mathbf{v}}, s) \mod N^2 \ ,$$

   where $s' = s^{N-2} \mod N$. This equation holds again due to the homomorphic property of the Paillier cryptosystem.
   d. Encrypt $\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2$ with Alice's public key $\langle N, g\rangle$ and a random $r \in \mathbb{Z}_N$ by computing

$$\mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2, r) = g^{\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2} \cdot r^N \mod N^2 \ .$$

   e. Compute

$$\begin{aligned} &\mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}}\cdot\hat{\mathbf{v}}, rs') \\ &= \mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 + (N-2)\hat{\mathbf{u}}\cdot\hat{\mathbf{v}}, rs') \\ &= \mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2, r) \cdot \mathsf{E}((N-2)\hat{\mathbf{u}}\cdot\hat{\mathbf{v}}, s') \mod N^2, \end{aligned} \tag{7}$$

   and send it back to Alice. Note that the first equality sign is because $\hat{v}_j^2 - 2\hat{\mathbf{u}}\cdot\hat{\mathbf{v}} = \hat{v}_j^2 + (N-2)\hat{\mathbf{u}}\cdot\hat{\mathbf{v}}$

Alice

$\hat{\mathbf{u}} = \langle \hat{u}_1, \ldots, \hat{u}_{(\gamma-1)d} \rangle$

Bob

$\hat{\mathbf{v}} = \langle \hat{v}_1, \ldots, \hat{v}_{(\gamma-1)d} \rangle$

Compute $\mathsf{E}(\hat{u}_j, r_j), \forall j \in [1, (\gamma-1)d]$

$\xrightarrow{\langle N, g \rangle, \mathsf{E}(\hat{u}_1, r_1), \ldots, \mathsf{E}(\hat{u}_{(\gamma-1)d}, r_{(\gamma-1)d})}$

Compute $\mathsf{E}(\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, s) = \prod_{j \in \mathcal{J}_{\mathbf{v}}} \mathsf{E}(\hat{u}_j, r_j) \mod N^2$

Compute $\mathsf{E}(\sum_{j=1}^{(\gamma-1)d} \hat{v}_j^2, r)$ with a random r

Compute $\mathsf{E}(\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, s)^{N-2} \cdot \mathsf{E}(\sum_{j=1}^{(\gamma-1)d} \hat{v}_j^2, r) \mod N^2$

$\xleftarrow{\mathsf{E}(\sum_{j=1}^{(\gamma-1)d} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, rs')}$

Obtain $\sum_{j=1}^{(\gamma-1)d} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}$ after decryption

Compute $\ell_1(\mathbf{u}, \mathbf{v}) = \sum_{j=1}^{(\gamma-1)d} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}} + \sum_{j=1}^{(\gamma-1)d} \hat{u}_j^2$

Fig. 2. Protocol 1 with Privacy Level I: $f(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{d} |u_i - v_i|$.

$\mod N$, and that the second is again due to the homomorphic property of the Paillier cryptosystem.

3. Alice decrypts $\mathsf{E}(\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}, rs')$ using her private key to get $\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}$ and finally computes

$$\ell_1(\mathbf{u}, \mathbf{v}) = \sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}} + \sum_{j=1}^{d(\gamma-1)} \hat{u}_j^2. \quad (8)$$

Protocol Analysis

We now analyze the privacy provision of Protocol 1 and the related computation and communication overhead.

***Theorem** 1: Protocol 1 ensures level-I privacy if the Paillier cryptosystem is semantically secure and a personal profile is a vector of dimension $d \geq 2$.*

Proof: Bob receives and operates only on ciphertexts $\{\mathsf{E}(\hat{u}_1, r_1)\}_{j=1}^{(\gamma-1)d}$ and does not know Alice's private key. Since the Paillier cryptosystem is semantically secure, computationally bounded Bob cannot decrypt the ciphertexts to learn anything about Alice's profile $\mathbf{u}$. As to Alice, she only get $\sum_{j=1}^{d(\gamma-1)} \hat{v}_j^2 - 2\hat{\mathbf{u}} \cdot \hat{\mathbf{v}}$. If she wants to find out Bob's profile $\mathbf{v}$, she must solve an equation with d unknowns, which is infeasible for $d \geq 2$. Therefore, Alice knows nothing about $\mathbf{v}$ other than the result $\ell_1(\mathbf{u}, \mathbf{v})$. ■

The computation overhead incurred by Protocol 1 is mainly related to modular exponentiations and multiplications. In particular, Alice needs to perform $(\gamma-1)d$ Paillier encryptions in Step 1.a, each costing two 1024-bit exponentiations and one 2048-bit multiplication according to Eq. (1). Note that Alice can preselect many random numbers and precompute the corresponding ciphertexts in an offline manner to reduce the online matching time.[1] In addition, Alice needs to perform one Paillier decryption in Step 3, which is essentially a 2048-bit exponentiation. As for Bob, he needs to perform $\sum_{i=1}^{d} v_i - 1$ 2048-bit multiplications in Step 2.b, one 2048-bit exponentiation in Step 2.c, two 1024-bit exponentiations and one 2048-bit multiplication (i.e., one Paillier encryption) in Step 2.d, and one 2048-bit multiplication in Step 2.e. Considering Alice and Bob together, we can approximate the online computation cost of Protocol 1 to be $\sum_{i=1}^{d} v_i + 1$ 2048-bit multiplications, two 2048-bit exponentiations, and two 1024-bit exponentiations. In contrast, a direct application of the secure dot-protocol in [14] will require Bob to perform totally $(\gamma-1)d-1$ more 2048-bit exponentiations in Steps 2.b and 2.c.

[1] Alice can even do such offline computations on her regular computer and then synchronize the results to her mobile device.

The communication overhead incurred by Protocol 1 involves Alice sending her public key $\langle N, g \rangle$ and $(\gamma-1)d$ ciphertexts in Step 1.c and Bob returning one ciphertext in Step 2.e. Since a public key and a ciphertext are of 1184 and 2048 bits, respectively, the total net communication cost of Protocol 1 is of $2048(\gamma-1)d + 3232$ bits without considering message headers and other fields.

B. Protocol 2 for Level-II Privacy

We now introduce Protocol 2 which can satisfy level-II privacy. In contrast to Protocol 1 working only for the ℓ_1 distance, Protocol 2 can apply to a family of additively separable matching metrics and also hide the matching metric chosen by one user from the other. The secrecy of a user's selected matching metric can help prevent an attacker from generating better tailored profiles to deceive the victim user into a successful matching.

To illustrate Protocol 2, we first introduce then definition of *additively separable* functions as follows.

***Definition** 4: A function $f(\mathbf{u}, \mathbf{v})$ is additively separable if it can be written as $f(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{d} f_i(u_i, v_i)$ for some functions $f_1(\cdot), \ldots, f_d(\cdot)$.*

Many common matching metrics are additively separable. For example, the ℓ_1 distance can be written as $\ell_1(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{d} |u_i - v_i|$, the dot product is $\mathbf{u} \cdot \mathbf{v} = \sum_{i=1}^{d} u_i v_i$, and the ℓ_α norm is $\ell_\alpha^\alpha = \sum_{i=1}^{d} |u_i - v_i|^\alpha$. In addition, assuming that Alice assigns a weight w_i to attribute i, we can define the weighted ℓ_1 distance as $\sum_{i=1}^{d} w_i |u_i - v_i|$ which is also additively separable.

By supporting general additively separable function, Protocol 2 enables the user to control the level of differentiation in private matching. For example, although different pairs of personal profiles could have the same ℓ_1 distances, such ambiguity can be avoided by using weighted ℓ_1 distance and assigning different weights to different individual attributes.

Protocol 2 works by first converting any additively separable function into a dot-product computation. In particular, given

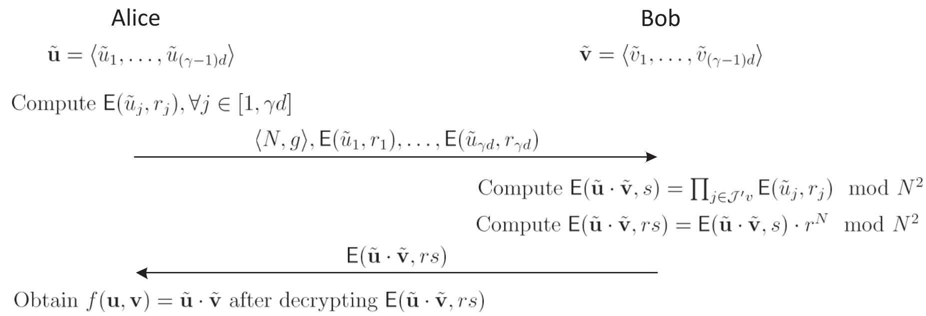


Fig. 3. Protocol 2 with Privacy Level II: $f(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{d} f_i(u_i, v_i)$.

an additively separable similarity function f of interest, Alice constructs a vector $\tilde{\mathbf{u}} = \langle \tilde{u}_1, \ldots, \tilde{u}_{\gamma d} \rangle$, where $\tilde{u}_j = f_i(u_i, k)$, $i = \lfloor (j-1)/\gamma \rfloor + 1$, and $k = (j-1) \mod \gamma$, for all $j \in [1, \gamma d]$. Assume that Bob also relies on his profile $\mathbf{v}$ to construct a binary vector $\tilde{\mathbf{v}} = (\tilde{v}_1, \ldots, \tilde{v}_{\gamma d})$, where the jth bit $\tilde{v}_j$ equals one for all $j \in \mathcal{J}'_{\mathbf{v}} = \{j | j = (i-1)\gamma + v_i + 1, 1 \leq i \leq d\}$ and zero otherwise. It follows that $\tilde{u}_j \tilde{v}_j = \tilde{u}_j = f_i(u_i, v_i)$ for all $j \in \mathcal{J}'_{\mathbf{v}}$ and zero otherwise. We then can easily obtain the following result.

$$\begin{aligned} f(\mathbf{u}, \mathbf{v}) &= \sum_{i=1}^{d} f_i(u_i, v_i) \\ &= \sum_{j \in \mathcal{J}'_{\mathbf{v}}} \tilde{u}_j \\ &= \sum_{j=1}^{\gamma d} \tilde{u}_j \tilde{v}_j = \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} \end{aligned} \tag{9}$$

So we can let Alice run a secure dot-protocol protocol with Bob to obtain $\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} = f(\mathbf{u}, \mathbf{v})$ without disclosing $\mathbf{u}$ or f to Bob.

Protocol Details

The detailed operations of Protocol 2 are shown in Fig. 3 and as follows.

1. Alice first constructs a vector $\tilde{\mathbf{u}}$ as discussed above and then chooses a distinct random $r_j \in \mathbb{Z}_N$ to compute $\mathsf{E}(\tilde{u}_j, r_j)$ for all $j \in [1, \gamma d]$ using her public key. Finally, she sends $\{\mathsf{E}(\tilde{u}_j, r_j)\}_{j=1}^{\gamma d}$ and her public key $\langle N, g \rangle$ to Bob.
2. Bob constructs a vector $\tilde{\mathbf{v}}$ as described above after receiving Alice's message. He then computes
$$\begin{aligned} \mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, s) &= \mathsf{E}(\sum_{j \in \mathcal{J}'_{\mathbf{v}}} \tilde{u}_j, \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j) \\ &= \prod_{j \in \mathcal{J}'_{\mathbf{v}}} \mathsf{E}(\tilde{u}_j, r_j) \mod N^2 , \end{aligned} \tag{10}$$
where $s = \prod_{j \in \mathcal{J}'_{\mathbf{v}}} r_j$. The equation holds due to Eq. (9) and the homomorphic property of the Paillier cryptosystem. Next, he selects a random number $r \in \mathbb{Z}_N$ to compute
$$\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, rs) = \mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, s) \cdot r^N \mod N^2 , \tag{11}$$
which holds due to the self-blinding property of the Paillier cryptosystem introduced in Section II-C. Finally, Bob returns $\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, rs)$ to Alice.
3. Alice uses her private key λ to decrypt $\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, rs)$ and finally get $\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}$, i.e., $f(\mathbf{u}, \mathbf{v})$.

Note that it is necessary for Bob to perform one more encryption in Step.2 using a random number r unknown to Alice. Otherwise, Alice may be able to easily infer Bob's profile $\mathbf{v}$ without decryption. In particular, suppose that Bob directly sends $\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, s)$ to Alice without self-blinding it with r as in Eq. (11). Alice can try all possible $\mathcal{J}'_{\mathbf{v}}$ to find the one satisfying Eq. (10) whereby to deduce $\tilde{\mathbf{v}}$ and $\mathbf{v}$ without actually decrypting $\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, s)$. The total number of all possible cases is λ^d, which may not be large for small d.

Protocol Analysis

We now analyze the privacy provision of Protocol 2 and the related computation and communication overhead.

***Theorem** 2: Protocol 2 ensures level-II privacy if the Paillier cryptosystem is semantically secure and a personal profile is a vector of dimension $d \geq 2$.*

Proof: The proof is similar to that of Theorem 1 except that Bob does not know the matching metric f employed by Alice. It is omitted here for lack of space. ■

The computation overhead incurred by Protocol 2 also mainly relates to modular exponentiations and multiplications. In particular, Alice needs to perform γd Paillier encryptions in Step 1, each requiring two 1024-bit exponentiations and one 2048-bit multiplication. As in Protocol 1, Alice can do these encryptions beforehand in an offline manner. In addition, Alice needs to perform one Paillier decryption in Step 3, corresponding to one 2048-bit exponentiation. Moreover, Bob needs to perform $d - 1 = |\mathcal{J}'_{\mathbf{v}}| - 1$ 2048-bit multiplications in Eq. (10) plus one 1024-bit exponentiation and one 2048-bit multiplication in Eq. (11). In summary, the total online computation overhead of Protocol 2 can be approximated by d 2048-bit multiplications, one 2048-bit exponentiation, and one 1024-bit exponentiation.

The communication overhead incurred by Protocol 2 involves Alice sending her public key $\langle N, g \rangle$ and γd ciphertexts in Step 1 and Bob returning one ciphertext in Step 2. Similar to that of Protocol 1, the total net communication cost of Protocol 2 can be computed as $2048(\gamma d + 1) + 1184$ bits without considering message headers and other fields.

Alice — $\tilde{\mathbf{u}} = \langle \tilde{u}_1, \ldots, \tilde{u}_{(\gamma-1)d} \rangle, \tau_A$

Bob — $\tilde{\mathbf{v}} = \langle \tilde{v}_1, \ldots, \tilde{v}_{(\gamma-1)d} \rangle$

Alice: Compute $\mathsf{E}(\tilde{u}_j, r_j), \forall j \in [1, \gamma d], \mathsf{E}(\tau_A, r_{\tau_A})$

Alice → Bob: $\langle N, g \rangle, \mathsf{E}(\tilde{u}_1, r_1), \ldots, \mathsf{E}(\tilde{u}_{\gamma d}, r_{\gamma d}), \mathsf{E}(\tau_A, r_{\tau_A})$

Bob: Choose $r_1', r_2', \delta > \delta_1 > \delta_2 \geq 0$

Compute $\mathsf{E}(\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1, s_1) = \mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, \prod_{j \in \mathcal{J}_{\mathbf{v}}'} r_j)^{\delta} \cdot \mathsf{E}(\delta_1, r_1') \mod N^2$

Compute $\mathsf{E}(\delta \tau_A + \delta_2, s_2) = \mathsf{E}(\tau_A, r_{\tau_A})^{\delta} \cdot \mathsf{E}(\delta_2, r_2') \mod N^2$

Bob → Alice: $\mathsf{E}(\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1, s_1), \mathsf{E}(\delta \tau_A + \delta_2, s_2)$

Alice: Obtain $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1$ and $\delta \tau_A + \delta_2$ after decryption

Output $f(\mathbf{u}, \mathbf{v}) < \tau_A$ if $\delta \cdot f(\mathbf{u}, \mathbf{v}) + \delta_1 < \delta \tau_A + \delta_2$

Fig. 4. Protocol 3 with Privacy Level III: $f(\mathbf{u}, \mathbf{v}) > \tau_A$.

C. Protocol 3 for Level-III Privacy

Protocol 3 is designed to offer level-III privacy. In contrast to Protocol 2, it only lets Alice know whether $f(\mathbf{u}, \mathbf{v})$ is smaller than her personal threshold τ_A, while hiding $f(\mathbf{u}, \mathbf{v})$ from her. Protocol 3 is desirable if Bob does not want Alice to know the actual similarity score $f(\mathbf{u}, \mathbf{v})$.

Protocol 3 is based on a special trick. In particular, assuming that there are three arbitrary integers δ, δ_1, and δ_2 such that $\delta > \delta_1 > \delta_2 \geq 0$, we have $0 < (\delta_1 - \delta_2)/\delta < 1$. Since we assume $f(\mathbf{u}, \mathbf{v})$ and τ_A both to be integers, $f(\mathbf{u}, \mathbf{v}) < \tau_A$ is equivalent to $f(\mathbf{u}, \mathbf{v}) + (\delta_1 - \delta_2)/\delta < \tau_A$ and thus $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1 < \delta \tau_A + \delta_2$. On the other hand, if $f(\mathbf{u}, \mathbf{v}) \geq \tau_A$, we would have $f(\mathbf{u}, \mathbf{v}) + (\delta_1 - \delta_2)/\delta > \tau_A$. According to this observation, Bob can choose random δ, δ_1, and δ_2 unknown to Alice and then send encrypted $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1$ and $\delta \tau_A + \delta_2$ to Alice. After decrypting the ciphertexts, Alice can check whether $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1$ is smaller than $\delta \tau_A + \delta_2$ to learn whether $f(\mathbf{u}, \mathbf{v}) < \tau_A$.

Protocol Details

The detailed operations of Protocol 3 are shown in Fig. 4 and as follows.

1. Alice first constructs a vector $\tilde{\mathbf{u}}$ as in Step 1 of Protocol 2. She then chooses a distinct random $r_j \in \mathbb{Z}_N$ to compute $\mathsf{E}(\tilde{u}_j, r_j)$ for all $j \in [1, \gamma d]$ and also another distinct random r_{τ_A} to compute $\mathsf{E}(\tau_A, r_{\tau_A})$. Finally, she sends $\{\mathsf{E}(\tilde{u}_j, r_j)\}_{j=1}^{\gamma d}$, $\mathsf{E}(\tau_A, r_{\tau_A})$, and her public key $\langle N, g \rangle$ to Bob.
2. Bob first constructs a binary vector $\tilde{\mathbf{v}}$ whereby to compute $\mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, \prod_{j \in \mathcal{J}_{\mathbf{v}}'} r_j)$ (i.e., $\mathsf{E}(f(\mathbf{u}, \mathbf{v}), \prod_{j \in \mathcal{J}_{\mathbf{v}}'} r_j)$) as in Step 2 of Protocol 2. He then randomly choose $r_1', r_2', \delta, \delta_1, \delta_2 \in \mathbb{Z}_N$ such that $\delta > \delta_1 > \delta_2$ to compute
$$\mathsf{E}(\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1, s_1) = \mathsf{E}(\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}}, \prod_{j \in \mathcal{J}_{\mathbf{v}}'} r_j)^{\delta} \cdot \mathsf{E}(\delta_1, r_1') \mod N^2 \quad (12)$$
and
$$\mathsf{E}(\delta \tau_A + \delta_2, s_2) = \mathsf{E}(\tau_A, r_{\tau_A})^{\delta} \cdot \mathsf{E}(\delta_2, r_2') \mod N^2, \quad (13)$$
where $s_1 = r_1'(\prod_{j \in \mathcal{J}_{\mathbf{v}}'} r_j)^{\delta} \mod N$ and $s_2 = r_2' r_{\tau_A}$. Both equations hold due to the homomorphic property of the Paillier cryptosystem. Finally, Bob returns $\mathsf{E}(\delta \tilde{\mathbf{u}} \tilde{\mathbf{v}} + \delta_1, s_1)$ and $\mathsf{E}(\delta \tau_A + \delta_2, s_2)$ to Alice.
3. Alice decrypts the ciphertexts to get $\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1$ and $\delta \tau_A + \delta_2$. If the former is smaller than the latter, Alice knows $f(\mathbf{u}, \mathbf{v}) < \tau_A$. Otherwise, she knows $f(\mathbf{u}, \mathbf{v}) \geq \tau_A$.

Protocol Analysis

We now analyze the privacy provision of Protocol 3 and the related computation and communication overhead.

***Theorem** 3: Protocol 3 ensures level-III privacy if the Paillier cryptosystem is semantically secure and a personal profile is a vector of dimension $d \geq 2$.*

Proof: The proof is similar to that of Theorem 2 except the additional points that Bob does not know Alice's threshold τ_A and that Alice does know the comparison result $f(\mathbf{u}, \mathbf{v})$. It is thus omitted here for lack of space. ■

The computation overhead incurred by Protocol 3 also mainly relates to modular exponentiations and multiplications. In particular, Alice needs to perform $\gamma d+1$ Paillier encryptions in Step 1, each requiring two 1024-bit exponentiations and one 2048-bit multiplication. As in Protocol 2, Alice can do these encryptions beforehand in an offline manner. In addition, Alice needs to perform two Paillier decryptions in Step 3, each corresponding to one 2048-bit exponentiation. Moreover, Bob needs to perform $d - 1 = |\mathcal{J}_{\mathbf{v}}'| - 1$ 2048-bit multiplications in Eq. (10) plus one 1024-bit exponentiation and one 2048-bit multiplication in Eq. (11). Furthermore, Bob needs to perform one 2048-bit exponentiation, one Paillier encryption, and one 2048-bit multiplication in each of Eqs. (12) and (13). In summary, the total online computation cost of Protocol 3 can be approximated by $d + 3$ 2048-bit multiplications, four 2048-bit exponentiation, and four 1024-bit exponentiations.

The communication overhead incurred by Protocol 3 involves Alice sending her public key $\langle N, g \rangle$ and $\gamma d+1$ ciphertexts in Step 1 and Bob returning two ciphertexts in Step 2. Similar to that of Protocol 2, the total net communication cost of Protocol 3 can be computed as $2048(\gamma d + 3) + 1184$ bits without considering message headers and other fields.

D. Discussion of Malicious Active Attacks

We have thus far only considered passive attacks as in previous work [2]–[4] on private profile matching. An active adversary may launch attacks to infer the target user's personal profile and/or disrupt PMSN operations. Now we discuss some possible active attacks on our protocols and their impact on

Alice	Bob
$\tau_{\max}$, $\tilde{\mathbf{u}} = \langle \tilde{u}_1, \ldots, \tilde{u}_{(\gamma-1)d} \rangle$, where $\tilde{u}_j = \phi_i(u_i, k, \tau_{\max})$	$\tilde{\mathbf{v}} = \langle \tilde{v}_1, \ldots, \tilde{v}_{(\gamma-1)d} \rangle$
Compute $\mathsf{E}(\tilde{u}_j, r_j), \forall j \in [1, \gamma d]$, $\mathsf{E}(\tau_A, r_{\max})$

$$\xrightarrow{\langle N, g\rangle, \mathsf{E}(\tilde{u}_1, r_1), \ldots, \mathsf{E}(\tilde{u}_{\gamma d}, r_{\gamma d}), \mathsf{E}(\tau_A, r_{\tau_A})}$$

Choose $r'_1, r'_2, \delta > \delta_1 > \delta_2 \geq 0$
Compute $\mathsf{E}(\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1, s_1)$
Compute $\mathsf{E}(\delta d + \delta_2, s_2)$

$$\xleftarrow{\mathsf{E}(\delta \tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1, s_1), \mathsf{E}(\delta d + \delta_2, s_2)}$$

Obtain $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1$ and $\delta d + \delta_2$ after decryption
Output $\ell_{\max}(\mathbf{u}, \mathbf{v}) > \tau_{\max}$ if $\delta f(\mathbf{u}, \mathbf{v}) + \delta_1 < \delta d + \delta_2$

Fig. 5. Protocol 4 with Privacy Level III: $\ell_{\max}(\mathbf{u}, \mathbf{v}) < \tau$.

profile matching. Due to tight space constraints, we only discuss possible countermeasures here and leave the detailed investigation to a separate paper.

Manipulating protocol output: Assuming that Bob is malicious, he may manipulate the protocol output by using an arbitrary profile and/or not faithfully following the protocol operations (e.g., by changing intermediate computation results). It is fundamentally difficult to defend against this attack without involving a trusted third party as in [2], [6]. In particular, Alice cannot tell whether the protocol output is caused by Bob's misbehavior or they indeed having similar profiles. Our protocols, however, can guarantee one of the three privacy levels for Alice against Bob. In addition, since Bob cannot infer Alice's personal profile after the protocol execution, he cannot purposefully control the protocol output.

Repeatedly matching with different profiles: Assuming that Bob is malicious, he may also repeatedly conduct private-matching protocols with Alice using different profiles, aiming at inferring additional information from each protocol output. Consider Protocol 1 as an example, each protocol execution essentially allows Bob to learn one linear equation about Alice's profile. If Bob performs private matching with Bob for sufficient times with a different profile each time, Bob can eventually learn Alice's profile. To defend against such attacks in practice, Alice can limit the number of private-matching requests from the same user.

Denial-of-service (DoS) attack: As almost all wireless protocols involving message exchanges among involved parties such as [2]–[4], our protocols are vulnerable to DoS attacks in which an attacker keeps sending or replying to chatting requests without finishing private matching with good users in order to consume their device resources. The DoS attack can be mitigated by incorporating message puzzles [16] into protocol design, for which we refer readers to [17] for more details.

IV. EXTENSION: MAX-DISTANCE MATCHING

In this section, we present another private-matching protocol based on the MAX distance. Given two personal profiles $\mathbf{u}$ and $\mathbf{v}$, the MAX distance between them is defined as follows.

$$\ell_{\max}(\mathbf{u}, \mathbf{v}) = \max\{|v_1 - u_1|, \ldots, |v_d - u_d|\} \tag{14}$$

Protocols 1 to 3 all enable a user to check whether the overall absolute difference between her and another user's profiles is below a personally chosen threshold. In contrast, Protocol 4 allows the user to check whether the maximum attribute-wise absolute difference does not exceed her personal threshold.

At the first glance, $\ell_{\max}(\mathbf{u}, \mathbf{v})$ is not additively separable, so it cannot be computed using Protocol 2 or 3. In what follows, we first show how to convert $\ell_{\max}(\mathbf{u}, \mathbf{v})$ into an additively separable function based on a concept called *similarity matching* and then present the protocol details and analysis.

A. From MAX Distance to Additively Separable Function

The conversion from $\ell_{\max}(\mathbf{u}, \mathbf{v})$ to an additively separable function relies on similarity matching defined as follows.

Definition 5: Given two personal profiles $\mathbf{u} = \langle u_1, \ldots, u_d \rangle$ and $\mathbf{v} = \langle v_1, \ldots, v_d \rangle$, their i attributes are considered **similar** if $|u_i - v_i| \leq \tau$ for some threshold τ.

Definition 6: The **similarity score** of $\mathbf{u}$ and $\mathbf{v}$, denoted by $\Phi(\mathbf{u}, \mathbf{v}, \tau)$, is defined as the number of similar attributes, i.e.,

$$\Phi(\mathbf{u}, \mathbf{v}, \tau) = \sum_{i=1}^{d} \phi(u_i, v_i, \tau) ,$$

where

$$\phi(u_i, v_i, \tau) = \begin{cases} 1 & \text{if } |u_i - v_i| \leq \tau, \\ 0 & \text{otherwise} . \end{cases}$$

The similarity score has three essential properties. First, it is additively separable, implying that Alice can run Protocol 2 with Bob to compute $\Phi(\mathbf{u}, \mathbf{v}, \tau)$ or Protocol 3 to check whether $\Phi(\mathbf{u}, \mathbf{v}, \tau) < \tau_A$. Second, it is directly affected by the value of τ. In particular, the larger τ, the higher $\Phi(\mathbf{u}, \mathbf{v}, \tau)$, and vice versa. Last, it relates to $\ell_{\max}(\mathbf{u}, \mathbf{v})$ based on the following theorem.

Theorem 4: *For all $\tau \geq \ell_{\max}(\mathbf{u}, \mathbf{v})$, we have $\Phi(\mathbf{u}, \mathbf{v}, \tau) = d$; likewise, for all $\tau < \ell_{\max}(\mathbf{u}, \mathbf{v})$, we have $\Phi(\mathbf{u}, \mathbf{v}, \tau) < d$.*

Proof: By the definition of the MAX distance, we have $|u_i - v_i| \leq \ell_{\max}(\mathbf{u}, \mathbf{v})$ for all $1 \leq i \leq d$. It follows that $\phi(u_i, v_i, \tau) = 1$ for all $1 \leq i \leq d$ if $\tau \geq \ell_{\max}(\mathbf{u}, \mathbf{v})$. Therefore, we have therefore have $\mathbf{s}(\mathbf{u}, \mathbf{v}, \tau) = d$ for all $\tau \geq \ell_{\max}(\mathbf{u}, \mathbf{v})$. Similarly, by the definition of MAX distance, there exists k such that $|u_k - v_k| = \ell_{\max}(\mathbf{u}, \mathbf{v})$. It follows that $\phi(u_k, v_k, \tau) = 0$, so we have $\Phi(\mathbf{u}, \mathbf{v}, \tau) < d$ for all $\tau < \ell_{\max}(\mathbf{u}, \mathbf{v})$. ■

B. Protocol 4: MAX-Distance Matching for Level-III Privacy

Protocol 4 depends on Protocol 3 for level-III privacy. Let $\tau_{\max}$ to denote Alice's MAX-distance threshold kept secret from Bob. According to Theorem 4, checking whether $\ell_{\max}(\mathbf{u}, \mathbf{v}) < \tau_{\max}$ is equivalent to checking whether $\Phi(\mathbf{u}, \mathbf{v}, \tau_{\max}) = d$.

Protocol Details

The detailed operations of Protocol 3 are shown in Fig. 5 and as follows.

1. Alice first constructs a vector $\tilde{\mathbf{u}} = \langle \tilde{u}_1, \ldots, \tilde{u}_{\gamma d} \rangle$, where $\tilde{u}_j = \phi_i(u_i, k, \tau_{\max})$, $i = \lfloor j/\gamma \rfloor + 1$, and $k = (j-1) \mod \gamma$ for all $j \in [1, \gamma d]$. He then chooses a random $r_{\max} \in \mathbb{Z}_N$ to compute $\mathsf{E}(d, r_{\max})$ and a distinct random $r_j \in \mathbb{Z}_N$ to compute $\mathsf{E}(\tilde{u}_j, r_j)$ for all $j \in [1, \gamma d]$. Finally, she sends $\{\mathsf{E}(\tilde{u}_j, r_j)\}_{j=1}^{\gamma d}$, $\mathsf{E}(d, r_{\tau_{\max}})$, and her public key to Bob.
2. Bob performs almost the same operations as in Step 2 of Protocol 2 (except replacing r_A by $r_{\max}$) and returns $\mathsf{E}(\delta\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1, s_1)$ and $\mathsf{E}(\delta d + \delta_2, r_2' r_{\tau_{\max}})$ to Alice. As in Protocol 2, we have $\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} = \Phi(\mathbf{u}, \mathbf{v}, \tau_{\max})$.
3. Alice does the same as in Step 3 of Protocol 3 to check whether $\delta\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} + \delta_1 < \delta d + \delta_2$. If so, she learns $\tilde{\mathbf{u}} \cdot \tilde{\mathbf{v}} < d$ (i.e., $\Phi(\mathbf{u}, \mathbf{v}, \tau_{\max}) < d$) and thus $\ell_{\max}(\mathbf{u}, \mathbf{v}) > \tau_{\max}$. Otherwise, Alice knows $\ell_{\max}(\mathbf{u}, \mathbf{v}) \leq \tau_{\max}$.

Since Protocol 4 is a special case of Protocol 3 and thus can also ensure level-III privacy with the same communication and computation overhead as that of Protocol 3.

V. Performance Evaluation

In this section, we evaluate the communication and computation overhead as well as overall execution time of our protocols. Since the previous work [2]–[5] is not applicable to fine-grained private matching, we only compare our work with RSV, which refers to the ℓ_1 distance protocol in [14] and can satisfy level-I privacy. We are not aware of any existing work that can offer level-III privacy as our Protocol 3. Due to space limitations, we omit the straightforward derivation process for the communication and computation costs of RSV and refer interested readers to [14] for details.

Table I summarizes the theoretical performance of Protocols 1$\sim$4 and RSV, where $\mathsf{mul}_1, \mathsf{mul}_2, \mathsf{exp}_1$, and exp_2 denote one 1024-bit multiplication, 2048-bit multiplication, 1024-bit exponentiation, and 2048-bit exponentiation, respectively. All our protocols obviously incur much lower online computation overhead with similar communication overhead.

A. Implementation

We implement our four protocols and RSV on LG P-970 smartphones, which has a 1GHz Cortex-A8 processor, 512 MB RAM, Android v2.2 Operating System, a 802.11 b/g/n WiFi interface, and Bluetooth v2.1 with Enhanced Data Rate (EDR). As in [5], we use a publicly available Java implementation of Paillier cryptosystem [15]. The whole PMSN application consists of 5000+ lines of Java code, in which our four protocols share the majority of the codes. Since Android platform currently does not support WiFi ad-hoc mode, we let two smartphones communicate with each other through Bluetooth. In our experiments, we are only able to achieve a transmission rate of approximately 800 kb/s, although Bluetooth v2.1 with EDR is expected to operate at a transmission rate of 2.1 Mb/s. If better Bluetooth implementations are available, the execution time of our protocols can be significantly reduced.

In our implementation, assuming that Alice initiates the matching protocol with Bob, each protocol consists of five main steps as follows.

1) Alice prepares the message through offline computation, e.g., generating a number of ciphertexts according to our protocol specifications;
2) Alice sends the message to indicate the start of the protocol;
3) Bob receives and buffers the message;
4) Once the transmission completes, Bob computes the intermediate result according to our protocol specifications, and sends it back to Alice;
5) On receiving the intermediate result, Alice computes the final matching result.

We use custom message headers in the application layer to distinguish these messages.

B. Experimental Results

We first measure the computation time of different basic operations of Paillier cryptosystem on LG P-970 and a Dell XPS 9100 desktop with Intel Core i7 920 2.6GHZ CPU, 9GB RAM, and Windows 7 Operating System. The desktop is used for offline computation. Table II shows the mean, maximum, minimum, medium, and standard deviation of the execution time of mul_1, mul_2, exp_1, exp_2, Enc, and Dec, where Enc and Dec denote one Paillier encryption and one decryption, respectively, and each value is computed statistically from 10,000 runs. We can see that it takes much less time to perform the same operation on Dell XPS 9100 than on LG P-970. For example, one Paillier encryption takes on average 167.21 ms and 37.53 ms on LG P-970 and Dell XPS 9100, respectively. In what follows, we assume that the offline computation is performed on desktop and is not counted into the protocol execution time.

In our experiments, we generate random profiles with each having d attributes, where every attribute value is chosen from $[0, \gamma - 1]$ uniformly at random. The performance metrics used include the offline computation time on the desktop, online computation time on the smartphone, the total net communication cost in bits, and the total online execution time including the online computation, communication, and internal processing time. Note that a complete matching process involves two independent executions of the same or even different private-matching protocols, initiated by Alice and Bob, respectively. For simplicity, we assume that Alice and Bob choose the same protocol and only show the results for one protocol execution. The total matching time thus should be twice the shown total online execution time. Finally, since Protocol 4 has the same communication and computation

TABLE I
COMPARISON OF PRIVATE-MATCHING PROTOCOLS

Protocol	Metric	Privacy	Offline Comp.	Online Comp.	Comm. (in bits)
RSV [14]	$\ell_1(\mathbf{u}, \mathbf{v})$	Level-I	$2(\gamma-1)d$ $\mathsf{exp}_1, (\gamma-1)d$ mul_2	$(\gamma-1)d+3$ $\mathsf{exp}_1, (\gamma-1)d+3$ mul_2	$2048(\gamma-1)d+3232$
Protocol 1	$\ell_1(\mathbf{u}, \mathbf{v})$	Level-I	$2(\gamma-1)d$ $\mathsf{exp}_1, (\gamma-1)d$ mul_2	2 $\mathsf{exp}_2, 2$ $\mathsf{exp}_1, \sum_{j=1}^{d} v_i+1$ mul_2	$2048(\gamma-1)d+3232$
Protocol 2	$f(\mathbf{u}, \mathbf{v})$	Level-II	$2\gamma d$ $\mathsf{exp}_1, \gamma d$ mul_2	1 $\mathsf{exp}_2, 1$ exp_1, d mul_2	$2048\gamma d+3232$
Protocol 3	$f(\mathbf{u}, \mathbf{v})<\tau$	Level-III	$2\gamma d+2$ $\mathsf{exp}_1, \gamma d+1$ mul_2	4 $\mathsf{exp}_2, 4$ $\mathsf{exp}_1, d+3$ mul_2	$2048\gamma d+7328$
Protocol 4	$\ell_{\max}(\mathbf{u}, \mathbf{v})<\tau$	Level-III	$2\gamma d+2$ $\mathsf{exp}_1, \gamma d+1$ mul_2	4 $\mathsf{exp}_2, 4$ $\mathsf{exp}_1, d+3$ mul_2	$2048\gamma d+7328$

TABLE II
EXECUTION TIME OF DIFFERENT OPERATIONS (MS)

(a) LG P-970

Operation	Mean	Max	Min	Median	Std
mul_1	0.73	40.25	0.58	0.61	1.16
exp_1	81.08	112.0	77.0	78.0	6.22
mul_2	0.88	43.00	0.73	0.76	1.14
exp_2	159.06	197.0	153.0	154.0	9.75
Enc	167.21	295.0	158.0	159.0	17.53
Dec	165.6	227.0	160.0	161.0	10.27

(b) Dell XPS 9100

Operation	Mean	Max	Min	Median	Std
mul_1	0.0076	0.10	0.0042	0.0062	0.0055
exp_1	18.84	28.0	17.0	18.0	1.54
mul_2	0.033	0.28	0.031	0.031	0.0080
exp_2	36.26	40.0	34.0	36.0	1.46
Enc	37.53	40.0	35.0	37.0	1.16
Dec	37.66	41.0	36.0	37.0	1.20

TABLE III
COMPARISON OF ENERGY CONSUMPTION FOR FOUR PROTOCOLS, WHERE $d = 100$ AND $\gamma = 5$

	Energy Consumption (J)				
Protocol	**Alice**	**Percentage**	**Bob**	**Percentage**	**Total**
RSV [14]	100	0.46%	98	0.45%	198
Protocol 1	12	0.055%	9.6	0.047%	21.6
Protocol 2	12.3	0.057%	9.6	0.047%	21.9
Protocol 3	13.2	0.061%	13.9	0.064%	27.1

overhead as Protocol 3, its performance results are not shown for brevity.

We first check the case when $\gamma = 5$ and d varies. It is not surprising to see from Fig. 6(a) that the offline computation costs of all the four protocols are proportional to d. In addition, Protocols 2 and 3 incur comparable offline computation overhead higher than that of Protocol 1 and RSV which incur the same computation overhead. The main reason is that both Protocols 2 and 3 require $\gamma d + 1$ offline Paillier encryptions,[2] while RSV and Protocol 1 require $(\gamma - 1)d$. Since users can do such offline computations on their regular computers and then synchronize the results to their mobile devices, the offline computation cost thus does not contribute to the total protocol execution time.

Fig. 6(b) shows the online computation costs of all the protocols in the $\log 10$ scale for a fixed $\gamma = 5$ and varying d. It is clear that Protocols 1 to 3 all incur much lower online computation overhead than RSV. The main reasons are that 1024-bit and 2048-bit exponentiations dominate the online computation costs of all the protocols and that Protocols 1 to 3 all require a constantly small number of modular exponentiations, while RSV requires a much larger number of modular exponentiations that increases almost linearly with d.

Fig. 6(c) compares the total net communication costs of all the protocols for a fixed $\gamma = 5$ and varying d. We can see that all the protocols incur comparable communication costs which all increase almost linearly with d, which is of no surprise.

Fig. 6(d) shows the total protocol execution time for a fixed $\gamma = 5$, which comprise the online computation, communication, and internal processing time and is dominated by the former. We can see that there are some fluctuations in the protocol execution time, mainly due to unstable transmission rate of the Bluetooth interface. In addition, Protocol 2 has the shortest execution time among Protocols 1 to 3, while Protocol 3 has the longest for achieving level-III privacy. All our protocols, however, can finish within 15 seconds under all simulated scenarios in contrast to the much longer execution time required by RSV. For example, when $d = 100$, RSV require 80.1 seconds to finish, while Protocols 1 to 3 only require 4.2, 4.2, and 4.7 seconds, respectively. Recall that a complete private-matching process involves two protocol executions. Our three protocols are thus much more feasible and user-friendly solutions to private matching for PMSN.

The impact of γ on the protocol performance is shown in Fig. 7, where d is fixed to be 100. Similar results can be observed as in Fig. 6. In particular, the online computation time of Protocols 1 to 3 are relatively insensitive to the increase in γ while that of RSV increases linearly as γ increases.

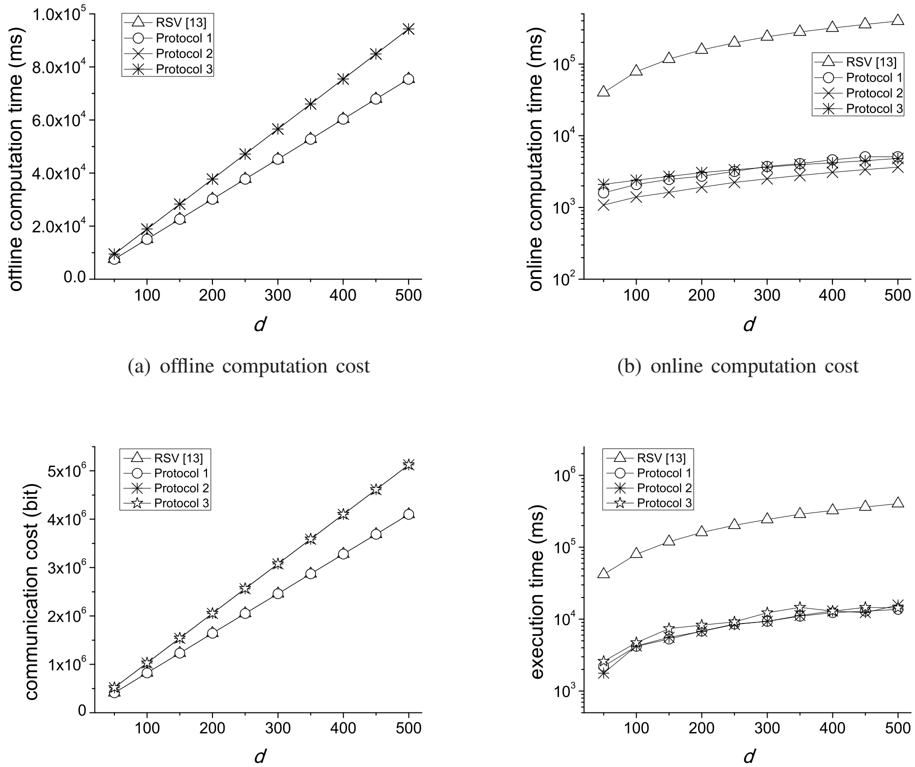
As in [5], we measure the energy consumption of our protocols using PowerTutor [18]. Table III shows that the energy consumption of one execution of RSV and Protocols 1 to 3, where $d = 100$ and $\gamma = 5$. We can see that all our protocols consumes about one eighth of the energy RSV does. In addition, a fully charged LG P-970 has 20,160J, and one execution of any of our protocols only consumes less than 0.06% of the total energy, meaning that our protocols are very practical in terms of power consumption.

C. Discussion

The experimental results show that all our protocols incur similar offline computation and communication overhead but significantly lower online computation overhead and thus total protocol execution latency and energy consumption in comparison with RSV, making them more practical than RSV to realize private matching in PMSN.

To further reduce the total execution latency of our protocols, there are two directions to explore. First, since a

[2]Recall that one Paillier encryption corresponds to two 1024-bit exponentiation and one 2048-bit multiplication.



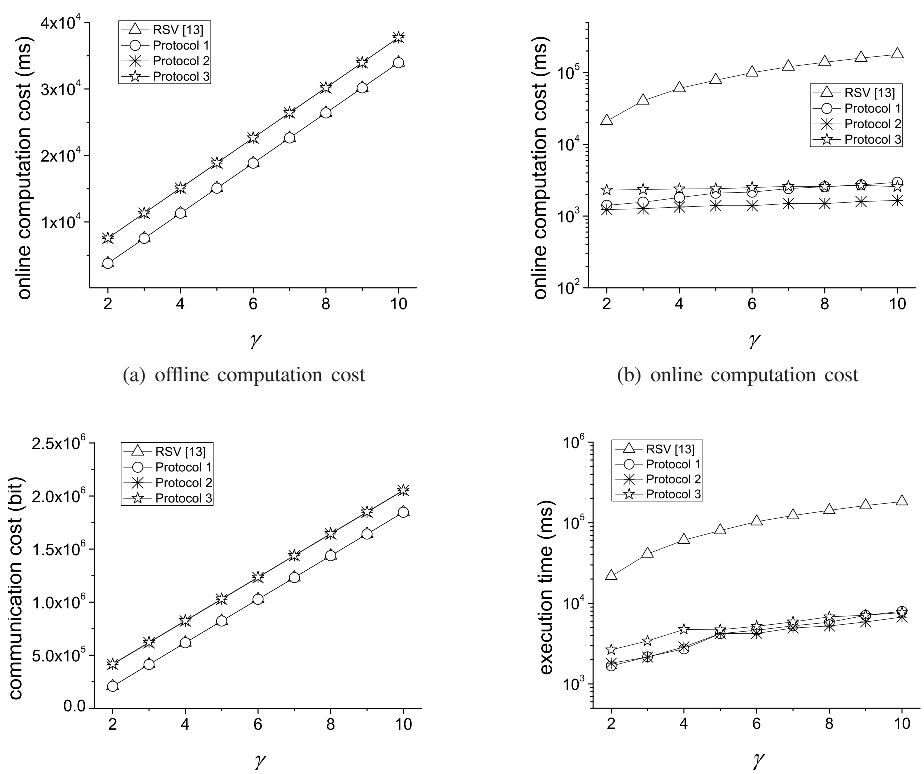
(a) offline computation cost

(b) online computation cost

(c) communication cost

(d) protocol execution time

Fig. 6. Impact of the profile dimension d, where $\gamma = 5$.



(a) offline computation cost

(b) online computation cost

(c) communication cost

(d) protocol execution time

Fig. 7. Impact of the highest attribute value γ, where $d = 100$.

significant portion of the total protocol execution time is the transmission time, it is possible to reduce the total execution latency by using advanced wireless interface with higher transmission rate. For example, Bluetooth 3.0 and 4.0 have a promised speed of 25 Mb/s, and Wi-Fi Direct has the maximum transmission rate of up to 250 Mb/s, while our current achievable transmission rate via Bluetooth interface on LG P-970 is only 800 kb/s. As more and more emerging mobile devices support these advanced interfaces, the transmission time and total protocol execution latency of our protocols will be significantly reduced. For instance, when $d = 100$ and $\gamma = 5$, with a transmission rate of 25 Mb/s, the total execution times of Protocols 1 to 3 will be close to the online computation time, i.e., 2.1s, 1.4s, and 2.4s, respectively. Second, our currently implementation uses the publicly available Java implementation of Paillier cryptosystem [15] without any optimization, further optimization is expected to further reduce the online computation time.

VI. Related Work

In addition to coarse-grained private matching for PMSN [2]–[4] introduced in Section I, the following work is most germane to our work in this paper.

Dong *et al.* [5] proposed to match two PMSN users based on the distance between their social coordinates in an online social network. By comparison, our work does not rely on the affiliation of PMSN users with a single online social network and supports more general fine-grained personal profiles as well as various matching metrics.

Private matching for PMSN can also be viewed as special instances of secure multi-party computation [20]. The prior work along this line focused on devising more efficient solutions for specific computational functions. Our work here gives efficient solutions to many PMSN matching metrics.

Securely computing some function over two vectors has also been investigated in the context of privacy-preserving data mining and scientific computation. In particular, secure dot-product computation was studied in [21]–[24]. As in [5], we adopt the method in [23] as a component of our protocols and make significant contributions on relating the computation of many PMSN matching metrics to secure dot-product computation. Moreover, some novel methods were proposed in [14] for securely computing the approximate ℓ_1 distance of two private vectors. As said before, our Protocol 1 is adapted from the protocols [14] but with significantly lower computation overhead and protocol execution latency. In addition, Du *et al.* proposed a set of protocols based on commutative encryptions for securely computing the difference between two private vectors based on different metrics [27], including the ℓ_1 distance, the ℓ_2 distance, and a more general function. Since all known commutative encryption schemes are deterministic in that the same plaintext always leads to the same ciphertext, it is not semantically secure as Paillier cryptosystem [2]. It is not clear how to apply their protocols to our problem here in an efficient and secure fashion. In addition, the different cryptographic choices between our work and [27] also lead to drastically different protocol design.

Private profile matching is also related to secret match-making schemes [32], [33], in which users express requirements about the properties expected from the other party and establish a secret key only if both users' requirements are satisfied. These schemes require the matching participants to have the valid credential issued by a certification authority, while PMSN users in our solution can set and update their own personal profiles without the need to be certified by a trusted third party. Moreover, these schemes often support a single matching matric by requiring the perfect match between two user properties. Although the recent work [30] considers fuzzy (approximate) attribute-based matching, it can only support coarse-grained matching. By comparison, our solution supports fine-grained personal profiles and also various matching metrics through drastically different protocol designs.

VII. Conclusion

In this paper, we formulated the problem of fine-grained private (profile) matching for proximity-based mobile social networking and presented a suite of novel solutions that support a variety of private-matching metrics at different privacy levels. Detailed performance analysis and experimental evaluation confirmed the high efficiency of our protocols over prior work under various practical settings.

References

[1] R. Zhang, Y. Zhang, J. Sun, and G. Yan, "Fine-grained private matching for proximity-based mobile social networking," in *INFOCOM'12*, Orlando, FL, Mar. 2012.

[2] M. Li, N. Cao, S. Yu, and W. Lou, "FindU: Privacy-preserving personal profile matching in mobile social networks," in *INFOCOM'11*, Shanghai, China, Apr. 2011.

[3] M. Arb, M. Bader, M. Kuhn, and R. Wattenhofer, "VENETA: Serverless friend-of-friend detection in mobile social networking," in *WIMOB'08*, Avignon, France, Oct. 2008.

[4] R. Lu, X. Lin, X. Liang, and X. Shen, "A secure handshake scheme with symptoms-matching for mhealthcare social network," *Mobile Networks and Applications*, pp. 1–12, 2010.

[5] W. Dong, V. Dave, L. Qiu, and Y. Zhang, "Secure friend discovery in mobile social networks," in *INFOCOM'11*, Shanghai, China, Apr. 2011.

[6] L. Kissner and D. Song, "Privacy-preserving set operations," in *CRYPTO'05*, Santa Barbara, CA, Aug. 2005.

[7] Q. Ye, H. Wang, and J. Pieprzyk, "Distributed private matching and set operations," in *ISPEC'08*, vol. 4991, Sydney, Australia, Apr. 2008.

[8] M. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *EUROCRYPT'04*, Interlaken, Switzerland, May 2004.

[9] E. Cristofaro and G. Tsudik, "Practical private set intersection protocols with linear complexity," in *FC'10*, vol. 6052, Tenerife, Canary Islands, Spain, Jan. 2010.

[10] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT'99*, Prague, Czech Republic, May 1999.

[11] R. Pickholtz, D. Schilling, and L. Milstein, "Theory of spread-spectrum communications–a tutorial," *IEEE Trans. Commun.*, vol. 30, no. 5, pp. 855–884, May 1982.

[12] R. Zhang, Y. Liu, Y. Zhang, and X. Huang, "JR-SND: jamming-resilient secure neighbor discovery in mobile ad-hoc networks," in *IEEE ICDCS'11*, Minneapolis, Minnesota, June 2011.

[13] S. Gollakota, N. Ahmed, N. Zeldovich, and D. Katabi, "Secure in-band wireless pairing," in *USENIX Security*, San Francisco, CA, Aug. 2011.

[14] S. Rane, W. Sun, and A. Vetro, "Privacy-preserving approximation of L_1 distance for multimedia applications," in *ICME'10*, Singapore, July 2010, pp. 492–497.

[15] Java implementation of Paillier cryptosystem, available at http://www.csee.umbc.edu/∼kunliu1/research/Paillier.html.

[16] A. Juels and J. Brainard, "Client puzzles: A cryptographic countermeasure against connection depletion attacks," in *NDSS'99*, San Diego, CA, Feb. 1999.

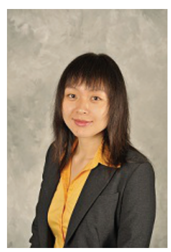
[17] P. Ning, A. Liu, and W. Du, "Mitigating dos attacks against broadcast authentication in wireless sensor networks," *ACM Trans. Sensor Networks (TOSN)*, vol. 4, no. 1, pp. 1–31, Jan. 2008.

[18] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. Dick, Z. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *CODES/ISSS'10*, Scottsdale, AZ, 2010.
[19] A. Yao, "Protocols for secure computations," in *SFCS'82*, Washington, DC, USA, Nov. 1982.
[20] O. Goldreich, S. Micali, and A. Wigderson, "How to play ANY mental game," in *STOC'87*, New York, NY, 1987.
[21] W. Du and M. Atallah, "Privacy-preserving cooperative statistical analysis," in *ACSAC'01*, New Orleans, Louisiana, Dec. 2001.
[22] I. Ioannidis, A. Grama, and M. Atallah, "A secure protocol for computing dot-products in clustered and distributed environments," in *ICPP'02*, Vancouver, British Columbia, Canada, Aug. 2002.
[23] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen, "On private scalar product computation for privacy-preserving data mining," in *ICISC'04*, vol. 3506, Seoul, Korea, Dec. 2004.
[24] M. Shaneck and Y. Kim, "Efficient cryptographic primitives for private data mining," in *HICSS'10*, Jan. 2010.
[25] H. Polat and W. Du, "Privacy-preserving collaborative filtering using randomized perturbation techniques," in *ICDM'03*, Melbourne, FL, Nov. 2003.
[26] A. Kiayias, B. Yener, and M. Yung, "Privacy-preserving information markets for computing statistical data," in *FC'09*, Berlin, Heidelberg, Feb. 2009.
[27] W. Du and M. Atallah, "Protocols for secure remote database access with approximate matching," in *WSPEC'00*, Athens, Greece, Nov. 2000.
[28] R. W. Baldwin and W. C. Gramlich, "Cryptographic protocol for trustable match making," in *IEEE S&P'85*, Oakland, CA, April 1985.
[29] J. Hoepman, "Private handshakes," in *ESAS'03*, Cambridge, UK, July 2007.
[30] G. Ateniese, M. Blanton, and J. Kirsch, "Secret handshakes with dynamic and fuzzy matching," in *NDSS'07*, San Diego, CA, Feb. 2007.
[31] A. Sorniotti and R. Molva, "A provably secure secret handshake with dynamic controlled matching," in *IFIP SEC'09*, Pafos, Cyprus, May 2009.
[32] J. Rubiera, J. Lopez, and J. Muro, "Secure matchmaking of fuzzy criteria between agents," in *KES'03*, Oxford, UK, Sept. 2003.
[33] J. Shin and V. Gligor, "A new privacy-enhanced matchmaking protocol," in *NDSS'08*, San Diego, CA, Feb. 2008.

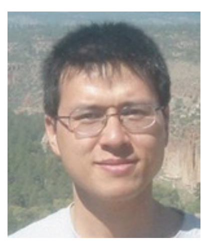
Rui Zhang received the B.E. in Communication Engineering and the M.E. in Communication and Information System from Huazhong University of Science and Technology, China, in 2001 and 2005, respectively, and the PhD degree in electrical engineering from the Arizona State University, in 2013. He was a software engineer in UTStarcom Shenzhen R&D center from 2005 to 2007. He has been an assistant professor in the Department of Electrical Engineering at University of Hawaii since July 2013. His primary research interests are network and distributed system security, wireless networking, and mobile computing.

Jinxue Zhang received B.E. in Communication Engineering from Nanjing University, China, in 2006 and the M.E. in Software Engineering from Tsinghua University, China, in 2011. He is currently a Ph.D. student in School of Electrical, Computer, and Energy Engineering at Arizona State University. His research interests are social networks, network and distributed system security, wireless networking, and mobile computing.

Yanchao Zhang received the B.E. in Computer Science and Technology from Nanjing University of Posts and Telecommunications, China, in 1999, the M.E. in Computer Science and Technology from Beijing University of Posts and Telecommunications, China, in 2002, and the Ph.D. in Electrical and Computer Engineering from the University of Florida in 2006. He is currently as an Associate Professor in School of Electrical, Computer, and Energy Engineering at Arizona State University. Before ASU, he was an Assistant Professor of Electrical and Computer Engineering at New Jersey Institute of Technology from 2006 to 2010. His primary research interests are network and distributed system security, wireless networking, and mobile computing. He is (was) an Associate Editor of IEEE Trans. Vehicular Technology, a Feature Editor of IEEE Wireless Communications, a Guest Editor of IEEE Wireless Communications Special Issue on Security and Privacy in Emerging Wireless Networks in 2010, and a TPC Co-Chair of Communication and Information System Security Symposium, IEEE GLOBECOM 2010. He received the NSF CAREER Award in 2009.



Jinyuan (Stella) Sun Jinyuan Sun received the BSc degree in computer information systems from Beijing Information Technology Institute, China, in 2003, the MASc degree in computer networks from Ryerson University, Canada, in 2005, and the PhD degree in electrical and computer engineering from the University of Florida, in 2010. She was a Network Test Developer at RuggedCom Inc., Ontario, Canada, 2005-2006. She has been an assistant professor in the Department of Electrical Engineering and Computer Science at University of Tennessee Knoxville since August 2010. Her research interests include the security protocol and architecture design of wireless networks. She is a member of the IEEE and ACM.



Guanhua Yan Guanhua Yan obtained his Ph.D. degree in Computer Science from Dartmouth College, USA, in 2005. From 2003 to 2005, he was a visiting graduate student in the Coordinated Science Laboratory at University of Illinois, Urbana-Champaign. He is now working as a Research Scientist in the Information Sciences Group (CCS-3) at Los Alamos National Laboratory. His research interests include cyber-security, networking, and large-scale modeling and simulation techniques. He has contributed about 50 articles in these fields.