FindU: Privacy-Preserving Personal Profile Matching in Mobile Social Networks

Ming Li*, Ning Cao*, Shucheng Yu† and Wenjing Lou*
*Dept. of ECE, Worcester Polytechnic Institute, Email: {mingli, ncao, wjlou}@ece.wpi.edu
†Dept. of CS, University of Arkansas at Little Rock, Email: sxyu1@ualr.edu

***Abstract*—Making new connections according to personal preferences is a crucial service in mobile social networking, where the initiating user can find matching users within physical proximity of him/her. In existing systems for such services, usually all the users directly publish their complete profiles for others to search. However, in many applications, the users' personal profiles may contain sensitive information that they do not want to make public. In this paper, we propose FindU, the first privacy-preserving personal profile matching schemes for mobile social networks. In FindU, an initiating user can find from a group of users the one whose profile best matches with his/her; to limit the risk of privacy exposure, only necessary and minimal information about the private attributes of the participating users is exchanged. Several increasing levels of user privacy are defined, with decreasing amounts of exchanged profile information. Leveraging secure multi-party computation (SMC) techniques, we propose novel protocols that realize two of the user privacy levels, which can also be personalized by the users. We provide thorough security analysis and performance evaluation on our schemes, and show their advantages in both security and efficiency over state-of-the-art schemes.**

I. Introduction

With the proliferation of mobile devices, mobile social networks (MSNs) are becoming an inseparable part of our lives. Leveraging networked portable devices such as smart phones and PDAs as platforms, MSN not only enables people to use their existing online social networks (OSNs) at anywhere and anytime, but also introduces a myriad of mobility-oriented applications, such as location-based services and augmented reality. Among them, an important service is to make new social connections/friends within physical proximity based on the matching of personal profiles. For example, MagnetU [1] is a MSN application that matches one with nearby people for dating or friend-making based on common interests. In such an application, a user only needs to input some (query) attributes in her profile, and the system would automatically find the persons around with similar profiles. The scopes of these applications are very broad, since people can input anything as they want, such as hobbies, phone contacts and places they have been to. The latter can even be used to find "lost connections" [2] and "familiar strangers" [3].

However, such systems also raise a number of privacy concerns. Let us first examine a motivating scenario. In a hospital, patients may include their illness symptoms and medications in their personal profiles in order to find similar patients, for physical or mental support. In this scenario, an initiating user (initiator) may want to find out the patient having the maximum number of identical symptoms to her, while being reluctant to disclose her sensitive illness information to the rest of the users, and the same for the users being matched with. If users' private profiles are directly exchanged with each other, it will facilitate user profiling where those information can be easily collected by a nearby user, either in an active or passive way; and those user information may be exploited in unauthorized ways. For example, a salesman from a pharmacy may submit malicious matching queries to obtain statistics on patients' medications for marketing purposes. To cope with user profiling in MSNs, it is essential to disclose minimal and necessary personal information to as few users as possible.

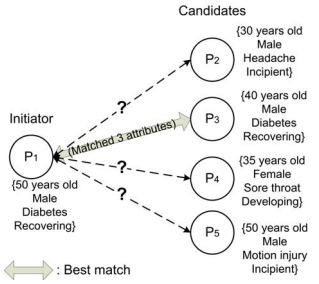


Fig. 1. Private profile matching in mobile social networks.

In fact, the ideal situation is to let the initiator and its best matching user directly and privately find out and connect to each other, without knowing anything about other users' profile attributes, while the rest of the users should also learn nothing about the two user's matching attributes. The scenario is illustrated in Fig. 1, where the party P_1 is the initiator and the others are called "candidates". P_1's best matching user is P_3, who shares the maximum number of symptoms with her. Since directly publishing all the profile attributes is undesirable, it is challenging to find out the matching users privately. One may think of simply turning off the cellphone or input very few attributes, but these would interfere with the system usability. Recently, Yang *et. al*. proposed E-SmallTalker [4], a practical system for matching people's interests before initiating a small-talk. However, E-SmallTalker reveals the exact common attributes between the initiator and every other user, which could be more than necessary. Another difficulty of private matching under a MSN setting is the lack of a centralized authority. Lu *et. al.* [5] proposed a symptom matching scheme for mobile health social networks, assuming the existence of a semi-online central authority.

In this paper, we overcome the above challenges and make the following main contributions.

(1) We formulate the privacy preservation problem of profile

matching in MSN. Three increasing levels of privacy are defined, where the information learnt by the initiator and each candidate includes: the intersection set between their profile attributes, the size of their intersection set, and the rank of their intersection set size, respectively.

(2) We propose two fully distributed privacy-preserving profile matching protocols. The basic ideas come from private set-intersection (PSI) techniques. However, solutions based on existing PSI schemes are less efficient. We leverage secure multi-party computation (SMC) based on polynomial secret sharing, and propose several key enhancements to improve the computation and communication efficiency. Also, users can choose personalized privacy levels when running the same matching instance.

(3) We provide thorough security analysis and performance evaluation for our schemes. Our schemes achieve several security properties not achieved by previous works, i.e., they are not only secure under the honest-but-curious (HBC) model but can also prevent several key malicious attacks. Meanwhile, they are shown to be more efficient under the settings of MSN.

II. Problem Definition

A. System Model

Our system consists of N users (or parties) denoted as $P_1, ..., P_N$, each possessing a portable device. We denote the initiating party (*initiator*) as P_1. P_1 launches the matching process and its goal is to find one party that best *"matches"* with it, from the rest of the parties $P_2, ..., P_N$ which are called *candidates*. Each party P_i's profile consists of a set of attributes $\mathcal{S}_i$, which can be strings up to a certain length. P_1 defines a matching query to be a subset of $\mathcal{S}_1$, and in the following we use $\mathcal{S}_1$ to denote the query unless specified. Also, we denote $n = |S_1|$ and $m = |S_i|, i > 1$, assuming each candidate has the same set size for simplicity.

There could be various definitions of "match". In this paper, to keep it simple, we consider $|\mathcal{S}_1 \cap \mathcal{S}_i| > 0$ as match (same with [4]). The *best match*, P_{i^*} is defined as the party having the maximum intersection set size with P_1. P_1 will first find out P_{i^*} via our protocols, and then they decide whether to connect with each other based on their actual intersection set.

For the network, we assume devices communicate through wireless interfaces such as bluetooth or WIFI. For simplicity, we assume every participating device is in the communication range of each other. In addition, we assume that a secure communication channel has been established between each pair of users, which can be done easily if each device has a public/private key pair. Otherwise, we can use the group device pairing technique [6] to establish pairwise session keys.

We *do not* assume the existence of a trusted third party during the protocol run; all parties carry out profile matching in a completely distributed way. They may cooperate with each other, i.e., when P_1 runs the protocol with each P_i, a subset of the rest of parties would help them to compute their results.

B. Adversary Model

An outsider can eavesdrop the communication channel or modify, replay and inject messages; however it is not our main focus to prevent against active attacks from outsiders. From now on, we will deal with insiders who are participators of the matching protocol. An insider's goal is to conduct *user profiling*, i.e., obtain as much personal profile information of other nearby users as possible. With a user's attributes, a bad guy could correlate and identify that user via its MAC addresses or public keys. However, we cannot absolutely prevent user profiling, because at least the initiator and its best matching user will mutually learn the intersection set between them to make connections. Thus we focus on minimizing the amount of private information revealed in one protocol run.

The parties could try to learn more information than allowed, by either inferring from the results but honestly following the protocol, or actively deviating from it. The former corresponds to the *honest-but-curious* (HBC) model, while the latter corresponds to the *malicious* model [7]. In this paper, the proposed protocols are proven secure under the HBC model; although not proven secure under the malicious model, we analyze a number of active attacks and show how they are secure against them.

The adversary may act alone (be any single party) or several parties may collude. We assume that the size of a coalition is smaller than a threshold t, where t is a parameter. And we shall also assume $N \geq 2t + 1$ for our proposed schemes.

C. Design Goals

1) Security Goals: Our main security goal is to thwart user profiling attack. Since the users may have different privacy requirements and it takes different amount of efforts in protocol run to achieve them, we hereby define three levels of privacy where a higher level leaks less information to the adversaries. Note that, by default, all of the following include letting P_1 and the best match P_{i^*} learn the intersection set between them at the end of a protocol run.

Definition 1 (Privacy Level 1 (PL-1)): When the protocol ends, P_1 and each candidate $P_i, 2 \leq i \leq N$ mutually learn the intersection set between them: $\mathcal{I}_{1,i} = \mathcal{S}_1 \cap \mathcal{S}_i$. An adversary $\mathcal{A}$ (whose behavior is defined in Sec. II-B) should learn nothing beyond what can be derived from the above outputs and its private inputs.

If we assume the adversary has unbounded computing power, PL-1 actually corresponds to unconditional security for all the parties under the HBC model. Obviously, in PL-1, P_1 can obtain all candidates' intersection sets just in one protocol run. Thus it reveals too much user information to the attacker if he assumes the role of P_1. Therefore we define privacy level 2 in the following.

Definition 2 (Privacy Level 2 (PL-2)): When the protocol ends, P_1 and each candidate $P_i, 2 \leq i \leq N$ mutually learn the size of their intersection set: $m_{1,i} = |\mathcal{S}_1 \cap \mathcal{S}_i|$. In addition, the best match P_{i^*} is allowed to know the $m_{1,i}$ values with other P_is. The adversary $\mathcal{A}$ should learn nothing beyond what can be derived from the above outputs and its private inputs.

In PL-2, except when $m_{1,i} = |\mathcal{S}_1|$ or $|\mathcal{S}_i|$, P_1 and each P_i both will not learn exactly which attributes are in $\mathcal{I}_{1,i}$. The additional information for P_{i^*} is intended for it to learn whether itself is the best match under active attacks. In PL-2, the adversary needs to run the protocol multiple times to

obtain the same amount of information with what he can obtain under PL-1 when he assumes the role of P_1. However, PL-2 still allows $\mathcal{A}$ to guess which attributes are in the matching set with non-negligible probability, especially when the attribute sets are small.

Definition 3 (Privacy Level 3 (PL-3)): When the protocol ends, P_1 and each P_i should only learn the ranks of each value $m_{1,i}, 2 \leq i \leq N$. $\mathcal{A}$ should learn nothing more than what can be derived from the outputs and its private inputs.

In PL-3, we can require that P_1 only contacts the *best match* P_{i^*}, such that it only obtains the intersection set $\mathcal{I}_{1,i^*}$ with the best match. If there is a tie, then the party with lowest ID is chosen as the best match. In this way, $\mathcal{A}$ will need at least $N-1$ protocol runs to learn all other user's exact profile attributes, and thus $\mathcal{A}$'s profiling capability is much limited.

2) Usability and Efficiency: For profile matching in MSN, it is desirable to involve as few human interactions as possible. In this paper, a human user only needs to explicitly participate in the end of the protocol run, e.g., decide whether to connect when he/she becomes the best match. In addition, the system design should be *lightweight and practical*, i.e., being efficient enough in computation and communication to be used in MSN. Finally, the users (especially the candidates) shall have the option to flexibly *personalize their privacy levels*.

D. Challenges

It is very challenging to achieve all the design goals simultaneously, especially if we desire high level of security but are unwilling to pay the high costs of computation and communication. Similar problems to ours can be found in the literature, namely private set intersection (PSI) and private cardinality of set intersection (PCSI) [7], and they are mostly tackled under the framework of Secure Multi-party Computation (SMC). The general SMC techniques [8] are often far from efficient. Researchers have proposed various customized solutions for those problems, but when applied to the ones defined here, they lead to high energy consumption and long protocol run time. In this paper, we explore novel methods with higher efficiency, while achieving reasonable security (resist a threshold number of colluders).

E. Relations to Existing Problems

In PL-1, each sub-protocol (between P_1 and P_i) relates to the two-party PSI problem [7], [9], [10], while the PL-2 relates to two-party PCSI [7], [9], [11]. PL-3 is most related to the privacy-preserving nearest neighbor search problem [12], [13].

Unlike most existing problems in PSI and PCSI, we require the output of the sub-protocol between P_1 and each P_i be secret-shared between them, so that the result can be revealed to both party at once to prevent cheating. This turns out to be an essential idea to minimize user profiling under malicious behavior. In addition, we define our security under the threshold cryptography model, which allows us to explore more efficient solutions. Finally, our problems are defined under the distributed setting, where there is no client-server relationship nor any central party. Such framework is applicable to many scenarios beyond the motivating problems in this paper.

III. Related Work

The related works are mainly in the area of PSI or distributed private matching. Early works such as [14] used hash function to compare the hashed attribute sets of two parties. The E-SmallTalker scheme [4] used bloom filter to represent a set. Current techniques can be categorized into three main approaches.

A. Protocols Based on Oblivious Polynomial Evaluation

This approach dates back to the FNP scheme [7], where a client and a server compute their intersection set such that the client gets the result while server learns nothing. Additive homomorphic encryption is used to obliviously evaluate a polynomial that represents client's input. Later, Kissner and Song (KS) proposed a scheme [11] that enables set intersection, union, cardinality and over-threshold operations, which is improved by Sang *et. al*. [15]. Ye *et. al*. [9] extended the FNP scheme to a distributed private matching scheme; recent work [16] focused on reducing the complexity in the malicious model. However, all these schemes heavily rely on homomorphic encryption and none of them essentially achieves linear computational complexity in terms of encryption operations.

Another line of works pursue information theoretical security. In [17], Li and Wu (LW) proposed an unconditionally secure multi-party PSI scheme. Their idea is similar to the KS scheme, while the inputs are shared among all parties using secret sharing, and computations are done on those shares. Later, Narayanan *et. al*. [18] improved the LW scheme using the same idea of FNP, and also proposed an N-party PCSI scheme. In [19] Shaneck and Kim proposed using secret sharing to compute dot product securely, but they assume the existence of trusted third parties. These schemes have low computation complexities, but high communication costs.

B. Protocols Based on Oblivious Pseudo Random Functions

This approach is due to Hazay and Lindell [20]. The idea is that two parties securely compute a pseudo random function where one of them holds the key while the other provides the input (set elements). They achieved linear complexity w.r.t. set size, but the number of exponentiations is large. To improve the efficiency, Jarecki and Liu [21] proposed a scheme where the complexity is smaller than FNP's when the set size is sufficiently large. Recently in [10], Cristofaro and Tsudik proposed a construction of PSI based on blind RSA signatures. The server computes $(H(x_j))^d$ for the client obliviously, where x_js are client's inputs and d is server's one-time RSA signing key. Their scheme is more efficient than previous ones; however, it does not achieve PCSI.

C. Protocols Based on Commutative Encryption

In [22], Agrawal *et. al*. proposed using commutative encryption to realize PSI and PCSI in information sharing between two databases. A commutative encryption scheme $E_k(\cdot)$ has the following property: $E_{k_1}(E_{k_2}(x)) = E_{k_2}(E_{k_1}(x))$, and is used as a keyed one-way hash function to generate a mapping for each element x such that no one knows the key. Later, Vaidya *et. al*. [23] extended their scheme to N-party setting. Arb *et. al*. [24] applied the idea to detect friend-of-friend in MSN. However, known commutative encryption primitives are all

TABLE I
MAIN NOTATIONS

N, t	Number of parties, maximum number of colluders
$[s]_i^{t,w}$	Party P_i's secret share of s (under (t,w)-SS)
$\mathcal{S}_1$, $\mathcal{S}_i$	P_1's query attribute set, and P_i's profile attribute set
$x_j, 1 \le j \le n$	P_1's query set elements, $n = \|\mathcal{S}_1\|$
$y_{ij}, 1 \le j \le m$	P_i's profile set elements, $m = \|\mathcal{S}_i\|, i \in \{2, \cdots, N\}$
$\mathcal{I}_{1,i}$	Intersection set between P_1 and P_i; $m_{1,i} = \|\mathcal{I}_{1,i}\|$
$\mathcal{F}_p$	The finite field used; $= \log p$: security parameter
$\mathcal{H}()$	A cryptographic hash function
$\overset{R}{\leftarrow}$, $\|$	Random sampling from a set, concatenation
$\mathcal{P}$, P_1, P_i	The set of all parties, the initiator and the ith party
$\mathcal{P}_i$, $\mathcal{P}_i'$	The computing set and reconstruction set for P_i

deterministic, i.e., they provide weaker security guarantees.

Remark. Protocols in the above achieve linear complexity; however, they are asymmetric, i.e., the result is only known or first known by one party (say, Alice), which is undesirable in MSN profile matching. Alice can send the result back to Bob but this requires Alice be trusted. Otherwise, Alice can cheat [25]. Thus, it is desirable to design a symmetric protocol such that the result is known and verifiable by both parties.

Other protocols: In [26], Emekci *et. al*. proposed a privacy-preserving intersection/equijoin protocol between two databases using third parties in a P2P network. Secret sharing is used, however the security is weakened since the polynomial used is not random. Chow *et. al*. [27] addressed the problem of privacy-preserving queries over distributed databases, and proposed a two-party computation model. Their problem setting involves two trusted parties.

IV. Notations and Technical Preliminaries

Notations. Please refer to Table I. Note that, unless specified, we denote $[s]_i$ as P_i's $(t, 2t+1)$-share of secret s under SS scheme, and when we mention P_i, we refer to $2 \le i \le N$.

Preliminaries. **Shamir Secret Sharing Scheme (SS)**. A (t,w)-SS scheme [28] shares secret s among w parties by giving each party P_i the value $[s]_i^{t,w}$, and if at most t parties collude, they cannot gain any useful information about s.

Secure Multiparty Computation (SMC) based on SS. For addition, SS is homomorphic: let α and β be two secrets shared using (t,w)-SS, we have $[\alpha+\beta]_i^{t,w} = [\alpha]_i^{t,w} + [\beta]_i^{t,w}$ [29]. However, for secure multiplication, one round of communication is needed and it is required that $w \ge 2t+1$ [30]. Gennaro *et. al.* proposed an efficient and secure SS-multiplication protocol [31]. Let the inputs of party P_i be $[\alpha]_i^{t,w}$ and $[\beta]_i^{t,w}$:

Round 1. Each party P_i shares the value $[\alpha]_i^{t,w}[\beta]_i^{t,w}$ by choosing a t-degree random polynomial $h_i(x)$, s. t. $h_i(0) = [\alpha]_i^{t,w}[\beta]_i^{t,w}$. He sends the value $h_i(j)$ to party P_j, $1 \le j \le w$.

Round 2: Every party P_j computes his share of $\alpha\beta$, i.e., the value $H(j) = [\alpha\beta]_j^{t,w}$ under a t-degree random polynomial H, by locally computing the linear combination $H(j) = \sum_{i=1}^{w} \lambda_i h_i(j)$, where $\lambda_1, ..., \lambda_w$ are known constants.

In this protocol, round 1 incurs $O(w^2\ )$ communication cost in total, and $O(w\ )$ for each party.

Additive Homomorphic Encryption. An additive homomorphic encryption scheme E allows one to compute $E(m_1 + m_2)$ given $E(m_1)$ and $E(m_2)$, without knowing the private key. This is only used in our protocol for PL-2.

V. Design of FindU

A. Overview

We present two protocols that aim at realizing one level of privacy requirement each. We start with the basic scheme realizing PL-1. We base our idea on the FNP scheme [7], but use secret sharing to compute polynomial evaluation securely. At a high level, for P_1 and each P_i $(2 \le i \le N)$, their inputs are shared among a subset $\mathcal{P}_i$ of $2t+1$ parties (the computing set) using $(t, 2t+1)$-SS, based on which they cooperatively compute shares of the function $F_i(x_j) = R_{ij} \cdot f_i(x_j) + x_j$ for each $1 \le j \le n$, where $f_i(y)$ is the polynomial representing P_i's set, and R_{ij} is a random number jointly generated by P_1 and P_i but not known to any party. We have $x_j \in \mathcal{I}_{1,i}$ iff. $F_i(x_j) = x_j$. The values of $\{F_i(x_j)\}_{1 \le j \le n}$ remain in secret-shared forms between P_1 and P_i before their shares are revealed to each other, to provide verifiability. To reduce the communication complexity, we propose an enhancement to the secure polynomial evaluation computation.

For PL-2, the advanced scheme achieves efficient PCSI. The main idea is that, the parties in $\mathcal{P}_i$ first compute the $(t, 2t+1)$-shares of the function $F_i(x_j) = R_{ij} \cdot f_i(x_j)$, $1 \le j \le n$ securely using the basic scheme, whereas $x_j \in \mathcal{I}_{1,i}$ iff. $R_{ij} \cdot f_i(x_j) = 0$. In order to blind from P_1 the correspondence between its inputs $\{x_j\}$ $(j \in \{1, \cdots, n\})$ and the outputs $F_i(x_{j'})$ $(j' \in \{1, \cdots, n\})$, we employ a blind-and-permute (BP) method. To reduce the number of invocations of the BP protocol, we use share conversion to convert the $(t, t+1)$-shares of $\{F_i(x_j)\}_{1 \le j \le n}$ (held by parties in the reconstruction set $\mathcal{P}_i'$) into $(2,2)$-shares shared between P_1 and P_i, so that only one BP invocation is needed between P_1 and each P_i. The security of both the basic and advanced schemes are proven.

Finally, we also discuss possible solutions to achieve PL-3, and leave practical solutions that achieve PL-3 as future work.

B. The Basic Scheme

The basic scheme consists of four phases, and is described in Fig. 2. The following two definitions capture the idea to involve the minimum number of parties during computation.

Definition 4 (Computing Set of P_i): A set of $2t+1$ parties $\mathcal{P}_i \subset \mathcal{P}$ who help P_1 and P_i to compute their shares of $F_i(x_j)$, $1 \le j \le n$. $\mathcal{P}_i$ includes P_1 and P_i, and the rest $2t-1$ parties are chosen as $P_{i+1}, P_{i+2}, \cdots$ with indices wrapping around.

Definition 5 (Reconstruction Set of P_i): A set of $t+1$ parties $\mathcal{P}_i' \subset \mathcal{P}_i$, who will contribute their shares of $F_i(x_j)$, $1 \le j \le n$ to P_1 and P_i for reconstruction. $\mathcal{P}_i'$ also includes P_1 and P_i, and the rest $t-1$ parties are chosen in the same way as for the computing set.

At first, each party has a set of attributes: P_1 has $\mathcal{S}_1 = \{x_1, x_2, ..., x_n\}$ and P_i has $\mathcal{S}_i = \{y_{i1}, y_{i2}, ..., y_{im}\}$, respectively, where each element is an encoded attribute in $\mathcal{F}_p$. For example, a hash algorithm can be used for encoding. Rather than publishing the sets as they are, each P_i first generates an m-degree polynomial based on $\mathcal{S}_i$ as follows:

$$f_i(y) = (y - y_{i1}) \cdot (y - y_{i2}) \cdots (y - y_{im}) = \sum_{k=0}^{m} a_{ik} y^k, \qquad (1)$$

where $\{a_{ik}\}_{0 \le k \le m}$ are coefficients and $a_{im} \equiv 1$. Obviously, for each of P_1's attribute x_j, $f_i(x_j) = 0$ iff. $x_j \in \mathcal{S}_i$. In order to let

P_1's private inputs: $\mathcal{S}_1 = \{x_1, ..., x_n\}$, $\mathcal{I}_{1,i} \leftarrow \emptyset$; P_i's private inputs: $\mathcal{S}_i = \{y_{i1}, ..., y_{im}\}$, $\mathcal{I}'_{1,i} \leftarrow \emptyset$. Public inputs: t, $\mathcal{P}_i, \mathcal{P}'_i, 2 \le i \le N$.

1. Data Share Distribution:

1) P_1 generates $\{r_{ij}\}_{2\le i\le N, 1\le j\le n}$ randomly from $\mathcal{F}_p$. And for each $2 \le i \le N$, it shares $\{r_{ij}\}_{1\le j\le n}$ among parties in $\mathcal{P}_i$ by giving $\{[r_{ij}]_l\}_{1\le j\le n}$ to each $P_l \in \mathcal{P}_i$, using $(t, 2t+1)$-SS. Also, P_1 shares $\{x_j, x_j^2, ..., x_j^m\}_{1\le j\le n}$ among all N parties in $\mathcal{P}$, by giving each $P_l \in \mathcal{P}$ the values $\{[x_j]_l^{t,N}, [x_j^2]_l^{t,N}, ..., [x_j^m]_l^{t,N}\}_{1\le j\le n}$, using (t, N)-SS.
2) Each $P_i \in \mathcal{P}$ generates $\{r'_{ij}\}_{1\le j\le n}$ randomly from $\mathcal{F}_p$, lets $f_i(y) = \prod_{k=1}^{m}(y - y_{ik}) = \sum_{k=0}^{m} a_{ik}y^k$, and shares $\{r'_{ij}\}_{1\le j\le n}$, $\{a_{ik}\}_{0\le k\le m-1}$ among parties in $\mathcal{P}_i$ using $(t, 2t+1)$-SS, by giving each $P_l \in \mathcal{P}_i$ the values $\{[r'_{ij}]_l\}_{1\le j\le n}$ and $\{[a_{ik}]_l\}_{0\le k\le m-1}$.

2. Computation: Each party $P_l \in \mathcal{P}$ executes the following:

1) For each i that $\mathcal{P}_i \ni P_l$ and for each $j, 1 \le j \le n$, P_l locally computes $z_{ijl} = [a_{i1}]_l[x_j]_l + ... + [a_{i,m-1}]_l[x_j^{m-1}]_l$.
2) P_l generates new t-degree random polynomials $\{h_{ijl}(y)\}_{\forall i, \mathcal{P}_i \ni P_l, 1\le j\le n}$ such that $h_{ijl}(0) = z_{ijl}$, and for each i that $\mathcal{P}_i \ni P_l$, it shares $\{z_{ijl}\}_{1\le j\le n}$ among parties in $\mathcal{P}_i$ using $(t, 2t+1)$-SS.
3) P_l receives $\{[z_{ijk}]_l\}_{\forall i, \mathcal{P}_i \supset \{P_l, P_k\}, 1\le j\le n}$, and for each i that $\mathcal{P}_i \ni P_l$ and $1 \le j \le n$, locally computes its $(t, 2t+1)$-share of $f_i(x_j)$ as: $[f_i(x_j)]_l = [a_{i0}]_l + \sum_{P_k \in \mathcal{P}_i} \lambda_{ik}[z_{ijk}]_l + [x_j^m]_l = H_{ij}(l)$, where $H_{ij}(y)$ is a random t-degree polynomial (unknown to P_l) and $\lambda_{i1}, ..., \lambda_{i,2t+1}$ are publicly known numbers pertinent to each $\mathcal{P}_i$.
4) P_l computes $[R_{ij}]_l = [r_{ij}r'_{ij}]_l$ and $[F_i(x_j)]_l = [R_{ij}f_i(x_j) + x_j]_l$, $\forall i$ that $\mathcal{P}_i \ni P_l$ and $1 \le j \le n$, through two invocations (rounds) of the SS-multiplication protocol.

3. Reconstruction:

1) Commit shares: each $P_l \in \mathcal{P}$ computes $c_i = \mathcal{H}([F_i(x_1)]_l \| [F_i(x_2)]_l \| \cdots \| [F_i(x_n)]_l)$, for each i that $\mathcal{P}'_i \ni P_l$, and sends them to both P_1 and P_i.
2) Aggregate shares: each $P_l \in \mathcal{P}$ sends its shares $\{[F_i(x_j)]_l\}_{\forall i, \mathcal{P}'_i \ni P_l, 1\le j\le n}$ to P_1 and P_i, who verify their authenticity. If any of them fails, an abort signal is broadcasted.
3) Reconstruction: if no abort signal is received, P_1 computes $F_i(x_j) = \sum_{P_l \in \mathcal{P}'_i} \gamma_{il}[F_i(x_j)]_l$ for each i that $\mathcal{P}'_i \ni P_1$, $1 \le j \le n$, where $\{\gamma_{il}\}_{P_l \in \mathcal{P}'_i}$ are Lagrangian coefficients of $\mathcal{P}'_i$. If $F_i(x_j) \in \mathcal{S}_1$, $\mathcal{I}_{1,i} \leftarrow \mathcal{I}_{1,i} \bigcup F_i(x_j)$. The similar is done for P_i.

4. Matching:

P_i sends a request to the best match P_{i^*} based on the intersection sets obtained; if P_{i^*} accepts, a "connection" is established.

Fig. 2. The basic scheme, which is carried out between the initiator P_1 and each candidate P_i.

P_i also know the result, we shall compute function $F_i(x_j) = R_{ij} \cdot f_i(x_j) + x_j$ for each $1 \le j \le n$, where $R_{ij} = r_{ij}r'_{ij}$, r_{ij} and r'_{ij} are random numbers generated by P_1 and P_i, respectively. In this way, if $F_i(x_j) \in \mathcal{S}_i$ then $x_j \in \mathcal{I}_{1,i}$ with high probability, and if $F_i(x_j) \notin \mathcal{S}_i$ then P_i knows $x_j \notin \mathcal{I}_{1,i}$. The same applies for P_1. P_1 and each P_i compute the above function securely by secret-sharing their inputs among P_i's *computing set*.

1) Data Share Distribution: First, P_1 generates the 1 to m powers of each of its set elements: $\{x_j^1, ..., x_j^m\}_{1\le j\le n}$, and shares them among all N parties using (t, N)-SS, by giving each $P_l \in \mathcal{P}$ the values $\{[x_j]_l^{t,N}, [x_j^2]_l^{t,N}, ..., [x_j^m]_l^{t,N}\}_{1\le j\le n}$. This is because P_1 is to compute the intersection set with each candidate P_i, which is in its own computing set $\mathcal{P}_i$. But for each P_i, its private inputs $\{a_{ik}\}_{0\le k\le m-1}$ are only shared among parties in its own computing set using $(t, 2t+1)$-SS. Also, P_1 and each P_i generate random numbers $\{r_{ij}\}_{1\le j\le n}$ and $\{r'_{ij}\}_{1\le j\le n}$, respectively and separately, to blind values $\{f_i(x_j)\}_{1\le j\le n}$. These numbers are shared among the parties in $\mathcal{P}_i$ similarly.

Now, each party $P_l \in \mathcal{P}$ possess the following: $\{[x_j]_l, [x_j^2]_l, ..., [x_j^m]_l\}_{1\le j\le n}$, $\{[a_{ik}]_l\}_{\forall i, \mathcal{P}_i \ni P_l, 0\le k\le m-1}$, $\{[r_{ij}]_l\}_{\forall i, \mathcal{P}_i \ni P_l, 1\le j\le n}$, and $\{[r'_{ij}]_l\}_{\forall i, \mathcal{P}_i \ni P_l, 1\le j\le n}$.

2) Computation Phase: In this phase, the parties in each $\mathcal{P}_i$ participate in secure computation of the shares of $\{F_i(x_j)\}_{1\le j\le n}$. In particular, to evaluate $f_i(x_j)$ securely, a straightforward way is to compute $m-1$ multiplications of $a_{ik}x_j^k$, $1 \le k \le m-1$ by invoking the SS-multiplication protocol $m-1$ times. However, this will introduce too much communication cost.

Therefore, we propose to aggregate those multiplications into one round. The idea is to let each party $P_l \in \mathcal{P}_i$ first locally compute $z_{ijl} = \sum_{k=1}^{m-1}[a_{ik}]_l[x_j^k]_l$ for each $1 \le j \le n$. Since z_{ijl} can be viewed as the $(2t, 2t+1)$-share of $Z_{ij} = \sum_{k=1}^{m-1} a_{ik}x_j^k$ under a new $2t$-degree polynomial which is not random, we need to re-randomize z_{ijl}. Also, to do further multiplications, we need to reduce that polynomial's degree to t. Using the same idea from [31], P_l re-shares each z_{ijl} among other parties in $\mathcal{P}_i$ using $(t, 2t+1)$-SS. Each party P_l upon receiving others' (secondary) shares of their z_{ijk} can linearly reconstruct its $(t, 2t+1)$-share of Z_{ij}. Since $F_i(x_j) = r_{ij}r'_{ij}(Z_{ij} + a_{i0} + x_j^m) + x_j$, P_l's share of $F_i(x_j)$ can then be easily computed by invoking two more SS-multiplications.

Lemma 1: The computation phase correctly computes the $(t, 2t+1)$-shares of $F_i(x_j)$ at each $P_l \in \mathcal{P}_i$.

Proof: Omitted due to limited space. Please see [32]. ■

3) Reconstruction Phase: In this phase, at least $t+1$ shares of $F_i(x_j)$ are needed to reconstruct $F_i(x_j)$. To this end, the parties in each reconstruction set $\mathcal{P}'_i$ (for P_i, $2 \le i \le N$) commit their shares to P_1 and P_i so that they cannot change them later. Next, they reveal their shares to P_1 and P_i only, who can verify the correctness of them and obtain $F_i(x_j)$ by polynomial interpolation. P_1 and P_i can test if $F_i(x_j) \in \mathcal{S}_1$ and $F_i(x_j) \in \mathcal{S}_i$ respectively, to determine their intersection set $\mathcal{I}_{1,i}$. Next, in the matching phase, we do not restrict P_1 to send connection requests only to the best match, since it already knows the intersection sets with all candidates.

C. The Advanced Scheme

Observe that, in the basic scheme if we set $F_i(x_j) = r_{ij}r'_{ij}f_i(x_j)$, then the result will be 0 if $x_j \in \mathcal{I}_{1,i}$, otherwise be a random number. In order to obtain the number of matching attributes ($m_{1,i}$), one way is to employ the equality-test protocol [33] based on SS, which tests whether each shared secret $F_i(x_j)$ equals to 0 or not, and the output is 0 or 1 which is still shared among the parties. However, the equality-test protocol incurs too high communication cost. Because in modern smart mobile devices, the wireless transmission is usually more costly than computation, we would like to tradeoff computation for communication efficiency.

Thus, we adopt a *blind-and-permute* (BP) method to obliviously permute P_1's shares of each $F_i(x_j)$, so that the linkage between $F_i(x_j)$ and its corresponding attribute x_j is broken. A BP protocol between two parties A and B where each data item (e.g., $F_i(x_j)$) is additively split between them is described in [12]. The main idea is, A encrypts each of its shares using additive homomorphic encryption and sends to

B. B then generates a different random number r_j for each shared item, and randomizes each of A's shares by adding r_j, while subtracting r_j for its own corresponding shares. B randomly permutes the randomized shares, and sends back to A. All the computations are done over the ciphertexts.

However, the BP protocol cannot be applied directly. In our protocol, each $F_i(x_j)$ is polynomially shared among $t+1$ parties in $\mathcal{P}'_i$, where at most t of them may be adversarial. Without loss of generality, assume P_i is the party that generates the random permutation (π). Then in order to randomly permute the shares of all the rest parties in $\mathcal{P}'_i$, there will be at least t invocations of the BP protocol between P_i and other parties in $\mathcal{P}'_i$, which is too computationally expensive.

Hence, in our scheme we propose an improvement that only requires *one* invocation of BP protocol between P_1 and each $P_i, 2 \le i \le N$. The main idea is to convert the $(t, t+1)$-shares of each secret $F_i(x_j)$ shared among $\mathcal{P}'_i$ into $(2,2)$-shares shared between P_1 and P_i. The conversion from $(t, t+1)$-share to $(2, t+1)$-share is fairly standard [34]. Each party P_l in $\mathcal{P}'_i$, holding $[F_i(x_j)]_l^{t,t+1}$, simply re-shares $[F_i(x_j)]_l^{t,t+1}$ among all other parties using $(2, t+1)$-SS. Then P_l interpolates all the received secondary shares to obtain $[F_i(x_j)]_l^{2,t+1}$.

To transform $(t, t+1)$-shares into $(2,2)$-shares, we use a trick in which all the parties in $\mathcal{P}'_i$ only re-share their $(t, t+1)$-shares of $F_i(x_j)$ to P_1 and P_i, so that *only* P_1 *and* P_i *together can reconstruct* s. Here the BP protocol is performed only once between them.

Finally, the matching phase enforces that P_1 only learns the best match's intersection set in the end. That is, if P_1 sends requests to multiple users, only P_{i^*} will reply. The idea is to let P_1 and each $P_i, 2 \le i \le N$ first commit their respective shares $[F_i(x_j)]_1^{2,2}$ and $[F_i(x_j)]_i^{2,2}$ to all other parties using broadcast (step 3.3). When P_1 sends a request to P_{i^*}, it also attaches a proof containing all the original shares of itself and each P_i. In this way, P_{i^*} can verify the correctness of those shares and derive whether itself is truly the best match. Note that, the $m_{1,l}$ values between P_1 and other parties P_l are learnt by P_{i^*}. The whole protocol is described in Fig. 3.

D. Towards Achieving PL-3

We consider a scheme that achieves PL-3. Continuing from the computation phase in the previous protocol, all parties could test if each $F_i(x_j)$ equals to 0 or not using the equality test protocol [33], and locally sum the shares of the results to obtain shares of $m_{1,i}$. After share conversion, $m_{1,i}$ is shared only between P_1 and P_i. Next they use a secure comparison protocol [33] to compare the $m_{1,i}$ values and finally learn the party with the largest m_{1,i^*} without knowing others' $m_{1,i}$. However, the communication cost of a single comparison is 279 + 5 invocations of the SS-multiplication protocol, which even exceeds the total number of SS-multiplication protocol invocations in the advanced scheme. Thus, construction of practical schemes achieving PL-3 is left as future work.

E. Personalizing Users' Privacy Levels

A user can choose her privacy level by telling other parties her choice in the beginning. For example, if a candidate P_i chooses PL-1, she can broadcast a message indicating "P_i selects PL-1". Then the parties in P_i's computing set and reconstruction sets will follow the basic scheme to compute the desired results for P_i. However, the initiator should always agree on the privacy level that each candidate proposes, since P_1 is at a position easier to conduct user profiling.

Inputs: basically the same as those in Fig. 2. In addition, P_1 sets $m_{1,i} = 0$ for each $2 \le i \le N$, and has a public key pk_1 and private key sk_1, and pk_1 is known by all others. P_i sets $m'_{1,i} = 0$.

1. Data Share Distribution: the same as in Fig. 2.

2. Computation: Basically the same as in Fig. 2, except each $P_l \in \mathcal{P}$ computes $[F_i(x_j)]_l = [R_{ij} f_i(x_j)]_l$, $\forall i, \mathcal{P}_i \ni P_l$, $1 \le j \le n$. For each $P_l \in \mathcal{P}'_i \subset \mathcal{P}_i$, its shares are also denoted as $[F_i(x_j)]_l^{t,t+1}$.

3. Reconstruction:

1) Share conversion: (executed by each $P_l \in \mathcal{P}$:)
   a) For each i that $\mathcal{P}'_i \ni P_l$, P_l generates a new 1-degree random polynomial $g_{ijl}(y)$ for each $1 \le j \le n$, and re-shares each $[F_i(x_j)]_l^{t,t+1}$ with P_1 and P_i, by giving them $g_{ijl}(1)$ and $g_{ijl}(i)$ (called *secondary share*), respectively, such that $g_{ijl}(0) = [F_i(x_j)]_l^{t,t+1}$. Note, P_1 and P_i send their secondary shares to each other.
   b) $P_l, l \ge 2$ computes $[F_l(x_j)]_l^{2,2} = \sum_{P_k \in \mathcal{P}'_i} \gamma_{ik} g_{ijk}(l)$ for each $1 \le j \le n$, where γ_{ik} is the Lagrangian coefficient as defined in Fig. 2. Similarly, P_1 computes $[F_l(x_j)]_1^{2,2}$ for each $2 \le l \le N$ and $1 \le j \le n$.
2) Blind and permute P_1's shares: P_1 and each $P_i \in \mathcal{P}, 2 \le i \le N$ involve in one BP-protocol where P_i generates permutation π and $\{r_j\}_{1 \le j \le n}$. (Note: $j' = \pi(j)$, P_i sets $[F_i(x_{j'})]_i^{2,2} \longleftarrow [F_i(x_{j'})]_i^{2,2} - r_{j'}$ while returning $E_{pk_1}([F_i(x_{j'})]_1^{2,2} + \alpha_i r_{j'}/\alpha_1)$, where α_1, α_i are Lagrange coefficients under $(2,2)$-sharing.)
3) Commit shares: P_1 computes $\{c_i = \mathcal{H}([F_i(x_{1'})]_1^{2,2} \| \cdots \| [F_i(x_{n'})]_1^{2,2})\}_{2 \le i \le N}$ and *broadcasts* to all parties; Also, each $P_i, 2 \le i \le N$ computes $c'_i = \mathcal{H}([F_i(x_{1'})]_i^{2,2} \| \cdots \| [F_i(x_{n'})]_i^{2,2})$ and *broadcasts* it.
4) Reconstruction: P_1 and each $P_i, 2 \le i \le N$ exchange their shares of $\{F_i(x_{j'})\}_{j' \in \{1, \cdots, n\}}$ and verify them against c'_i and c_i, respectively. If no verification fails, they compute $F_i(x_{j'}) = \alpha_1 [F_i(x_{j'})]_1^{2,2} + \alpha_i [F_i(x_{j'})]_i^{2,2}$ for each j'. Then P_1 count the number of $F_i(x_{j'}) = 0$, and set this number to $m_{1,i}$. The similar is done for each P_i.

4. Matching:

1) P_1 locally ranks the numbers $\{m_{1,i}\}_{2 \le i \le N}$ and finds the best match P_{i*}. Then P_1 sends to P_{i^*} a connection request with the following proofs: $\{[F_i(x_{j'})]_i^{2,2}\}_{2 \le i \le N, 1 \le j' \le n}$, $\{[F_i(x_{j'})]_1^{2,2}\}_{2 \le i \le N, 1 \le j' \le n}$.
2) P_{i^*} upon receiving the request, repeats the computations P_1 did in step 3.4, verifies the proof and tests whether itself has the largest m_{1,i^*} value. If not, then do nothing.
3) Otherwise, P_{i^*} and P_i will mutually find out their intersection set $\mathcal{I}_{1,i^*}$ by running one sub-protocol of basic scheme between them, and P_{i^*} decides whether to connect with P_1.

Fig. 3. The advanced scheme.

VI. Security Analysis

A. Analysis of the Basic Scheme

1) Security Under the HBC Model:

Lemma 2: In the computation phase, a passive adversary $\mathcal{A}$ controlling up to t parties cannot gain any additional information about each $f_i(x_j)$ other than what can be learnt from its own inputs and outputs.

Proof: Please refer to our technical report [32]. ■

Theorem 1: The basic scheme is unconditionally secure under a passive adversary controlling up to t parties.

Proof: Please refer to our technical report [32]. ■

Remark on security parameters. Due to unconditional security, it is enough as long as the field $\mathcal{F}_p$ can represent all the attributes in the attribute dictionary. Therefore, assume there are 1×10^6 attributes, we choose $= \log p = 24$. For the hash function, 160 bits should be enough.

2) Security Under Active Attacks: The following include attacks conducted by single party (P_1 or any $P_i, 2 \leq i \leq N$), or a coalition including up to t parties. Note that, we do not consider attacks that aim at disrupting the protocol. In addition, whenever we mention P_i, we mean it is not in the same coalition with P_1.

a) P_1 can use a large attribute set as input to find out as many attributes in P_i's set as possible. Note this attack can be prevented by limiting the size of all parties' input sets. For example, we can set this limit to 200.

b) All zero polynomial attack [7]. If the coefficients of P_i's polynomial is all zero, after reconstruction P_i gets every $x_j, 1 \leq j \leq n$ of P_1. Thus, we mandatorily set a_m to 1 that prevents this attack, the same approach as in [17].

c) A candidate P_i within a t-party coalition $\mathcal{A} \subset \mathcal{P}_i$ can try to inflate its intersection set $\mathcal{I}_{1,i}$ in the reconstruction phase. Especially, if there is no "commit shares" step, before P_i reveals its shares, as long as some $P_k \in \mathcal{P}_i$, $P_k \notin \mathcal{A}$ sends $[F_i(x_j)]_k$ to P_i, P_i can reconstruct $F_i(x_j)$ and the underlying t-degree polynomial. Thus P_i may use a different attribute x'_j that he guesses to be in P_1's set, and modify $[F_i(x_j)]_i$ such that $F_i(x_j) = x'_j$ to inflate $\mathcal{I}_{1,i}$.

The basic scheme prevents such kind of attack by letting each party commit to its shares before any reconstruction happens (step 3.1). Due to the pre-image resistance and second pre-image resistance of hash function, parties in $\mathcal{A}$ can neither know the shares of honest parties in advance, nor change their own shares after others reveal theirs, except with negligible probability. We do not consider $\mathcal{A}$ giving wrong shares in phases 1 and 2, because it cannot predict how the output will be and can only change it randomly.

B. Analysis of the Advanced Scheme

We assume computationally bounded adversaries here.

1) Security Under the HBC Model:

Theorem 2: Assuming the random permutation is truly random and the hash function is pre-image resistant, the advanced scheme satisfies the security properties of PL-2, under a passive adversary controlling up to t parties.

Proof: Please refer to our technical report [32]. ■

Remark. The advanced scheme does not achieve unconditional security because broadcasting the hash commitments in step 3.3 leaks information about other parties' shares to P_i. However, this is the only way the attacker can learn about other parties' $m_{1,i}$. Apart from this, the rest of the protocol is still unconditionally secure, i.e., we can still use a small p.

2) Security Under Active Attacks: a) We still consider P_i in a t-party coalition trying to inflate the intersection set size in the reconstruction phase. During share conversion, every honest party P_k (not in $\mathcal{A}$) will send $g_{ijk}(1)$ to party P_1. Since P_1 is not in $\mathcal{A}$, $\mathcal{A}$ can only collect at most t pairs of $g_{ijl}(1)$ and $g_{ijl}(i)$ ($P_l \in \mathcal{A}$). Lacking at least one secondary share, P_i cannot recover $F_i(x_j)$ in step 3.1 unless P_1 reveals its own shares $[F_i(x_j)]_1^{2,2}$ to P_i later.

In addition, during the BP protocol (step 3.2), P_1's shares are encrypted. By the semantic security of the homomorphic encryption, the shares of P_1 are hidden from P_i. Moreover, after BP and before actual reconstruction, P_1 and P_i's shares are committed first (step 3.3). Due to the one-wayness of hash commitments, P_i cannot learn $[F_i(x_j)]_1^{2,2}$ before all parties finish sending their commitments. Thus, P_i cannot change the outputs ($m_{1,i}$) in its favorable way in phase 3.

b) If a coalition of t parties that includes P_1 wants to inflate $m_{1,i}$ for some P_i, similarly, this attack will not succeed.

c) In the matching phase, P_1 may trick some party P_i to believe it is the best match and then obtain its intersection set. But each P_i will be able to detect whether itself is the best match by verifying the authenticity of shares and compute all $m_{1,i}$ values by itself. If not, it will not let P_1 learn $\mathcal{I}_{1,i}$.

d) *Input manipulation*. If P_1 inputs a few attributes (e.g., only one), it can know exactly which parties have those attributes whenever $\mathcal{S}_1 \subseteq \mathcal{S}_i$. To thwart this attack, we can set a lower limit to the number of query attributes.

C. Security of Personalized Privacy Levels

The adversary may try to deviate from executing the corresponding protocol chosen by a specific user. Suppose a user P_i chooses PL-2, $\mathcal{A}$ will try to execute the basic scheme instead, and then learn the intersection set between P_1 and P_i. However, $\mathcal{A}$ will not succeed, because it can only control t parties at most; as long as there is one honest party following P_i's own choice of protocol, $\mathcal{A}$ cannot obtain its desired results. Thus, each user P_i's privacy level is still enforced.

VII. Performance Evaluation

In this section, we analyze the complexity of our proposed schemes, carry out an extensive simulation study of the protocols' efficiency, and compare them with several state-of-the-art schemes in terms of security and efficiency.

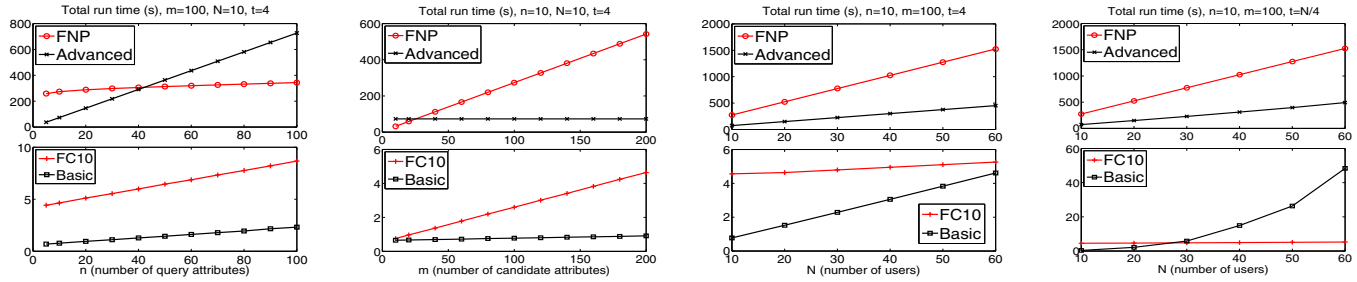
A. Complexity Analysis

The computational cost is evaluated using the number of modular multiplication and exponentiation operations, while the communication cost is calculated in terms of number of transmitted/received bits. We compare our basic and advanced schemes with existing schemes, FC10 (PSI) [10] and FNP (PCSI under HBC model) [7], respectively. Due to space limitations, we only present the results in Table II, where mul_1 stands for -bit multiplication operations, mul_2 is 1024-bit multiplication, exp_2 is 1024-bit exponentiation, and exp_3 is 2048-bit exponentiation. And we set $q = 1024$, $= 24$. The details of the analysis are presented in [32].

From Table II, the basic scheme's total computation complexity is much smaller than that of FC10's since $q \gg$, while that of the advanced scheme's is smaller than FNP's when n is relatively small w.r.t. m. Although the total communication costs may seem large in our schemes, they are on the same orders with the compared schemes in terms of m, n and N. The effect of the $O(t^2)$ factor is moderate unless t scales linearly with N, as we will see in the simulation results.

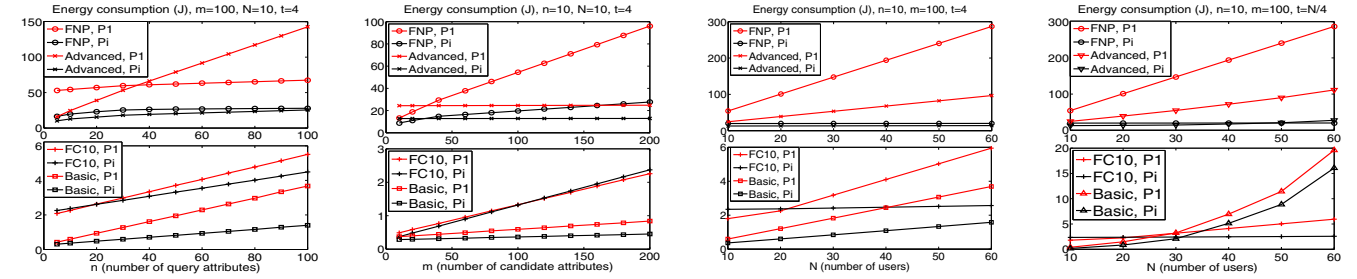
TABLE II
COMPARISON OF COMPLEXITY (q = 1024, = 24)

	Party	Basic scheme	FC10 (PSI) [10]	Advanced scheme	FNP (PCSI) [7]
Computation	P_1	$6nNt + mnN(1 + t\log N/\ )$ mul$_1$	$1.5nN$ mul$_2$	$3nN$ exp$_3$	$(2n + mN)$ exp$_3$
	P_i	$2mnt + 2(m + 6nt)t^2\log N/\ $ mul$_1$	$(m + n)$ exp$_2$	n exp$_3$	$m\log(\log n)$ exp$_3$
Comm. (TX)	P_1	$(mnN + 8nNt)$	$2nNq$	$(mnN + 8nNt)\ + 2nNq$	$2nq$
	P_i	$2t(m + 6nt)$	$(n + m)q$	$2t(m + 6nt)\ + 2nq$	$2mq$
Comm. (RX)	P_1	$[N(m + n) + 8nNt]$	$(n + m)Nq$	$N[m + (t + 3N)n]\ + 2nNq$	$2mNq$
	P_i	$[m(n + 2t) + 12nt^2]$	$2nq$	$[m(n + 2t) + 12nt^2]\ + 2nq$	$2nq$
Comm. total	All	$[mnN + tN(8n + 2m + 12nt)]$	$N(3n + m)q$	$[mnN + tN(8n + 2m + 12nt)]\ + 4nNq$	$2(n + mN)q$



(a) Change n. (b) Change m. (c) Change N, fix $t = 4$. (d) Change N, change $t = \lfloor N/4 \rfloor$.

Fig. 4. Total protocol run time. (Y-axis: protocol run time (s).)



(a) Change n. (b) Change m. (c) Change N, fix $t = 4$. (d) Change N, change $t = \lfloor N/4 \rfloor$.

Fig. 5. Energy consumption for P_1 and P_i. (Y-axis: energy consumption (J).)

B. Simulation Study

We implement our proposed and compared schemes in NS-2 [35]. Since the related crypto libraries for smartphone platforms are unavailable yet, assuming 400MHz CPU and WIFI, we simulate the protocols' communications and computations by telling the simulator the sizes and number of packets each party should send, fill each packet with dummy contents, and estimate the latency of each computation. We adopt energy consumption and total run time as unified evaluation metrics, which include both communication and computation costs. Please refer to [32] for more detailed simulated methodology.

Simulation Results The results are shown in Fig. 4 and Fig. 5. In Fig. 4 (a) and Fig. 5 (a), we fix the network size $N = 10$, maximum allowable number of colluders $t = 4$, number of profile attributes $m = 100$, and change number of query attributes n. It can be seen that, the basic scheme takes less than 1 second when $n < 100$. Both the total run time and energy consumption of it is much less than that of FC10's in this case and they all increase linearly with n. For the advanced scheme, the time and energy are smaller than those of FNP's when $n < 40$. This is mainly because the computation in the advanced scheme scales as $O(nN)$, which is $O(n+mN)$ for the FNP scheme. Nevertheless, for the profile matching application, in reality it is often the case that n is small.

On the other hand, from Fig. 4 (b) and Fig. 5 (b) in which n is fixed to 10 but m changes from 10 to 200, one can see that both the basic and the advanced schemes are more efficient than their counterparts, more importantly our schemes are hardly affected by m. This shows our schemes are more practical when the number of profile attributes is large, while the number of query attributes is relatively small.

Next, we change the total number of parties, and first fix $t = 4$. From Fig. 4 (c) and Fig. 5 (c), one can observe that the basic scheme's costs increase linearly with N. This is because its run time and energy costs are both dominated by communication which is linear in N when t is fixed. The FC10 scheme is much more computationally intensive in this case. For the advanced scheme and FNP, their costs are both dominated by computation rather than communication. However, the advanced scheme performs much better, since in FNP m is the dominating factor of computation rather than n in the advanced scheme, and $n < m$.

Finally, we scale t with N ($t = \lfloor N/4 \rfloor$) in Figs. 4 and 5 (d). Interestingly, the advanced scheme is still much more efficient than the FNP scheme, and it exhibits a super-linear

TABLE III
COMPARISON OF SECURITY

Schemes	Basic	Advanced	FC10 [10]	Agrawal [22]	FNP [7]	CANS09 [18]	CANS09 [18]
Category	PSI	PCSI	PSI	PSI/PCSI	PSI/PCSI	PSI	PCSI
Security	Uncond./HBC (PL-1)	PL-2/HBC	ROM/HBC	ROM/HBC	Standard/HBC	Uncond./HBC	Uncond./HBC
Resist active attacks	Yes	Yes	No	No	No	No	No

effect ($O(t^3)$ in overall communication) only when $N > 50$ for P_i's energy consumption. Meanwhile, the basic scheme suffers from this effect earlier than the advanced scheme (better than FC10 when $N < 30$), since it is dominated by communication.

The above demonstrates the advantage of our proposed schemes when n and N are both relatively small (in the order of tens), which is usual in mobile social networks. In the advanced scheme the energy consumption seldom exceeds 100J (equivalent to using WIFI for 5min), while that of the basic scheme is below 10J.

C. Further Comparison with Related Works

We now compare our schemes with more state-of-the-art schemes, in terms of security and efficiency. The Agrawal's scheme [22] represents schemes using commutative encryption, while the CANS'09 [18] schemes represent a category with unconditional security. For FC10 (PSI) and FNP (PCSI) schemes, they are designed solely under the HBC model, and can not prevent the malicious attacks mentioned in this paper, such as the intersection set inflation attack. For the CANS'09 PSI scheme, we modify it to fit our problem setting, and it differs from the basic scheme in that it does not aggregate the multiplications in the computation phase. The security comparison is summarized in Table III, and our schemes are better than or comparable in efficiency with those schemes.

VIII. CONCLUSION

In this paper, we for the first time formalize the problem of privacy-preserving profile matching in MSNs, and propose two concrete schemes that achieves increasing levels of user privacy preservation. Towards designing lightweight protocols, we utilize Shamir secret sharing as the main secure computation technique, while we propose additional enhancements to lower the proposed schemes' communication costs. Through extensive security analysis and simulation study, we show that 1) our schemes are not only secure under the HBC model, but also prevents certain active attacks; 2) our schemes are much more efficient than state-of-the-art ones in MSNs where the network size is in the order of tens, and when the number of query attributes is smaller than the number of profile attributes.

Acknowledgment This work was supported in part by the US National Science Foundation under grants CNS-0716306, CNS-0831628, and CNS-0746977. The authors would like to thank Prof. William Martin for his helpful discussions at the early stage of the protocols.

REFERENCES

[1] "Magnetu." [Online]. Available: http://magnetu.com
[2] J. Manweiler, R. Scudellari, and L. P. Cox, "Smile: encounter-based trust for mobile social services," in *ACM CCS '09*, 2009, pp. 246–255.
[3] "Familiar strangers." [Online]. Available: http://www.paulos.net/research/intel/familiarstranger/index.htm
[4] Z. Yang, B. Zhang, J. Dai, A. Champion, D. Xuan, and D. Li, "E-smalltalker: A distributed mobile system for social networking in physical proximity," in *IEEE ICDCS '10*, June. 2010.
[5] 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.
[6] M. Li, S. Yu, W. Lou, and K. Ren, "Group device pairing based secure sensor association and key management for body area networks," in *IEEE INFOCOM '10*, Mar. 14-19 2010, pp. 1–9.
[7] M. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *EUROCRYPT'04*. Springer-Verlag, 2004, pp. 1–19.
[8] A. C. Yao, "Protocols for secure computations," in *SFCS '82*, 1982, pp. 160–164.
[9] Q. Ye, H. Wang, and J. Pieprzyk, "Distributed private matching and set operations," in *ISPEC'08*, 2008, pp. 347–360.
[10] E. D. Cristofaro and G. Tsudik, "Practical private set intersection protocols with linear complexity," in *Financial Cryptography and Data Security '10*, 2010.
[11] L. Kissner and D. Song, "Privacy-preserving set operations," in *CRYPTO '05, LNCS*. Springer, 2005, pp. 241–257.
[12] Y. Qi and M. J. Atallah, "Efficient privacy-preserving k-nearest neighbor search," in *IEEE ICDCS '08*, 2008, pp. 311–319.
[13] M. Shaneck, Y. Kim, and V. Kumar, "Privacy preserving nearest neighbor search," in *Machine Learning in Cyber Trust*, 2009, pp. 247–276.
[14] U. Manber, "Finding similar files in a large file system," in *USENIX WINTER 1994 TECHNICAL CONFERENCE*, 1994, pp. 1–10.
[15] Y. Sang, H. Shen, and N. Xiong, "Efficient protocols for privacy preserving matching against distributed datasets," in *ICICS '06*. Springer - Verlag, 2006, pp. 210–227.
[16] D. Dachman-Soled, T. Malkin, M. Raykova, and M. Yung, "Efficient robust private set intersection," in *ACNS '09*, 2009, pp. 125–142.
[17] R. Li and C. Wu, "An unconditionally secure protocol for multi-party set intersection," in *ACNS '07*, 2007, pp. 226–236.
[18] G. S. Narayanan, T. Aishwarya, A. Agrawal, A. Patra, A. Choudhary, and C. P. Rangan, "Multi party distributed private matching, set disjointness and cardinality of set intersection with information theoretic security," in *CANS '09*. Springer - Verlag, Dec. 2009, pp. 21–40.
[19] M. Shaneck and Y. Kim, "Efficient cryptographic primitives for private data mining," in *HICSS '10*, 2010, pp. 1–9.
[20] C. Hazay and Y. Lindell, "Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries," in *TCC'08*, 2008, pp. 155–175.
[21] S. Jarecki and X. Liu, "Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection," in *TCC '09*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 577–594.
[22] R. Agrawal, A. Evfimievski, and R. Srikant, "Information sharing across private databases," in *SIGMOD '03*. New York, NY, USA: ACM, 2003, pp. 86–97.
[23] J. Vaidya and C. Clifton, "Secure set intersection cardinality with application to association rule mining," *J. Comput. Secur.*, vol. 13, no. 4, pp. 593–622, 2005.
[24] M. von Arb, M. Bader, M. Kuhn, and R. Wattenhofer, "Veneta: Serverless friend-of-friend detection in mobile social networking," in *IEEE WIMOB '08*, Oct. 2008, pp. 184 –189.
[25] Y. Li, J. Tygar, and J. Hellerstein, "Private matching," in *Computer Security in the 21st Century*, 2005, pp. 25–50.
[26] F. Emekci, D. Agrawal, A. E. Abbadi, and A. Gulbeden, "Privacy preserving query processing using third parties," in *ICDE '06*, 2006, p. 27.
[27] S. S. M. Chow, J.-H. Lee, and L. Subramanian, "Two-party computation model for privacy-preserving queries over distributed databases," in *NDSS*, 2009.
[28] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.
[29] J. C. Benaloh, "Secret sharing homomorphisms: Keeping shares of a secret secret (extended abstract)," in *CRYPTO '86*, 1987, pp. 251–260.
[30] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *ACM STOC '88*, 1988, pp. 1–10.
[31] R. Gennaro, M. O. Rabin, and T. Rabin, "Simplified vss and fast-track multiparty computations with applications to threshold cryptography," in *ACM PODC '98*, 1998, pp. 101–111.
[32] M. Li, N. Cao, S. Yu, and W. Lou, "Findu: Privacy-preserving personal profile matching in mobile social networks." [Online]. Available: http://ece.wpi.edu/~mingli/findu.pdf
[33] T. Nishide and K. Ohta, "Multiparty computation for interval, equality, and comparison without bit-decomposition protocol," in *PKC'07*, pp. 343–360.
[34] E. Kiltz, "Unconditionally secure constant round multi-party computation for equality, comparison, bits and exponentiation," in *TCC '05*. Springer, 2005.
[35] "Ns2," *http://www.isi.edu/nsnam/ns*.