SPFM: Scalable and Privacy-preserving Friend Matching in Mobile Cloud

Mengyuan Li, Ruan Na*, QiYang Qian, Haojin Zhu, Xiaohui Liang, Le Yu

***Abstract*—Profile (e.g., contact list, interest, mobility) matching is more than important for fostering the wide use of mobile social networks. The social networks such as Facebook, Line or Wechat recommend the friends for the users based on users personal data such as common contact list or mobility traces. However, outsourcing users' personal information to the cloud for friend matching will raise a serious privacy concern due to the potential risk of data abusing. In this study, we propose a novel Scalable and Privacy-preserving Friend Matching protocol, or SPFM in short, which aims to provide a scalable friend matching and recommendation solutions without revealing the users personal data to the cloud. Different from the previous works which involves multiple rounds of protocols, SPFM presents a scalable solution which can prevent honest-but-curious mobile cloud from obtaining the original data and support the friend matching of multiple users simultaneously. We give detailed feasibility and security analysis on SPFM and its accuracy and security have been well demonstrated via extensive simulations. The result show that our scheme works even better when original data is large.**

***Index Terms*—Friend matching, Privacy preserving, Cloud security, XOR.**

I. Introduction

Along with the popularity of the smartphone and ubiquitous wireless access, mobile clouds are becoming an inseparable part of our life. People use different clouds provided by different applications to store their private data such as contacts, mail address lists or bank accounts while mobile applications use these data to provide a wide range of service such as friend recommendation. Profile (e.g., contact list, interest, mobility) matching is more than important for fostering the wide use of mobile social networks because recommending the individuals of the common contacts list/similar interests is always the first step for any social networking. The social networks such as Facebook, Line or Wechat recommend the friends for the users based on contact list or mobility traces.

The existing mobile social network systems pay little heed to the privacy concerns associated with friend matching and recommendation based on users' personal information. For example, in Facebook, it provides the feature of People You May Know, which recommends the friends based on the education information, the contact lists obtained from users' smartphone, and other users' personal information. However, outsoucing users' personal information to the cloud for friend matching will raise a serious privacy concern. The potential risk for personal data to be abused for economic and social discrimination, hidden influence and manipulation, is alarming [1], which has been well demonstrated by the facts such as iCloud Data Breach[2].Existing researches show that loss of privacy can expose users to unwanted advertisement and spams/scams, cause social reputation or economic damage, and make them victims of blackmail or even physical violence[3].

Recently, there are quite a few proposals for private profile matching, which allow two users to compare their personal profiles without revealing private information to each other[4–7]. In a typical private profile matching scheme, the personal profile of a user consists of multiple attributes chosen from a public set of attributes (e.g., various interests[4], disease symptoms[8], or friends[9] etc.). The private profile matching problem could then be converted into Private Set Intersection (PSI)[10, 11], or Private Set Intersection Cardinality (PSI-CA)[12, 13]. However, we argue that the existing works may fail to work in practice due to the following two reasons. Firstly, the best practice in industry for friends recommendation is a multiple-users matching problem rather than a two-party matching problem. Some pre-share parameters between users are more likely to leak. Secondly, most of the existing works involve multiple rounds of protocols, which will suffer from a serious performance challenge.

In this study, we propose a novel Scalable and Privacy-preserving Friend Matching protocol, or SPFM in short, which aims to provide a scalable friend matching and recommendation solutions without revealing the users personal data to the cloud. Our basic motivation is that each user obfuscates every bit of the original personal data (e.g., contact list) before uploading by performing XOR operations with a masking sequence which is generated with a certain probability. Our design can ensure that the same data maintain a statistical similarity after obfuscation while different data can be statistically classified without leaking the original data. The contributions of this work are summarized as follows:

- We propose a Scalable and Privacy-preserving Friend Matching scheme (SPFM) to prevent privacy leakage in friend matching and recommendation system.
- We provide a detailed feasibility and security analysis as well as the discussion of correctness, True-Negative rate and True-Positive rate.
- Extensive evaluations have been performed on SPFM to demonstrate the feasibility and security. The result show that our scheme works even better when original data is large.

This paper is organized as follows. In section II, we give the related work. In section III, we introduce our system model and corresponding adversary models. In section IV, we describe system implement in detail. A thorough feasibility

*Corresponding author, Email: naruan@cs.sjtu.edu.cn and security analysis are demonstrated in section V. In Section VI we give the experiment performance and in section VII, we conclude this paper.

II. Related work

Agrawal et al. [14] first proposes Privacy-Preserving Data Mining (PPDM), which tries to perturb individual records in an aggregated database. However, this paper mainly focuses on statistical information of unordered databases and doesn't rise to the level of data matching.

There are lots of relevant work can complete the cloud data matching, such as deterministic encryption[15, 16] and order-preserving encryption[17, 18]. However, these work also need negotiation between users, which cannot be applied in our scenario since conspiracy attack between malicious users and cloud server may occur.

With the interaction among users in the existing applications received more and more attention, a lot of research on the issue of privacy data matching has been raised recently.

Li et al. [4] applies additive homomorphic encryption in privacy preserving in a scenario with many intermediate computing parties. Narayanan et al. [19] and Dong et al. [5] computes social proximity to discover potential friends by leveraging both homomorphic cryptography and obfuscation, which is more efficient.

More privacy-preserving computation schemes based on encryption, obfuscation or cooperative computing such as [20–27] require multiple exchanges between participants,which is not suitable for the cases where users are not connected to each other or is not connected or number of users is relatively large.

III. Problem formulation

In this section, we will describe our problem in detail. We will first introduce application scenarios and system model. Then we present assumptions and adversary models of our scenario.

A. Application scenario

Nowadays, it is observed that many mobile social networks (e.g., Facebook, Wechat, Line) have provided the functionality of friend recommendation, which recommends the new friends to a user based on his contact list, education, mobility and other factors. To achieve this service, the various social networks need to collect the personal data of the users. Take Facebook's "People You May Know" as an example. It is stated by Facebook that it shows the potential friends "based on mutual friends, work and education information, networks you're part of, contacts you've imported and many other factors". For some of user personal data such as contact lists, it relies on apps installed on smart phones to collect the data and upload the data to the cloud. The cloud can determine if two users are friends by checking their common attributes such as the same school, common friends or similar mobility patterns.

However, the sensitive data uploaded to the cloud may face the risk of leaking users' sensitive data and compromising users' privacy. In this work, we consider a privacy preserving friend matching scenario, in which the users' data will be obfuscated before uploading to the cloud. Thus in the friend matching process, the server has no idea of the original sensitive data but it can still perform the friend matching and recommendation service.

B. System definition

In Section VI our system will be illustrated to be a profile matching system that can be widely used in many scenarios. In order to present the universal scene of our matching scheme, we need to define some concepts first.

Definition 1. $(Potential\ Matching\ Probability)$. *A Potential Matching Probability* (P_m) *is the probability of two individuals' original data are the same on the condition of they are in the same group.*

Universal scene of our system can be described as follows. Each of K parties has an N-bit sequence. Some of these uncorrelated sequences may have the same value and the unknown probability is called P_m. Now these K parties need a notary to judge whom of these people may have the same sequence and they do not want the notary public to get their own precise sequence.

Our system has been proven in section V to be effective when P_m is in an appropriate range, usually larger than 0.00001. In reality, the value range of P_m may be large and in order to ensure P_m to be in a suitable range, we define Data Tag to make our system able to be applied in wider scene.

Definition 2. $(Data\ Tag)$. *Data Tag is an identifier of certain original private data, which has only limited information of the original data.*

A Data Tag can be meaningful or meaningless labels. It aims at improving P_m by dividing the original large group into series of small groups in the case of almost no leakage of user's privacy. For a certain system, the generation of Data Tag need to satisfy a certain rule. For example, to determine whether some users have common friend in contacts, system can use contacts' name abbreviations as labeled data, like AT for Alan Turing. In this paper, we define P_T as the probability of two individual data are the same when these 2 individuals share a same Data Tag.

In this paper, we have chosen such a mobile cloud scenario. Our Mobile Cloud Storage System (MCSS) consists of a cloud server denoted as C and some users (or parties) denoted as $\mathbb{U} = \{u_1, ..., u_K\}$. These users only establish a connection with cloud server C, and don't establish contact or share any information with each other.

These users upload their obfuscated data to cloud sever after masking the original private data by using a masking sequence generation probability preset by the cloud server before. Cloud server C launches the data match processing function and in this paper its goal is to find whether there are some users owning the same original data.

C. Assumptions and Adversary models

In our MCSS, cloud server also determines two users are real friends by matching how many friends they have in common. However, instead of fetching their exact contact list, MCSS tries to match using low-level tags (usually insensitive information, in this paper, abbreviation of linkmen's name as an example) and corresponding obfuscated data e.g. phone number, mail address after obfuscating process.

We assume two real friends have a number of common friends and they store some common friends in contact list with the same name and telephone number.

There are two main adversary models in this paper. One is external attacker, who will try to get your privacy through hacking your cloud account. In this case, what attacker can get is usually relatively limited, generally only obfuscated data and Data Tags.

Another adversary model which is also mainly considered in this paper is honest-but-curious server (HBCS), i.e. a cloud server which will try to find out your privacy and record it but honestly follow the whole protocol. The data a cloud server can get is huge and a small loophole may cause hundreds of millions privacy leakage.

IV. System Design

In this section, we describe the system design in detail. In our system, there are four steps in process of applying SPFM in MCSS. Fig. 2 provides an overview of the work-flow of our approach. Algorithm 1 explains how SPFM works and we will describe each step in detail.

A. Parameters Setup

In the first step, the system needs to set up *masking generation probability* p_k. p_k is a value greater than 0.5, and p_k will determine the masking degree. The more p_k is close to 0.5, the greater the degree of disturbance and the privacy-protect ability of the whole system is. However, this will reduce the data matching accuracy as well. In practical applications, the system will determine a p_k by different needs of security and privacy. The *masking generation probability* is a common knowledge for the cloud and all users.

B. Data Masking

The second step is performed on each user's device. In this step, each user will use *masking generation probability* p_k to deal with private data needed to be uploaded. For each original sequence, a masking sequence of a same length is needed to obfuscate the original sequence. In a binary case, for each bit of a masking sequence, it has probability p_k to be a 0,$1-p_k$ to be a 1.

For example, for an N-bit binary original sequence represented by $B = \{b_1, b_2, b_3, ..., b_N\}$. The user firstly repeat random process N times under this probability p_k and we will get a binary masking sequence of length N, which is represented by $O = \{o_1, o_2, o_3, ..., o_N\}$. Then do XOR of the original sequence and the masking sequence to generate a N-bit sequence called obfuscated data. Obfuscated sequence and the process are showed as follows.

$$C = \{c_1, c_2, c_3, ..., c_N\}$$

$$c_i = b_i \oplus o_i, i = 1, 2, ..., N$$

After all sequences are obfuscated, user need to upload these obfuscated sequences C and the corresponding Data Tag to the cloud. As for those masking sequences, users can store them in other cloud or other devices, which will be used when restore original data from the cloud.

Algorithm 1 SPFM $(\mathbb{U}, Cloud, p_k)$

```
 1: $\mathbb{U}$,Cloud$\leftarrow p_k$.
 2: for each $u_i \in \mathbb{U}$ do
 3:     $\mathbb{D}_i \leftarrow ListOfContact(u_i)$.
 4:     $\mathbb{T}_i \leftarrow AbbreviationOfName(\mathbb{D}_i)$.
 5:     $\mathbb{B}_i \leftarrow PhoneNumber(\mathbb{D}_i)$.
 6:     for each $B_{ix} \in \mathbb{B}_i$ do
 7:         $O_x \leftarrow MaskGeneration(p_k)$.
 8:         $C_{ix} = B_{ix} \oplus O_x$.
 9:     end for
10:     $u_i$ upload $\mathbb{C}_i$ with $\mathbb{T}_i$ to Cloud .
11: end for
12: $u_i$ put forward matching demand.
13: Cloud carries out the following operations.
14: for $u_j (i \neq j) \in \mathbb{U}$ do
15:     for each $t_x \in \mathbb{T}_i$ do
16:         for each $t_y \in \mathbb{T}_j$ do
17:             if $t_x == t_y$ then
18:                 $MatchResult \leftarrow MatchingPhase(C_{ix}, C_{jy})$.
19:             else
20:                 $MatchResult = No \leftarrow (C_{ix}, C_{jy})$.
21:             end if
22:         end for
23:     end for
24: end for
```

C. Profile Matching

In the third step, we first introduce two definitions Threshold and Matching Ambiguity in this step to adjust the matching accuracy, we use the threshold n_{th} to describe matching criteria, and the matching ambiguity K_{th} to be the ratio of the threshold and the original data's length.

Definition 3. $(Threshold)$. *Threshold* (n_{th}) *is the minimum number of same bits in two scrambled sequences (e.g. C and* $C^{'}$*) from 2 users when server judges these two users have the same original sequences* $(B = B^{'})$*. For an* N*-bit binary private sequence, scrambled sequence is also* N*-bit. Cloud server will do XOR of two scrambled sequences to judge how many bits they differ. (The XOR result of two different bits is 1 while the XOR result of two same bits is 0.) The result of XOR process indicates the deviation of* C *and* $C^{'}$*. We use* $D = \{d_1, d_2, d_3, ..., d_N\}$ *to represent it.*

$$d_i = c_i \oplus c_i^{'}, i = 1, 2, ..., N$$

Threshold n_{th} *is the minimum number of 0 in* D *when server judge* C *and* $C^{'}$ *having the same original sequences which means* $B = B^{'}$.

Definition 4. $(Matching\ ambiguity)$. *The ratio of Threshold and the original data's length is defined as Matching*

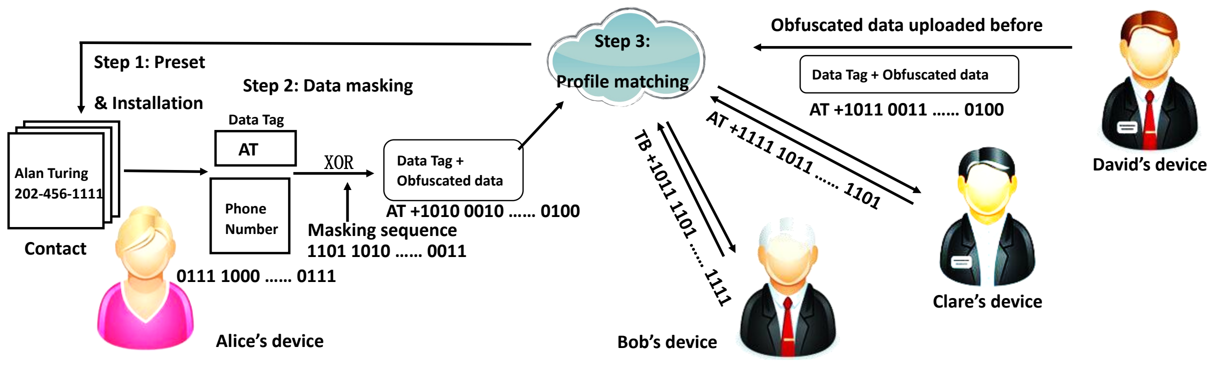


Fig. 1: System overview

Ambiguity (K_{th}). *For an* N*-bit binary sequence, the ratio of Threshold and N is defined as the Matching Ambiguity.*

$$K_{th} = \frac{n_{th}}{N}$$

The increase of the Matching Ambiguity will significantly increase the probability of T-P at the expense of some accuracy. In the actual development, system should select an appropriate Matching Ambiguity according to p_k *and* N*.*

When a user requests for matching, the server will use the obfuscated sequence and Data Tag to match. Now suppose that the server tries to find out whether two users are real friends in reality. The key is to find out how many common friends they have in contacts. The server first does a traversal through two users' Data Tags and fine all of the same Data Tags. For one of these same Data Tag, do XOR operation of their obfuscated sequences. If the number of 0 in the XOR results is more than n_{th}, then server considers that the original data of these two obfuscated sequences are the same. In the application scenario of this paper, server will consider the telephone number stored in two users under this Data Tag are the same. In other words, the Data Tag represents a common friend of these two users.

After a thorough traversal of all the Data Tags, server will get the number of common friends between these two users, which can be further used as a judge basis whether the two users are real friends in reality.

V. Protocol Analysis

A. Notations and Technical Preliminaries

Notations. Please refer to Table I. B, B', O, O', C, C', C^i, D are all binary sequence which can expressed like $B = \{b_1, b_2, ..., b_N\}$.

Preliminaries. We assume that each bit in a series of original private data is independent and has equal probability to be '0' or '1'. Then

$$P(S_b = n_b) = \binom{N}{n_b}(\frac{1}{2})^N$$

B. Priori Probability Analysis

We define priori probability as the probability of matching($S_d \geq n_{th}$) on condition of certain n_b. So it can be expressed as $P(S_d \geq n_{th}|S_b = n_b)$. Theorem 1 gives its solution.

TABLE I: Notation

N	Sequence length
p_k	Probability to generate O, O'
p_1	Priori probability for 1 bit
M	Number of different Cs of a same B
B, B'	N-bit original binary sequence
O, O'	N-bit masking sequence
C, C', C^i	N-bit obfuscated sequence
D	XOR result of C, C'
S_b	Number of 0s in B and B'
S_d	Number of 0s in D
n_{th}	Threshold for S_d to judge matching
K_{th}	Ratio of n_th and N
P_{T-P}	True-Positive probability
P_{T-N}	True-Negative probability
P_T	Probability of $B = B'$ when Data Tag matching
$P_g(n)$	Probability of guesssing an n-bit B

Theorem 1. *Given* n_b*, threshold* n_{th}*, sequence length* N *and masking probability* p_k*, the priori probability is:*

$$P(S_d \geq n_{th}|S_b = n_b) = \sum_{n_d=n_{th}}^{N} \sum_{n_0=max\{0,n_d+n_b-N\}}^{min\{n_d,n_b\}} G \tag{1}$$

where $G = \binom{n_b}{n_0}\binom{N-n_b}{n_d-n_0}p_1^{N-n_b-n_d+2n_0}(1-p_1)^{n_b+n_d-2n_0}$.

Proof: For each $i \in [0, N]$, we have

$$P(d_i = 0|b_i = b_i') = P(d_i \neq 0|b_i \neq b_i') = p_k^2 + (1-p_k)^2$$

$$P(d_i \neq 0|b_i = b_i') = P(d_i = 0|b_i \neq b_i') = 2p_k(1-p_k)$$

Here we define

$$p_1 = p_k^2 + (1-p_k)^2 \in (0.5, 1)$$

Then consider the case of N bits. Given n_d and n_b, we have

$$P(S_d = n_d|S_b = n_b) = \sum_{n_0=max\{0,n_d+n_b-N\}}^{min\{n_d,n_b\}} G$$

where $G = \binom{n_b}{n_0}\binom{N-n_b}{n_d-n_0}p_1^{N-n_b-n_d+2n_0}(1-p_1)^{n_b+n_d-2n_0}$.

Finally, to calculate the priori probability we add all probability of satisfying n_d up. So we get

$$P(S_d \geq n_{th}|S_b = n_b) = \sum_{n_1=n_{th}}^{N} P(S_d = n_1|S_b = n_b)$$

■

Because the priori probability is the sum of all $P(S_d = n_d|S_b = n_b)$ of all n_d satisfying $n_d \geq n_{th}$, it is negatively correlates with threshold n_{th}.

Consider Data Tag. According to the above derivation, once S_b is decided, the probability of $S_d \geq n_{th}$ is decided. It means the matching of Data Tags has no effect on the priori probability. Using '$T = T^{'}$' to represent the matching of Data Tags, and '$T \neq T^{'}$' on the contrary. Then we have

$$\begin{aligned} P(S_d \geq n_{th}|S_b = n_b, T = T^{'}) \\ &= P(S_d \geq n_{th}|S_b = n_b, T \neq T^{'}) \\ &= P(S_d \geq n_{th}|S_b = n_b) \end{aligned} \tag{2}$$

C. True-Positive and True-Negative Analysis

True-Positive(T-P for short) means matching($S_d \geq n_{th}$) on condition of $B = B^{'}$(or $S_b = N$). Its probability is defined as P_{T-P} which can be expressed as follows.

$$\begin{aligned} P_{T-P} &= P(S_d \geq n_{th}|S_b = N) \\ &= \sum_{n_d=n_{th}}^{N} \tbinom{N}{n_d} p_1^{n_d}(1-p_1)^{N-n_d} \end{aligned} \tag{3}$$

True-Negative(T-N for short) means not matching($S_d < n_{th}$) on condition of $B \neq B^{'}$(or $S_b \neq N$). Its probability is defined as P_{F-N} which can be expressed as follows.

$$P_{T-N} = \frac{\sum_{n_b=0}^{N-1} P(S_d < n_{th}|S_b = n_b)P(S_b = n_b)}{P(S_b \neq N)}$$

D. Correctness Analysis

The correctness (posterior probability) indicates the probability of $B = B^{'}$ on condition of matching($S_d \geq n_{th}$) and Data Tag matching. So it can be expressed as follows.

$$P(S_b = N|S_d \geq n_{th}, T = T^{'})$$

Theorem 2. *Given sequence length N, masking probability p_k, threshold n_{th} and P_T, the correctness is:*

$$\begin{aligned} &P(S_b = N|S_d \geq n_{th}, T = T^{'}) = \\ &\frac{P_{T-P}P_T}{\frac{1-P_T}{2^N-1}\sum_{n_b=0}^{N-1}\tbinom{N}{n}P(S_d \geq n_{th}|S_b = n_b) + P_{T-P}P_T} \end{aligned} \tag{4}$$

where $P_{T-P} = P(S_d \geq n_{th}|S_b = N)$

Proof: To make it simple and clear to derivate, here we define E_B to represent the event of $B = B^{'}$ $(S_b = N)$, $E_{B(n)}$ to represent the event of $S_b = n$, E_M to represent the event of matching ($S_d \geq n_{th}$) and E_T to represent the event of Data Tag matching ($T = T^{'}$). Then the correctness can be expressed as follows.

$$\begin{aligned} P(E_B|E_M \cap E_T) &= \frac{P(E_B \cap E_M \cap E_T)}{P(E_M \cap E_T)} \\ &= \frac{P(E_M|E_B \cap E_T)P(E_B|E_T)}{P(E_M|E_T)} \end{aligned} \tag{5}$$

where

$$P(E_M|E_T) = \sum_{n=0}^{N} P(E_M|E_{B(n)} \cap E_T)P(E_{B(n)}|E_T)$$

And we define $P_T = P(E_B|E_T)$ as a parameter decided by the type of Data Tag. Then the posterior probability can be simplified as follows.

$$P(E_B|E_M \cap E_T) = \frac{P(E_M|E_B)P_T}{\sum_{n=0}^{N} P(E_M|E_{B(n)})P(E_{B(n)}|E_T)} \tag{6}$$

When $n \neq N$,

$$\begin{aligned} P(E_{B(n)}|E_T) &= P(E_{B(n)}|S_b \neq N)P(S_b \neq N|E_T) \\ &= \frac{\tbinom{N}{n}}{2^N-1}(1-P_T) \end{aligned} \tag{7}$$

Therefore,

$$\begin{aligned} &P(E_B|E_M \cap E_T) = \\ &\frac{P(E_M|E_B)P_T}{\frac{1-P_T}{2^N-1}\sum_{n=0}^{N-1}\tbinom{N}{n}P(E_M|E_{B(n)}) + P(E_M|E_B)P_T} \end{aligned} \tag{8}$$

■

E. Security Analysis

When our SPFM is facing the external attacker, the attacker can get a share of obfuscated data by hacking user's cloud account. When our SPFM is under honest-but-curious attack model, the attack can get some series of obfuscated data of the same private data.

Assume there are M series of obfuscated data attacker can get from M clients. These obfuscated data are represented by $C^1 = \{c_1^1, c_2^1, ..., c_N^1\}, C^2 = \{c_1^2, c_2^2, ..., c_N^2\}, ..., C^M = \{c_1^M, c_2^M, ..., c_N^M\}$. And the cloud server confirms that the corresponding original private data are all the same, which is $B = \{b_1, b_2, ..., b_N\}$. So the cloud server can use these series together to guess the original data.

For each b_i, the cloud server can calculate the number of 1s and 0s . Since 1 and 0 are symmetrical on math, commonly the server can use a guessing scheme that guess the number whose amount is more than the other.

Let n_0 represent the number of 0s in $\{c_i^1|i = 1, 2, ..., M\}$ and n_1 represent the number of 1s . So when $n_0 > n_1$ guess 0, when $n_0 < n_1$ guess 1 and when $n_0 = n_1$ guess 0 or 1 at random. Then the priori probability can be expressed as follow.

$$P(n_0|b_i = 0) = \tbinom{M}{n_0}p_k^{n_0}(1-p_k)^{M-n_0}$$

$$P(n_1|b_i = 1) = \tbinom{M}{n_1}p_k^{n_1}(1-p_k)^{M-n_1}$$

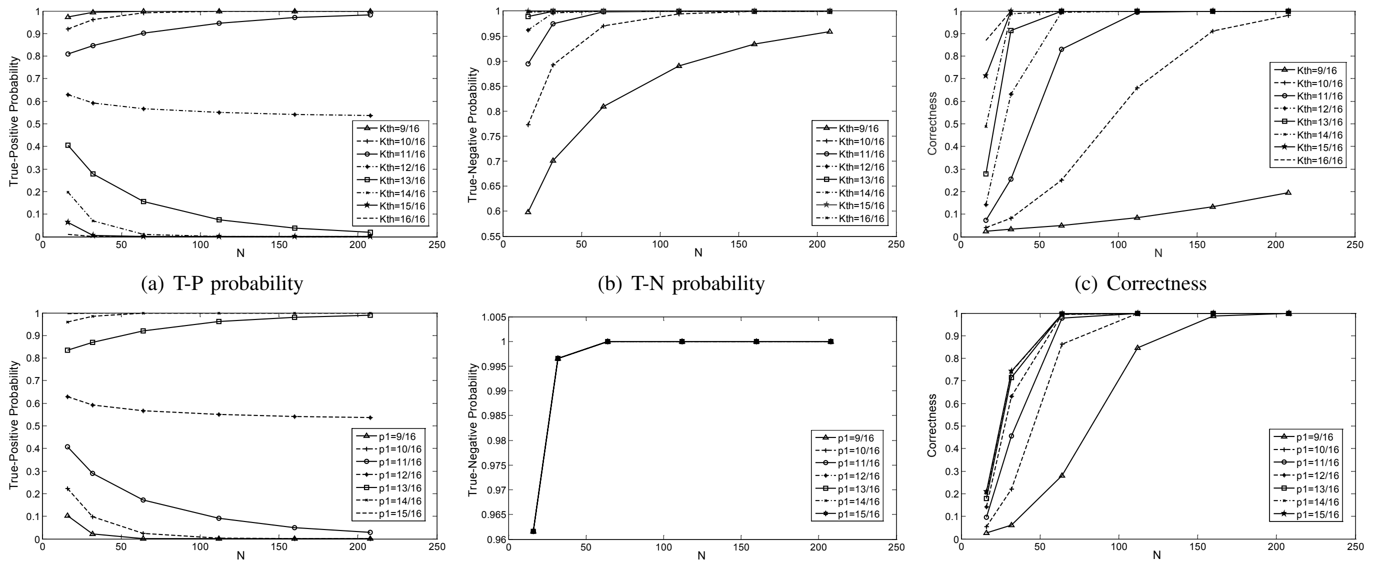
$$P(guess0|b_i = 0) = \sum_{n_0>n_1} P(n_0|b_i = 0) + \frac{1}{2}P(n_0 = n_1|b_i = 0)$$

$$P(guess1|b_i = 1) = \sum_{n_1>n_0} P(n_1|b_i = 1) + \frac{1}{2}P(n_0 = n_1|b_i = 1)$$

where $n_0 + n_1 = M$.

Using Bayes' Theorem and Law of Total Probability, the correctness of guessing can be expressed as follow.

$$\begin{aligned} &P(b_i = 0|guess0) = \\ &\frac{P(guess0|b_i = 0)P(b_i = 0)}{P(guess0|b_i = 1)P(b_i = 1) + P(guess0|b_i = 0)P(b_i = 0)} \end{aligned} \tag{9}$$



(a) T-P probability (b) T-N probability (c) Correctness

(d) T-P probability (e) T-N probability (f) Correctness

Fig. 2: T-P probability, T-N probability and Correctness for N, K_{th} and p_1

Since $P(b_i = 0) = P(b_i = 1) = \frac{1}{2}$,

$$P(b_i = 0|guess0) = \frac{P(guess0|b_i = 0)}{P(guess0|b_i = 1) + P(guess0|b_i = 0)}$$

Since the symmetry between 0 and 1, we have $1 - P(guess0|b_i = 1) = P(guess1|b_i = 1) = P(guess0|b_i = 0)$. So the correctness can be simplified as follow.

$$P(b_i = 0|guess0) = P(guess0|b_i = 0)$$

the same as the case of guessing 1.

Thus the total correctness for 1 bit is,

$$\begin{aligned} P_g(1) &= P(b_i = 0 \cap guess0 \cup b_i = 1 \cap guess1) \\ &= P(b_i = 0|guess0)P(guess0) \\ &\quad + P(b_i = 1|guess1)P(guess1) \\ &= P(b_i = 0|guess0) = P(b_i = 1|guess1) \\ &= P(guess0|b_i = 0) = P(guess1|b_i = 1) \end{aligned} \tag{10}$$

For the whole N-bits series B, the probability of guessing is,

$$P_g(N) = P_g(1)^N$$

VI. Evaluation

In this section, we carry out a simulation study of the feasibility, accuracy and security when using SPFM. The evaluation results show a good performance of SPFM in different data length N.

In SPFM, output variables are True-Positive probability P_{T-P}, True-Negative probability P_{T-N}, Correctness and guessing probability $P_g(N)$ (a measure of insecurity). Arguments are p_k(or p_1), N, K_{th}(or n_{th}), P_T and M.

A. Complexity Analysis

To perform one time of DMS, we need to do N times of subtraction, N-1 times of addition, one time of division and one time of comparison. Obviously, the complexity of DMS is $O(N)$. So DMS is a very time-saving and energy-saving scheme.

B. T-P and T-N probability

Fig.2(a) and Fig.2(d) show the correlations between P_{T-P} and arguments N, K_{th} and p_1. Clearly, P_{T-P} positively correlates with p_1 but negatively correlates with K_{th}. N has different influence on P_{T-P} according to other 2 arguments. When $K_{th} < p_1$, P_{T-P} positively correlates with N and converges to 1 quickly, otherwise P_{T-P} negatively correlates with N and converges to 0 quickly. This characteristic of P_{T-P} is important in following trade-off analysis.

Fig.2(b) and Fig.2(e) show the case of P_{T-N}. P_{T-N} positively correlates with K_{th} but has no correlation with p_1. And on every condition, P_{T-N} can always converge to 1 by increasing N. T-N probability is relatively much easier to converge to 1 compared with T-P probability, so we will firstly ignore it when doing following analysis and check it at last.

C. Correctness

Fig.2(c) to Fig.2(f) show the case of correctness. Clearly, the correctness positively correlates with all 3 arguments. And the correctness can converge to 1 by increasing any argument.

Argument P_T also has positive influence on the correctness. Referring to the semi-Log figure Fig.3, we can see that given other arguments, the correctness positively correlates with P_T and converges to 1. And it has a great gradient getting close to 1 while it changes little near 1. This means there is a range where the correctness is close to 1 and P_T has little influence on it. When increasing N or K_{th}, this range gets wider very quickly. For example, when $N = 128, n_{th} = 112$, the correctness stays nearly 1 when P_T is as small as 10^{-6}. This characteristic gives a loose requirement when selecting Data Tag for SPFM.

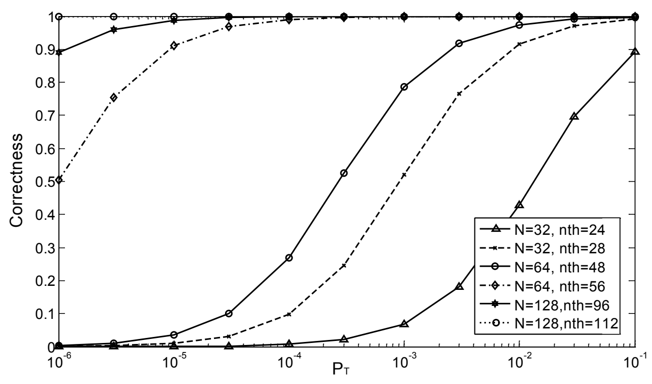


Fig. 3: T-P, T-N, Correctness for N

D. Security

We use the probability of guessing the whole N-bit sequence ($P_g(N)$) correctly as a measure of insecurity. So we suppose $P_g(N)$ to be relatively small.

Fig.4 shows the probability of guessing 1 bit ($P_g(1)$) for different M and p_k. Clearly, $P_g(1)$ positively correlates with both M and p_k. According to M, N and the security demand, certain p_k can be determined. To achieve larger P_{T-P} and correctness, p_k should be as large as possible, which may bring privacy risk. So there needs a trade-off about security and accuracy, which will be analyzed in next part.

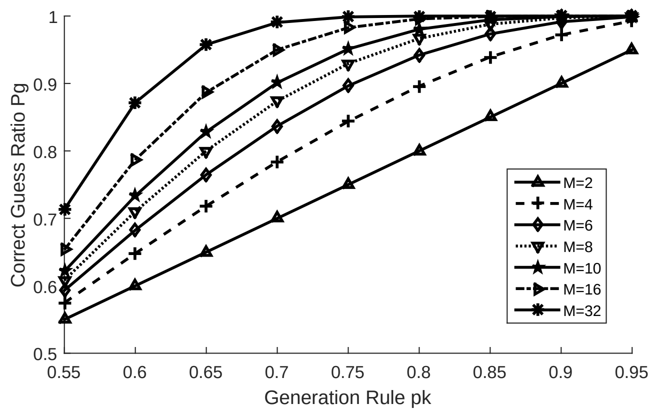


Fig. 4: Correct guess ratio for p_k

E. Trade-off analysis

A trade-off is needed between T-P probability P_{T-P}, T-N probability P_{T-N}, correctness and security(opposite of the guessing probability $P_g(N)$). Firstly we conclude the general correlation between these output variables and all arguments in Table II, where 'P' represents 'positive correlation', 'N' represents 'negative correlation', 'U' represents 'uncertain correlation' and '-' represents 'no correlation'. Details have been analyzed in above parts.

TABLE II: Correlation between T-P, Correctness, Security and arguments

Argument	T-P	T-N	Correctness	Security
p_k	P	-	P	N
N	U	P	P	-
K_{th}	N	P	P	-
M	-	-	-	N
P_T	-	-	P	-

We can divide the trade-off into 2 steps, because p_k has opposite effect on T-P probability, correctness and security while N and K_{th} have opposite effect on T-P probability and correctness.

- Choose p_k according to M, N and system security demand to make a trade-off between security and T-P probability, correctness. To do this step we can refer to the security analysis and Fig.4.
- Roughly select P_T referring to Fig.3. Make another trade-off between T-P probability and correctness by adjusting N and K_{th}.
- Check F-N probability and all other output variables. Repeat all steps if not meeting demand.

To make it clear to see the correlation of T-P probability and correctness and make a trade-off, a 3-D plot is given at Fig.5. The x-axis represents N ranging from 16 to 208 and the y-axis represents K_{th} ranging from $\frac{9}{16}$ to 1. The z-axis represents the probability of T-P or correctness. The trend of T-P probability and correctness with N and K_{th} increasing conforms our above analysis.

Consider the intersection line of these 2 surfaces which means the T-P probability equals to correctness. On the direction of N increasing, the probability converges to 1 and K_{th} gets close to a certain number that is larger than 0.5. This means T-P probability and correctness can both converge to 1 at the same time by adjusting N and K_{th}.

We propose a possible method to make a trade-off here. Recall that when $K_{th} < p_1$ P_{T-P} positively correlates with N and converges to 1 and the correctness always positively correlates with all arguments. So when $K_{th} < p_1$, P_{T-P} and correctness can both converge to 1 at the same time by increasing N. Therefore, to make a trade-off, firstly choose a $K_{th} < p_1$ and try increasing N to satisfy the demand and repeat the method if not satisfied.

Fig.6 gives an example that all of T-P, T-N probability and correctness converge to 1 when $p_1 = 0.75$, $K_{th} = 0.6875$. When $N = 48$, we have $P_{T-P} = 0.846405$, $P_{T_N} = 0.9934$, $correctness = 0.5715$. When $N = 208$, we have $P_{T-P} = 0.9830$, $P_{T_N} = 0.9999$, $correctness = 0.9999$. Assuming $M = 10$, referring to security analysis we can get the guessing probability is: $P_g(208) = 0.3487$ which is relatively small. In scene that in need of high security but don't need a high degree of accuracy, we can improve the security level through reduce p_1. When $p_1 = 0.745$, $N = 48$, we have $P_{T-P} = 0.5450$, $P_{T_N} = 0.9996$, $correctness = 0.6034$ and $P_g(48) = 0.0004$ when $M = 1$, $P_g(48) = 0.76$ when $M = 10$. When $N = 208$, $p_k = 0.75$, we have $P_{T-P} = 0.6417$, $P_{T_N} = 0.9995$, $correctness = 0.5458$ and $P_g(208) = 1e-26$ when $M = 1$, $P_g(208) = 2.94e-5$ when $M = 10$.

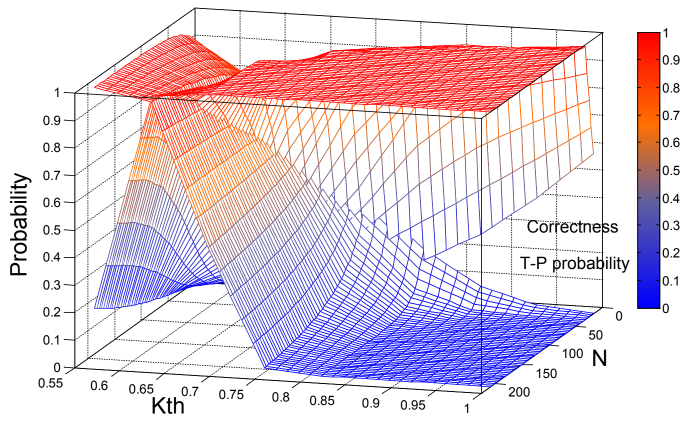


Fig. 5: T-P, Correctness for N, K_{th}

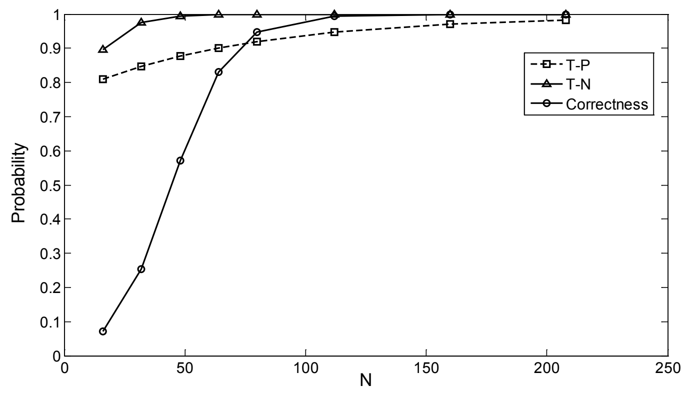


Fig. 6: T-P, T-N, Correctness for N

VII. Conclusion

In this paper we tackles the problem of conflicting phenomenon that arise from variety of mobile cloud storage nowadays. The problem stems from the conflict about exciting functions cloud providing and the potential security issues in cloud. Honest-but-curious server, cloud account loss or cloud attack all may lead to exposure of users' private data, which will be an irreversible disaster. Thus, we develop SPFM to achieve high accuracy matching while not expose accurate private data to cloud. We provide thorough feasibility and security proof and demonstrate the feasibility and security by analyzing experiment performance.

References

[1] A. Acquisti, L. Brandimarte, and G. Loewenstein, "Privacy and human behavior in the age of information," *Science*, vol. 347, no. 6221, pp. 509–514, 2015.

[2] D. Lewis, "icloud data breach: Hacking and celebrity photos," http://www.forbes.com/sites/davelewis/2014/09/02/icloud-data-breach-hacking-and-nude-celebrity-photos/.

[3] R. Shokri, G. Theodorakopoulos, C. Troncoso, J.-P. Hubaux, and J.-Y. Le Boudec, "Protecting location privacy: optimal strategy against localization attacks," in *Proceedings of the ACM conference on Computer and communications security*. ACM, 2012, pp. 617–627.

[4] M. Li, N. Cao, S. Yu, and W. Lou, "Findu: Privacy-preserving personal profile matching in mobile social networks," in *IEEE INFOCOM*. IEEE, 2011, pp. 2435–2443.

[5] W. Dong, V. Dave, L. Qiu, and Y. Zhang, "Secure friend discovery in mobile social networks," in *IEEE INFOCOM*. IEEE, 2011, pp. 1647–1655.

[6] J. He, M. Dong, K. Ota, M. Fan, and G. Wang, "Netseccc: A scalable and fault-tolerant architecture for cloud computing security," *Peer-to-Peer Networking and Applications*, vol. 9, no. 1, pp. 67–81, 2016.

[7] M. Dong, H. Li, K. Ota, L. T. Yang, and H. Zhu, "Multicloud-based evacuation services for emergency management," *Cloud Computing, IEEE*, vol. 1, no. 4, pp. 50–59, 2014.

[8] R. Lu, X. Lin, X. Liang, and X. Shen, "A secure handshake scheme with symptoms-matching for mhealthcare social network," *Mobile Networks and Applications*, vol. 16, no. 6, pp. 683–694, 2011.

[9] M. Von Arb, M. Bader, M. Kuhn, and R. Wattenhofer, "Veneta: Serverless friend-of-friend detection in mobile social networking," in *IEEE International Conference on Wireless and Mobile Computing*. IEEE, 2008, pp. 184–189.

[10] L. Kissner and D. Song, "Privacy-preserving set operations," in *Advances in Cryptology–CRYPTO*. Springer, 2005, pp. 241–257.

[11] Q. Ye, H. Wang, and J. Pieprzyk, "Distributed private matching and set operations," in *Information Security Practice and Experience*. Springer, 2008, pp. 347–360.

[12] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *Advances in Cryptology-EUROCRYPT*. Springer, 2004, pp. 1–19.

[13] E. De Cristofaro and G. Tsudik, "Practical private set intersection protocols with linear complexity," in *Financial Cryptography and Data Security*. Springer, 2010, pp. 143–159.

[14] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *ACM Sigmod Record*, vol. 29, no. 2. ACM, 2000, pp. 439–450.

[15] M. Bellare, A. Boldyreva, and A. ONeill, "Deterministic and efficiently searchable encryption," in *Advances in Cryptology-CRYPTO*. Springer, 2007, pp. 535–552.

[16] A. Boldyreva, S. Fehr, and A. ONeill, "On notions of security for deterministic encryption, and efficient constructions without random oracles," in *Advances in Cryptology–CRYPTO*. Springer, 2008, pp. 335–359.

[17] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proceedings of the ACM SIGMOD international conference on Management of data*. ACM, 2004, pp. 563–574.

[18] A. Boldyreva, N. Chenette, Y. Lee, and A. Oneill, "Order-preserving symmetric encryption," in *Advances in Cryptology-EUROCRYPT*. Springer, 2009, pp. 224–241.

[19] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh, "Location privacy via private proximity testing." in *NDSS*. Citeseer, 2011.

[20] A. Thapa, M. Li, S. Salinas, and P. Li, "Asymmetric social proximity based private matching protocols for online social networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 26, no. 6, pp. 1547–1559, 2015.

[21] X. Liang, M. Barua, R. Lu, X. Lin, and X. S. Shen, "Healthshare: Achieving secure and privacy-preserving health information sharing through health social networks," *Computer Communications*, vol. 35, no. 15, pp. 1910–1920, 2012.

[22] R. Zhang, Y. Zhang, J. Sun, and G. Yan, "Fine-grained private matching for proximity-based mobile social networking," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 1969–1977.

[23] R. Zhang, J. Zhang, Y. Zhang, J. Sun, and G. Yan, "Privacy-preserving profile matching for proximity-based mobile social networking," *Selected Areas in Communications, IEEE Journal on*, vol. 31, no. 9, pp. 656–668, 2013.

[24] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 362–375, 2013.

[25] J. Sun, R. Zhang, and Y. Zhang, "Privacy-preserving spatiotemporal matching," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 800–808.

[26] T. Jung, X. Mao, X.-Y. Li, S.-J. Tang, W. Gong, and L. Zhang, "Privacy-preserving data aggregation without secure channel: Multivariate polynomial evaluation," in *IEEE INFOCOM*. IEEE, 2013, pp. 2634–2642.

[27] J. Sun, R. Zhang, and Y. Zhang, "Privacy-preserving spatiotemporal matching for secure device-to-device communications," 2016.