LINKTIMITE: User Privacy-Preserved Social Network using SPFM Protocol

Aswin Pradeep Das.T.H
Student, Department of Computer Science and Engineering, University College of Engineering, Nagercoil.
Arun Kumar.V.K
Student, Department of Computer Science and Engineering, University College of Engineering, Nagercoil.
Boothalingam.M
Student, Department of Computer Science and Engineering, University College of Engineering, Nagercoil.
Karthik.G
Student, Department of Computer Science and Engineering, University College of Engineering, Nagercoil.
Mrs.K.L.Neela
Assistant Professor, Department of Computer Science and Engineering, University College of Engineering, Nagercoil.

Abstract – The Internet of Things represents the intersection of individuals (meatspace), systems (cyberspace) and the physical world (atomspace). The growth of Internet services and popularity of social networking sites has actively increased over the past few years. The privacy concerns with the social networking services are threatening and it comes under the data privacy, involving the unwanted advertisements and spams or scams, cause social reputation or economic damage, and make them victims of blackmail or even physical violence. User's profile (e.g., contact list, interest, location) matching is more important for promoting the wide use of mobile social networks. The social networks such as Facebook, Instagram, Line, WeChat recommend the friends for the users based on user personal data such as common contact list, interest attributes, mobility traces. However, outsourcing user's sensitive profile to the cloud for friend matching will raise a potential privacy concern due to the serious risk of data abusing. More privacy-preserving computation schemes based on encryption, or cooperative computing requires multiple exchanges between participants, which is not suitable for the cases where users are not connected to each other or number of users is relatively large. This project proposes the concept of Scalable and Privacy-preserving Friend Matching protocol, or SPFM in short, which can overcome the limitations of previously proposed schemes and provide a scalable friend matching and recommendation solutions without revealing the user's private data to the cloud.

Index Terms – Friend matching, Privacy preserving, XOR, Clustering algorithm.

1. INTRODUCTION

The social media is a broad phenomenon, as the count of worldwide users of social network is predicted to grow from 2.34 billion in 2016 to around 2.95 billion in 2020 [17], which is around one third of Globe's entire population. The average daily time spent on social networking has also expanded in the past few years, as has the influence interactions on social networks might have on anything from one's politics to musical tastes, leisure or purchase reviews, feelings and emotional sharing. Users employ OSNs as a tool to connect with family, friends, colleagues, associates, or people with the same interests for different purposes such as professional networking, advertising their brands and businesses, job searches, making profit, or entertainment. Some networks, such as Facebook, started out as web-based and then extended towards access through mobile browsers and smartphone apps, while other networks, such as Instagram, were initially mobile-only and later extended into cross-platform availability as well with the help of web apps. An increasing number of social networks are therefore accessible through multiple platforms in order to offer users access to different features according to their needs, time and preferred device.

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 attention 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, outsourcing users' personal information to the cloud for friend matching will raise a serious privacy risks. The potential risk for personal data to be abused for economic and social discrimination, hidden influence and manipulation, is alarming. Existing researches show that loss of privacy can expose users to unwanted advertisement and spams and scams, cause social reputation or economic damage, and make them victims of blackmail or even physical violence [18]. It can lead to devastating financial and social consequences as well as causing extreme psychological trauma to the person involved.
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. 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, disease symptoms, or friends etc.). The private profile matching problem could then be converted into Private Set Intersection (PSI) [1,2]. 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 friend 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.
Hence the 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 user's personal data to the cloud. The 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, however the friend matching used in SPFM protocol [18] uses only Friends-of-Friends algorithm, which isn't efficient enough to scrutinize matching on its own when the number of users are relatively large, hence attribute based clustering can be used along with the SPFM protocol, hence it can deliver a stable state of balance between the privacy and friend matching. The contributions of this work are summarized as follows:

- Scalable and Privacy-preserving Friend Matching scheme (SPFM) is used to prevent privacy leakage in friend matching and recommendation system.
- Friends-of-Friends (FoF Algorithm) an extension of Breadth First Search Algorithm (BFS) is used for computation of mutual friends for matching by avoiding self-connected loops.
- k-means clustering partitions the total observations of mutual friends into k clusters in which each observation belongs to the cluster with the nearest mean, using the Euclidean distance as another Closeness Factor, finally serving as an attribute based recommendation system.
- In the overall process of friend matching and recommendation the sensitive private is nowhere leaked.

The contents of this paper are as follows. In section 2, 3, 4 & 5, we describe related works, problem formulation, system design respectively. In section 6, we conclude the paper.

2. RELATED WORK

Agrawal et al. [3] first proposes Order preserving encryption for numeric data, which allows comparison on encrypted data without decrypting the values. However, this paper mainly focuses on statistical information of unordered databases and doesn't rise to the level of data matching.

M. Von Arb et al. [4] proposes Veneta: Server-less friend-of-friend detection in mobile social networking, which allows decentralized Friends-of-Friends (FoF) finding. However, it is inaccurate and works only in short range wireless communication.

Q. Ye et al. [5] proposes Distributed private matching and set operations, which uses homomorphic cryptosystem and uses set intersection for matching but this shows high computational complexity.

Xinjun Qi et al. [6] proposes 'An Overview of Privacy Preserving Data Mining(PPDM)' which uses random data distortion technique and provides minimal information loss along with techniques for data reconstruction. However, data reconstruction can be done in a polynomial time.

Georgia Athanasopoulou et al. [7] proposes eMatch: An Android Application for Finding Friends in Your Location which aims at of connecting people with common interests in nearby location using clustering. However, it fails to maintain a balance between privacy and friend matching, it strictly focuses on friend matching alone.

More privacy-preserving computation schemes [8-15] based on encryption, or cooperative computing requires multiple exchanges between participants, which is not suitable for the cases where users are not connected to each other or number of users is relatively large.

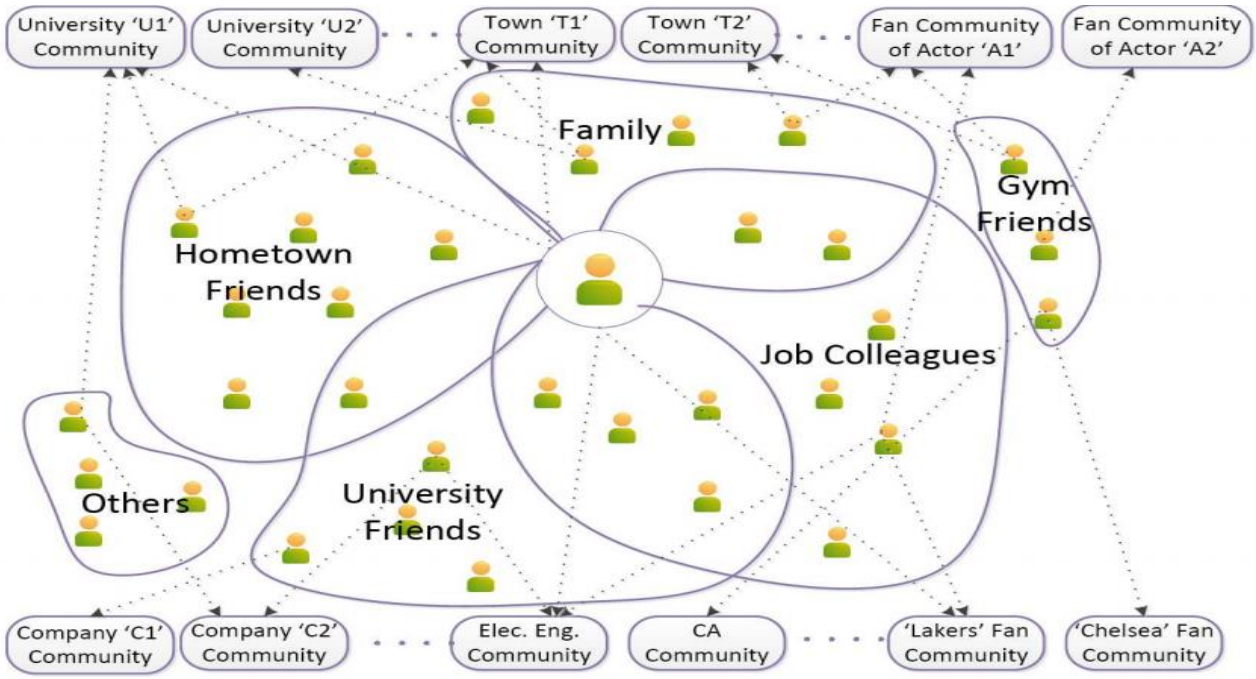

Fig. 1: Friends distribution representation of a user in different communities with different attributes of interests.

3. PROBLEM FORMULATION

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

3.1 Application Scenario

Online social network has obtained a significant increase in recent years. Making friends is a way of creating social relationships with others in online social network to be in contact with their friends in the real world and to have access to the information they are interested in. Therefore, friend recommendation is becoming a very important aspect and attracting extensive attention in visual communities and social network. Various available friend recommendation techniques apply social network configuration, user profile information and/or user interactions for this purpose. 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 are 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, and in the contact list for the identity of a user's profile the mobile number is used, which is considered to be a sensitive private profile information, because the mobile number acts as a factor of identity of that user and using which the user can be identified outside the social network.

But, the sensitive information uploaded to the cloud may face the risk of leaking user's sensitive data and compromising users' privacy. In this work, a privacy preserving friend matching scenario, in which the user's 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.

However, the previous work [18] only deals with Friends-of-Friends matching or matching with mutual friends which is not accurate enough to produce recommendation results since if considering a user may have common friends scattered in large areas or communities as shown in the figure. 1, [8] the users within a community is assumed to have a similar interest attributes. Hence this paper proposes and introduces the attribute based recommendation in-order to further scrutinize the recommendation result and provide more accurate results.

3.2 System definition

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 (Pm) 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 effective when P_m is in an appropriate range, usually larger than 0.00001 [18]. 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 practical scenario then Data tag contains the contact list and limited attributes of a user.

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 $U = \{u_1, ..., u_K\}$. 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.

Definition 3. *(k-Means clustering) k-Means clustering intends to partition n objects into k clusters in which each object belongs to the cluster with the nearest mean. This technique produces specifically k completely different clusters of greatest doable distinction. The best number of clusters k leading to the greatest separation (distance) must be computed from the data.*

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2$$

$\left\| \mathbf{x_i^{(j)}} - \mathbf{c_j} \right\|^2$ denotes the Euclidean distance function. Here the k-clusters denotes the k number of attributes or interests and n denotes the number of mutual friends for the user (u_i) obtained as the result of FoF algorithm. In practical scenario, x_i represents the location of person and c_j , the centroids location.

Definition 4. *(Euclidean distance) The Euclidean distance between points p and q is the length of the line segment connecting them.*

In Cartesian coordinates, if $p = (p_1, p_2,..., p_n)$ and $q = (q_1, q_2,..., q_n)$ are two points in Euclidean n-space, then the distance (d) from q to p, or from p to q is given by the Pythagorean formula:

$$\| p - q \| = \mathrm{d}(\mathbf{p}, \mathbf{q}) = \mathrm{d}(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

3.3 Assumptions and Adversary models

The cloud server determines two users are real friends by matching how many friends they have in common and also matches the attributes and their locations. 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 of the phone number in the 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, and further they are in nearby locations to each other with similar attributes of interest. For example, if a group of students in the same location with same matching attributes then there is a maximum possibility of they are in the same work place or an educational institution.

There are two main adversary models in this paper. One is external attacker, who will try to get your privacy through

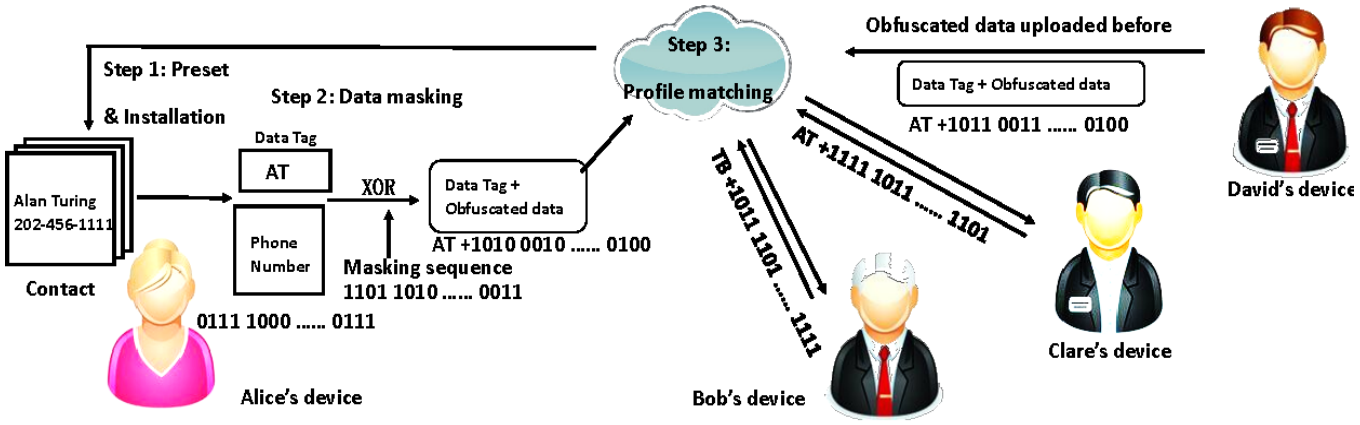

Fig. 2: System Design

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) [18], 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.

4. SYSTEM DESIGN

In this section, we describe the system design in detail. The overall functionality of the system can be divided into four steps. Fig. 2 [18] provides an overview of the work-flow of our approach. Algorithm 1 explains how the system works and we will describe each step in detail.

4.1 Setting up the parameters

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.

4.2 Data Obfuscation

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.

Algorithm 1 SPFM (U, Cloud, pk)

1: U, Cloud← p_k.
2: **for** each $u_i \in U$ **do**
3: $D_i \leftarrow$ ListOfContact(u_i).
4: $T_i \leftarrow$ AbbreviationOfName(D_i).
5: $B_i \leftarrow$ PhoneNumber(D_i).
6: **for** each $B_{ix} \in B_i$ **do** ▷Data obfuscation starts
7: $O_x \leftarrow$ MaskGeneration(p_k).
8: $C_{ix} = B_{ix} \oplus O_x$.
9: **end for** ▷Data obfuscation ends
10: u_i upload C_i with 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 ($i \neq j$) $\in U$ **do** ▷FoF Algorithm starts
15: **for** each $t_x \in T_i$ **do**
16: **for** each $t_y \in T_j$ **do**
17: **if** $t_x == t_y$ **then**
18: MatchResult ← MatchingPhase(C_{ix},C_{jy},u_i u_j).
19: **else**
20: MatchResult = No ← (C_{ix}, C_{jy}).
21: **end if**
22: **end for**
23: **end for**
24: **end for** ▷FoF Algorithm ends

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.

4.3 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 5. *(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.*

$$d_i = c_i \oplus c_i', i = 1,2, \dots N$$

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 6. *(Matching ambiguity). The ratio of Threshold and the original data's length is defined as Matching Ambiguity (K_{th}). For an N-bit binary sequence, the ratio of Threshold and N is defined as the Matching Ambiguity.*

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

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 server first does a traversal through two user's Data Tags and find 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, the Friends-of-Friends algorithm will be used for getting the common friends or mutual friends of a user, for that purpose server will consider the telephone number stored in two users under this Data Tag are the same. In other words, the Data Tag will contain the contact list and attributes of the specified user. After a thorough traversal of all the Data Tags, server will get the common friends between these two users.

Table 1. System notations

N	Length of data sequence
p_k.	Masking generation probability
B	N-bit original binary sequence
O	N-bit masking sequence
C, C'	Obfuscated sequence
D	XOR (comparison) result of C, C'
n_{th}	Threshold to decide same original sequence
K_{th}	Ratio of n_{th} and N
u_i	User who request for friend matching
u_j	A direct friend of user u_i
k	Total number of clusters
J	Closeness factor (Euclidean distance) between two users
$\vec{\mu}_0$	Centroid of cluster in which user u_i belongs
$\vec{x}_j$	Directional vector of user who is a friend of user u_j
n	Number of friends of the user u_j

Getting the common friends with Friends-of-Friends algorithm alone insufficient for deciding whether two users are real friends in a practical application scenario. Hence the proximity factor or closeness factor is introduced. Here the attributes of a user including interests and location is used as a closeness factor for judging the proximity between two users. The attributes used here won't affect the privacy criteria since mobile number is obfuscated which acts as an identity for identifying the user outside the social network, since one

minute of latitude or longitude accounts for about 1.15 mi and can't be used as an identity since no precise location is used. Hence using k-means clustering will be efficient and produce a drastically accurate recommendation result. The Algorithm 2 explains in detail how the Friend matching phase works. The k-means clustering algorithm starts if the user u_j is a mutual friend. The following steps are performed in the k-means clustering algorithm.

Algorithm 2 MatchingPhase *(C_{ix}, C_{jy}, u_i u_j)*

1: **for** $i = 1,2, \dots, N$ **do**
2: $d_i = c_{ix} \oplus c_{iy}$.
3: **end for**
4: **if** $(NoOfZeros(d_i) \leq n_{th})$ ▷if mutual friend.
5: $\vec{x}_j \leftarrow ListofContact(u_j)$. ▷k-means starts
6: $n \leftarrow Length(\vec{x}_j)$
7: $\vec{\mu}_0 \leftarrow Atttributes(u_i)$
8: **set** $\vec{\mu}_1, \dots, \vec{\mu}_k$ *be distinct randomly selected input attributes from* $\vec{x}_i, \dots, \vec{x}_n$.
9: **repeat**
10: **for** $i = 0,1, \dots, n$ **do**
11: **if** $\begin{pmatrix} J = argmin_j \parallel \vec{x}_j - \vec{\mu}_j \parallel^2 and \\ |\vec{\mu}_i| == |\vec{\mu}_j| \end{pmatrix}$
12: $\gamma_{ij} \leftarrow 1.$ ▷assign cluster membership
13: **else**
14: $\gamma_{ij} \leftarrow 0.$
15: **end for**
16: **for** $j = 1,2, \dots, k$ **do**
17: $n_j \leftarrow \sum_{i=1}^{n} \gamma_{ij}$.
18: $\vec{\mu}_j \leftarrow \frac{1}{n_j} \sum_{i=1}^{n} \gamma_{ij} . \vec{x}_j$ ▷Recalculate centroid
19: **end for**
20: **until convergence**
21: **Return** $\vec{\mu}_0$
▷k-means ends, cluster having user u_i alone returned
22: **else**
23: **Return** 0.

Each centroid defines one of the clusters. The cluster $\vec{\mu}_0$ denotes the cluster to which the user u_i belongs to. Each user is assigned to its nearest centroid, based on the squared Euclidean distance. More formally, if $\vec{\mu}_j$ is the collection of centroids , then each data point $\vec{x}_j$ is assigned to a cluster based on minimum Euclidean distance between the users and they should also have the same interest attributes. In line 17, the centroid is updated. This is done by taking the mean of all data points (users) assigned to that centroid's cluster. The iteration is continued until converge to a result. Finally, the cluster containing user u_i is returned. The final result will be filtered result of users for recommendation.

5. EVALUATION

In this section, we carry out an experimental study of the complexity, performance and comparison results when using SPFM and Friends of Friends algorithm along with k-means clustering algorithm. The evaluation results show the good performance of the proposed system in different aspects.

5.1 . Complexity Analysis

In the previous works [3,6,11,16,17] the social networking and cloud privacy areas have come across various techniques and models still it is a greater challenge in providing a less complex technique without compromising the privacy criteria as well the runtime complexity of algorithms. There are different algorithms proposed previously notable one are the homomorphic cryptographic algorithms [16, 17], since the homomorphic crypto algorithms allows the process of carrying out computations on encrypted data. This technique allows for privacy preserving computation. [6] uses Pallier's cryptosystem which is a classical algorithm and considered to be more complex one. [16, 17] uses Partially homomorphic encryption with RSA algorithm. The malleability in the RSA algorithm is a phenomenon that means that the multiplication of cipher text is equal to the multiplication of the original messages and hence it is convincing to use in our problem statement, since it helps in computation over the encrypted data for friend matching. It is claimed [21] that RSA is the less time complex homomorphic algorithm in practical usage.

However, in our system the SPFM uses the data obfuscation or data masking technique. We say that this data obfuscation using the ordinary XOR bit operation is less complex and doesn't compromise the privacy. Since the masking sequence is unnecessary for the computation of friend matching by the public cloud, we can say this technique provide an extended privacy.

The Table 2. shows the comparative experimental result of execution time in milliseconds of both RSA homomorphic encryption and the data masking in SPFM protocol for different lengths of data in number of bits.

Fig.3 shows the comparison of RSA homomorphic encryption algorithm and data obfuscation algorithm in SPFM for different set of length of data sequence in bits. The result show that the data masking in SPFM has comparatively less time complexity.

Table 2. Execution time of RSA homomorphic encryption vs Data masking in SPFM protocol.

Length of Bits	Runtime of RSA (ms)	Runtime of Data Obfuscation in SPFM (ms)
2083	49.9453 milliseconds	28.546 milliseconds
2247	56.3892 milliseconds	34.8794 milliseconds
2524	62.7788 milliseconds	36.247 milliseconds
2762	67.6062 milliseconds	39.9431 milliseconds
3209	83.7358 milliseconds	51.5983 milliseconds
3739	108.6065 milliseconds	66.2567 milliseconds
4680	142.941 milliseconds	89.7213 milliseconds
6095	224.8633 milliseconds	141.855 milliseconds
9242	641.4635 milliseconds	332.5659 milliseconds
18580	3818.8579 milliseconds	781.6587 milliseconds

The results of the analysis show that data obfuscation in SPFM perform well than the RSA homomorphic encryption. Further the complexity analysis show that encryption in RSA Homomorphic algorithm has a time complexity of *Big-O(2^N)*. But, in order to perform one time of Data Obfuscation of N bits, 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 is *Big-O(N).*

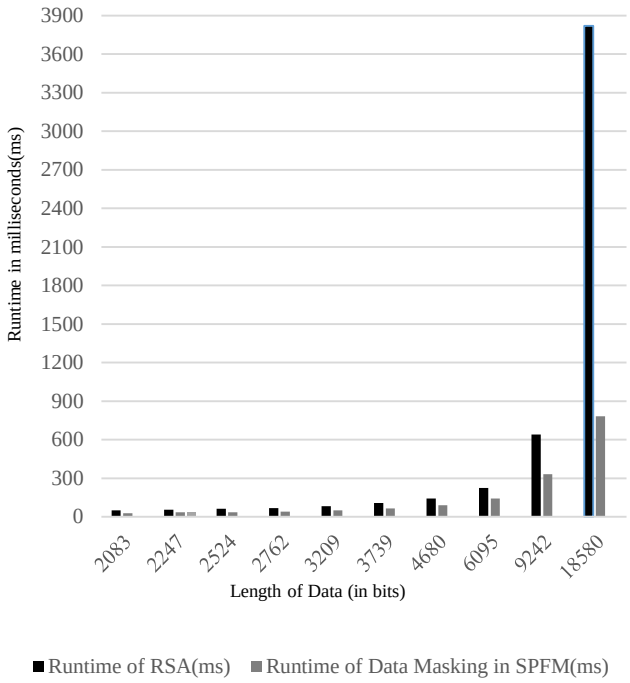

Fig. 3: Runtime comparison of RSA homomorphic encryption Algorithm Vs Data Obfuscation algorithm in SPFM.

5.2 . Evaluation of Friend Matching

For the evaluation of the proposed friend matching module of Friends of Friends along with k-means clustering Algorithm we collected the contact list of a user with 103 contacts. The dataset consisted of 6 interest categories, the latitudinal and longitudinal location values of 103 users. All the 103 users come under the 6 categories they are automobile, computer, cuisine, medicine, electronics, teaching. The clustering is done in the recommenderlab package in R Studio for analysis, since our system does the same function. The latitude and longitude values are fed to the Euclidean formula to get the closeness factor value, that is the Euclidean distance between the two user. The dataset is inserted into the R Studio and clustering is performed using the following command

attributeCluster <- kmeans(kmean[, 2:3], 6, nstart = 20)

where, nstart is the random value to initialize the cluster centroid but in the practical implementation the random function is used. Then the attribute list is inserted and the clusters are plotted using the ggplot function. The Fig.4 shows the distribution of clusters. From the graph it is evident that similar interest or like-minded people have greater possibility of location closeness and social proximity.

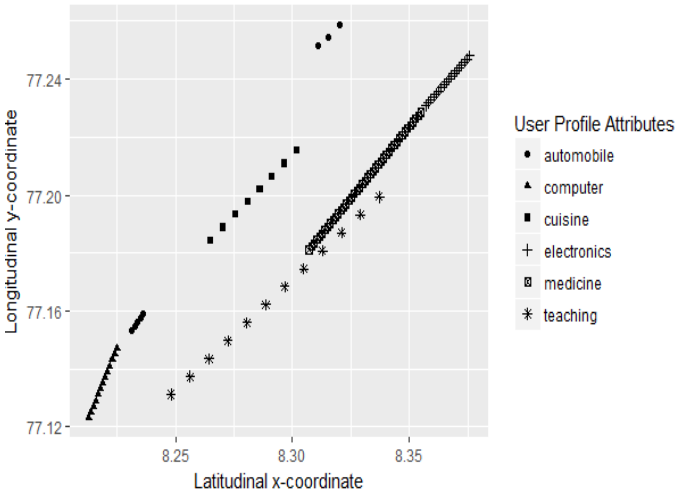

Fig. 4: Distribution of friends of user u_j into different clusters.

For the evaluation of we compare the previous work [18] with Friends of Friends algorithm alone and our work with Friends of Friends algorithm with k-means clustering with different evaluation metrics such as Positive Predictive Value (PPV) or Precision, False positive rate (FPR) and F1-score. Here we give some definitions for the terms and concepts in our system context that we make use of to evaluate the friend matching module.

Definition 7. *(True Positives) True Positives (T_P) refer to the number of recommended users who were actual friends to the friend recommendation requested user in reality.*

Definition 8. *(False Positives) False Positives (F_P) refer to the number of recommended users who were strangers to the friend recommendation requested user in reality.*

Definition 9. *(True Negatives) True Negatives (T_N) refer to the number of non-recommended users who were strangers to the friend recommendation requested user in reality.*

In other words, it is the number of missed out false recommendations.

Definition 10. *(False Negatives) False Negatives (F_N) refer to the number of non-recommended users who were actual friends to the friend recommendation requested user in reality.*

In other words, it is the number of missed out true recommendations.

Definition 11. *(Sensitivity or Recall or True Positive Rate) Sensitivity (SN) is calculated as the number of correct positive recommendations divided by the total number of positives.*

In other words, it is the proportion of users who are actual friends in reality were recommended by our system as friends. It is also called recall or true positive rate. The best case sensitivity is 1.0, whereas the worst case is 0.0.

$$SN = \frac{T_P}{T_P + F_N}$$

Definition 12. *(False Positive Rate) False Positive Rate (FPR) is calculated as the number of incorrect positive recommendations divided by the total number of negatives.*

In other words, it is the rate of false hit. The best case false positive rate is 0.0 whereas the worst case is 1.0. It is also known as 'false alarm rate'.

$$FPR = \frac{F_P}{T_N + F_P}$$

Definition 13. *(Positive Predictive Value or Precision) Positive Predictive Value (PPV) is the calculated as the number of correct positive recommendations divided by the total number of positive recommendations.*

In other words, it is the proportion of users who were recommended by our system as friends were actual friends in reality. It is also called positive predictive value (PPV). The best case precision is 1.0, whereas the worst case is 0.0.

$$PPV = \frac{T_P}{T_P + F_P}$$

Definition 14. *(F1-score) F1-score is a harmonic mean of precision and recall. It is a measure of precision and recall of a recommender system. The best case F1-score is 1.0, whereas the worst case is 0.0.*

$$F1 - score = \frac{2.PPV.SN}{PPV + SN}$$

Now consider the assumption that one of the user u_i who is also a subset of the 103 users in the dataset request for recommendation. The user from whom the contact list is extracted in the form of dataset i.e., user u_j who is the friend of ui in reality. Then the total number of users taken as input to the friend matching system are 102. The user u_i is interested in

J	*N1*	T_P	T_N	F_N	F_P	*PPV*	*SN*	FPR	*F1-Score* with FoF-Algorithm alone
0.01	31	27	52	19	4	0.8709	0.586	0.0714	0.701
0.04	55	37	38	9	18	0.6727	0.804	0.3214	0.732
0.08	67	41	30	5	26	0.6119	0.891	0.464	0.743
0.12	75	43	24	3	32	0.5733	0.934	0.5714	0.71
0.16	88	43	11	3	45	0.4886	0.934	0.803	0.641
0.211	102	46	0	0	56	0.4509	1	1	0.605

Table 3. Matching matrix for friend matching with only FoF-Algorithm

medicine and that user get recommended like-minded users in our proposed system whereas with Friends of Friends algorithm alone he will get all 102 users to be recommended. The same concept can be derived from the dataset and can be represented in the form of matching matrix in the Table.3 and Table.4. The matching matrix in Table.3 shows the True positives, True negatives, False positives, False negatives Sensitivity, Positive Predictive Value and F1-Score of the recommendation system while using only FoF- Algorithm for different Euclidean distance between users or Closeness factor (J). Although the FoF algorithm doesn't compute the closeness factor (J) it is introduced here to compare the results of performance metrics at different physical proximity levels between the users. Here N1 is the total number of friends of u_j recommended to the user u_i.

Table 4. Notations used in evaluation

T_P	True Positives
F_P	False Positives
T_N	True Negatives
F_N	False Negatives
SN	Sensitivity or Recall or True Positive Rate
FPR	False Positive Rate
PPV	Positive Predictive Value or Precision
J	Closeness factor
N1	Total number of recommended users

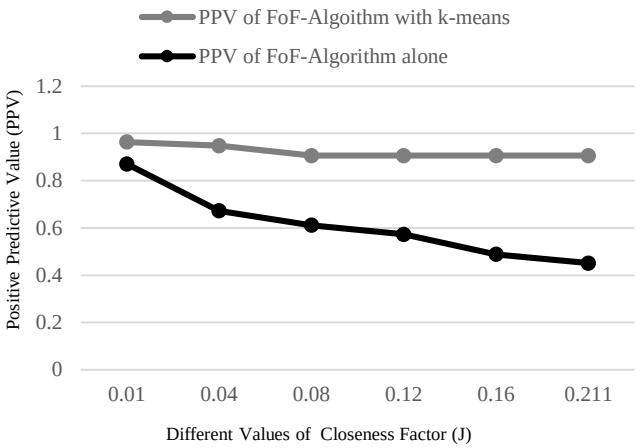

Fig. 5: Positive Predictive Value (PPV) comparison

The Fig. 5 shows the comparison of the Positive Predictive Value (PPV) between the Friend matching done with FoF-Algorithm alone and FoF-Algorithm along with k-means clustering. The result shows that using k-means increases the PPV and there is a drop in PPV in the other case with an increase in the Closeness factor value (J). The Fig. 6 shows the comparison of the False Positive Rate (FPR) between the Friend matching done with FoF-Algorithm alone and FoF-Algorithm along with k-means clustering. The result shows that using k-means lowers the FPR to an extend and there is a High FPR in the other case with an increase in the Closeness factor value (J) or the Euclidean Distance. The Fig. 7 shows the comparison of the F1-Score between the Friend matching done with FoF-Algorithm alone and FoF-Algorithm along with k-means clustering. The result shows using k-means increases the F1-score than the other case for different values of the Closeness Factor (J).

J	*N1*	T_P	T_N	F_N	F_P	*PPV*	*SN*	FPR	*F1-Score* with k-means Algorithm
0.01	28	27	55	19	1	0.964	0.586	0.0178	0.7297
0.04	39	36	54	9	2	0.948	0.804	0.0357	0.867
0.08	43	39	52	7	4	0.906	0.847	0.0714	0.8764
0.12	43	39	52	7	4	0.906	0.847	0.0714	0.8764
0.16	43	39	52	7	4	0.906	0.847	0.0714	0.8764
0.211	43	39	52	7	4	0.906	0.847	0.0714	0.8764

Table 5. Matching matrix for friend matching with only FoF-Algorithm along with k-means Algorithm

The Table.5 shows the values of metrics that we obtained when the k-means clustering is used along with the FoF-Algorithm. The value of SN or Sensitivity increases with an increase in closeness factor that is because all the users belonging to the contact list of user u_j were recommended to user u_i and there are no false negative users at all.

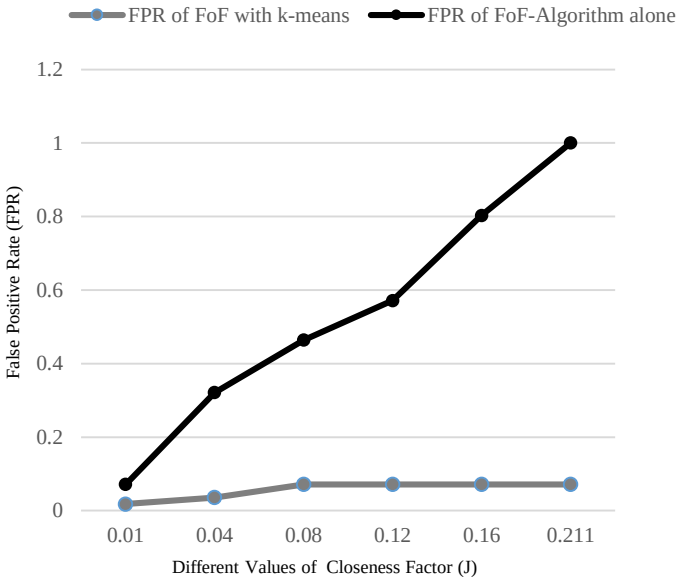

Fig. 6: False Positive Rate (FPR) comparison.

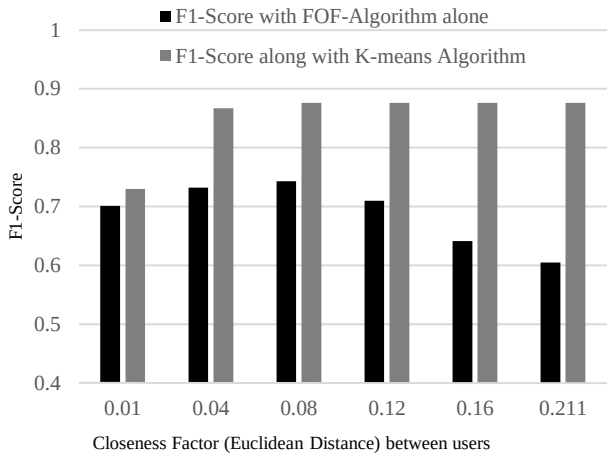

Fig. 7: F1-Score comparison

Thus, the conclusion is that FoF-along with k-means clustering performs well in all aspects than using the FoF-Algorithm alone.

6. CONCLUSION

This paper gives an apt solution to tackles the privacy concerns with the social networking services and also provides an accurate friend matching service while not exposing the sensitive user information to the public cloud. This can protect the user's sensitive private information from leakage and prevents various spams, scams and abuses. The attribute based clustering along with the SPFM protocol delivers a stable state of balance between the privacy and friend matching. We also provided evaluation using various metrics to demonstrate the accuracy and efficiency of our system.

REFERENCES

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

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

[3] 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.

[4] 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.

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

[6] Xinjun Qi and Mingkui Zong, "An Overview of Privacy Preserving Data Mining," in 2011 International Conference on Environmental Science and Engineering (ICESE 2011), pp.2012.01.432.

[7] Georgia Athanasopoulou and Polychronis Koutsakis, "eMatch: An Android Application for Finding Friends in Your Location" in Hindawi Publishing Corporation Mobile Information Systems, pp.2015, Article ID 463791, 11 pages.

[8] Arun 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

[14] 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.

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

[16] Kurra Mallaiah, and S. Ramachandram, "Applicability of Homomorphic Encryption and CryptDB in Social and Business Applications: Securing Data Stored on the Third Party Servers while Processing through Applications," in International Journal of Computer Applications (0975 – 8887), on Volume 100– No.1, August 2014.

[17] Maha TEBAA, and Said EL HAJII, "Secure Cloud Computing through Homomorphic Encryption," in International Journal of Advancements in Computing Technology(IJACT), on Volume5, Number16, December 2013.

[18] Mengyuan Li, Ruan Na, QiYang Qian, Haojin Zhu, Xiaohui Liang, and Le Yu, "SPFM: Scalable and Privacy-preserving Friend Matching in Mobile Cloud", in IEEE Internet of Things Journal, pp.2016.2582780.

[19] Statista, "Social media - Statistics & Facts", https://www.statista.com/-topics/1164/social-networks/.

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

[21] A.E.Okeyinka, "Computational Speeds Analysis of RSA and ElGamal Algorithms on Text Data" in Proceedings of the World Congress on Engineering and Computer Science, pp.2015 Volume I