Hindawi
Scientific Programming
Volume 2020, Article ID 7832875, 13 pages
https://doi.org/10.1155/2020/7832875

Research Article

A User-Defined Location-Sharing Scheme with Efficiency and Privacy in Mobile Social Networks

Tao Peng,[1] Jierong Liu,[1] Guojun Wang,[1] Qin Liu,[2] Jianer Chen,[1] and Jiawei Zhu[1]

[1]*School of Computer Science, Guangzhou University, Guangzhou, Guangdong 510006, China*
[2]*School of Information Science and Engineering, Hunan University, Changsha, Hunan 410082, China*

Correspondence should be addressed to Guojun Wang; csgjwang@163.com

Received 13 February 2020; Accepted 31 July 2020; Published 26 October 2020

Academic Editor: Chenxi Huang

The popularity of the modern smart devices and mobile social networks (MSNs) brings mobile users better experiences and services by taking advantage of location-aware capabilities. Location sharing, as an important function of MSNs, has attracted attention with growing popularity. While the users get great benefits and conveniences from MSNs, they also have high concerns about the privacy of location. However, in the existing solution, the privacy of users can hardly be guaranteed without the assumption of full trust in the service provider (SP), and few previous research studies have discussed the individual requirement of mobile users in MSNs. In this paper, we propose a user-defined location-sharing scheme (ULSS) to achieve enhanced privacy preservation under different contexts. We present a coarse-grained proximity detection method and a lightweight order-preserving encryption- (OPE-) based method to provide the users with flexible privacy preservation at different privacy levels. The proposed scheme preserves user's location privacy with respect to SP, friends, and other adversaries, getting rid of the introduction of fully trusted party (TTP). Extensive experiments were conducted to verify the effectiveness and efficiency of our proposed scheme.

1. Introduction

With the development of the smartphone and wireless communications technology, mobile social networks (MSNs) have seen a tremendous rise and are becoming a kind of important tool for our daily communication. MSNs such as Foursquare, Wechat, and Google Latitude bridge the users' physical and social worlds taking advantage of location-aware capabilities by mobile devices. Sharing and interaction around content and information is the key feature of the MSNs. In particular, *location sharing* [1], as an important function of MSNs, allows users to share their current locations to the friends on the social networks or to find the nearby friends within a certain physical distance. Based on it, MSNs bring mobile users better experiences and services by various location-based and personalized services, e.g., proximity-based detection [2] and friend locator [3], or just simple location sharing [1].

MSNs facilitate the mobile users and make it easier for social friends to connect in real life. While the users reap great benefit and convenience from MSNs, they also have high concerns about the privacy of location, since the users have to submit their exact location information to their friends or to the service provider (SP) when they enjoy the service. The centralized SP collects all user's information, e.g., the user identity, the exact location, and query content. If the private information is disclosed by potentially untrustworthy SP or is compromised by an adversary, it will put the users' sensitive information in jeopardy. Specifically, in MSNs, the user's location information is associated with the social network ID and the social relationships of users, which makes the user privacy preservation more significant and more complicated [4–7]. Therefore, there are both the development chance and challenge in MSNs, and the key is how to protect the privacy of users when providing high-quality services.

Over the past years, some approaches have been proposed to address location privacy in MSNs [8–10], and a few solutions [11–14] were presented to focus on user privacy preservation in case of location sharing. Li et al. [11] presented a secure location-sharing scheme in which the user information was stored in two kinds of servers, e.g., location server (LS) and social network server (SNS), which incurs high communication cost due to the multiple-round interaction between the user and the servers. Moreover, the privacy of users can hardly be guaranteed without the assumption of full trust in the SP. In our previous work [3], we proposed a lightweight privacy-aware friend locator (PAFL) to provide privacy guarantee for the user at low computational and communication cost. However, the coarse-grained friend locator method cannot meet the personalized needs of mobile users. The challenge is that the user may submit diverse requirements and wants to set different levels of privacy protection due to the varying contextual conditions. Here is a typical example of proximity-based location sharing: Alice travels to a city, she first wants to find out the friends who are located in the destination she is visiting by the MSNs, e.g., Facebook Connect. Next, according to the name list returned by server, Alice wants to query some specified friends whether they are in the nearby vicinity region defined by her, e.g., within a mile or so. In this application, Alice refuses to reveal her exact location to any third party (including SP and other users), whether she is the initiator of the query or the location provider (friend). While the existing schemes [11, 14] are inflexible for users to change the privacy setting, it is difficult to meet the individualized demand preferences for different users in the traditional system.

In this paper, to satisfy the users' individual requirements, we propose a user-defined location-sharing scheme (ULSS) to realize proximity query of users in different scales of vicinity regions. Our motivation is to provide the mobile users with flexible privacy controls under different contexts in an efficient and friendly way. We first propose a coarse-grained proximity detection method based on the Hilbert curve to determine the friends in a wide range of vicinity. Then, we provide a lightweight order-preserving encryption- (OPE-) based method to enable users to query the specular friends in the nearby specified vicinity, e.g., within 5 km. In the whole process, users can submit their proximity query according to their individual requirements and different personal privacy settings, without revealing the exact location information to any SP or to any other possibly malicious users.

Our main contributions are summarized as follows:

(1) We propose a flexible ULSS for private proximity detection, which allows each user to maintain his own privacy-preserving policy and provides the users with flexible privacy protection at different privacy levels.

(2) We propose two protocols to process the user queries in different phases, which preserve user's location privacy with respect to SP, friends, and other adversaries. We also prove that the proposed scheme is secure under the stronger security model with enhanced privacy.

(3) We evaluate the performance of our ULSS scheme by extensive experiments. Experiment results demonstrate that our scheme is extremely efficient for coarse-grained proximity detection and provides user-defined nearby friend locator at low computational and communication cost as well.

In our scheme, the fully-trusted third party is not required, the SP is assumed to be "honest but curious," and it honestly executes instructions in the system, like storing user data, handling user's queries, and returning the results to the issuer. However, it is curious about the collected information of users and may try to determine or locate a query user.

The remainder of this paper is organized as follows. In Section 2, we briefly introduce the technical preliminaries, including Hilbert curve and OPE. Then, the overview of the system is stated in Section 3. In Section 4, we describe the system design, and the security analysis of our scheme is provided in Section 5. We conduct a set of experiments to evaluate the effectiveness of our scheme in Section 6. The related work is presented in Section 7. Finally, the conclusion and future work are discussed in Section 8.

2. Preliminaries

2.1. Hilbert Curve. The Hilbert curve is a space-filling curve [15], which visits each point once and once only in a specified order by some algorithms in 2-dimensional (2-D) or multidimensional space. Given 2-D square space with the resolution of $n * n$ divided into $2^N * 2^N$, $n = 2^N$ equal-sized cells, the N-order Hilbert curve recursively resolves the space into four equal-sized blocks by a defined way. For the N-order Hilbert curve, the iterative process is described as four subblocks that are replicated and partitioned after rotation and reflection from the (N-1)-order curve blocks.

According to the order of the curve traverses, the sequence number for each cell is determined, which ranges from 0 to $2^{2N} - 1$. The integer value associated with each cell is denoted as the Hilbert value (H-value). For example, Figure 1 shows the 2-order Hilbert curve goes through $2^2 * 2^2$ cells in the given space, and each cell is given a sequence number (H-value). The Hilbert curve is often used as one of the space transformation tools, transforming a 2-D space point into 1-D H-value. Similar to our previous work [3], we present each cell in the given square space by a grid coordinate <X, Y>, and the corresponding Hilbert value of each cell can be determined by a Hilbert curve.

Definition 1. Given a Hilbert curve visiting each cell in a 2-D square space, the grid coordinate of a cell $c < X_c, Y_c >$ can be transformed to 1-D Hilbert value by a function $\dot{f}$ as follows:

$$H(c) = \dot{f}(\langle X_c, Y_c \rangle), \tag{1}$$

where $\dot{f}$ is the spatial transformation function and $0 \le X_c, Y_c \le 2^N - 1, 0 \le H(c) \le 2^{2N} - 1$. Once the Hilbert

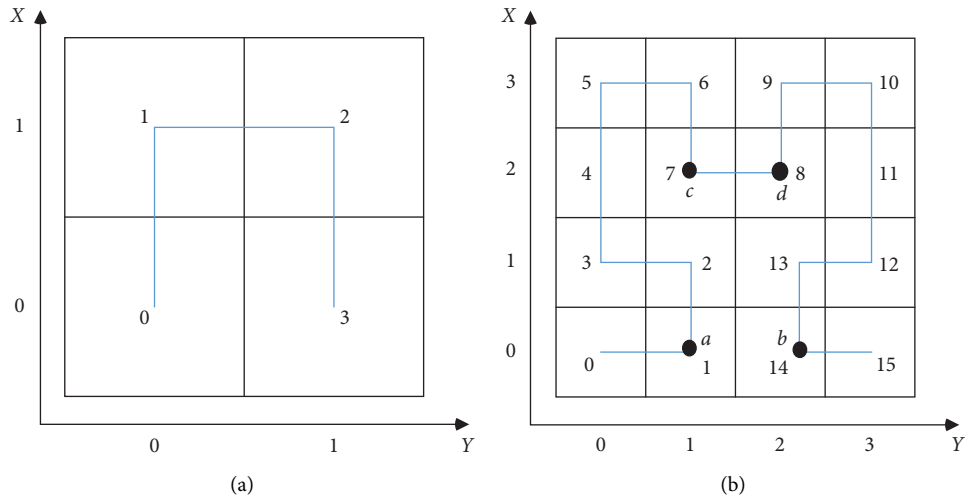

Figure 1: Hilbert curve: (a) 1-order; (b) 2-order.

curve setting parameters are determined, the generated H-values mapping to all cell are assigned. The setting parameters in the generating algorithm refer to the curve's starting point $<X_o, Y_o>$, curve order N, curve orientation Γ, and curve scale factor Θ. We term this parameter as spatial transformation parameter (STP), STP = $\{(X_o, Y_o), N, \Gamma, \Theta\}$. In our example, in Figure 1, users *a, b, c,* and *d* are with the grid coordinates of $\langle 1, 0\rangle$, $\langle 2, 0\rangle$, $\langle 1, 2\rangle$, and $\langle 2, 2\rangle$. As the 2-order Hilbert curve orderly visits each cell, generated Hilbert values of these users are 1, 14, 7, and 8, respectively.

In our study, we use the Hilbert curve to transform the user's real location for the purpose of privacy preservation. From the security aspect, the space transformation function $\dot{f}$ should be a one-way function, which is presented that it is easy to encode the elements of two-dimension space and generate one-dimension H-value but difficult to decode the one-dimension H-value back to the original space. The Hilbert curve has the above features and is suitable for our scheme. For an adversary without the STP, it is computationally impossible [15, 16] to reverse the function $\dot{f}-1$ and determine the correct Hilbert curve by comparing the H-values for all original spaces.

2.2. Order-Preserving Encryption. Order-preserving encryption (OPE) [17–20] is a deterministic symmetric encryption scheme which preserves the sort order of plaintexts in the ciphertexts. This character makes OPE very useful when performing range queries on encrypted data. It allows a remote untrusted database server to index the encrypted data and directly performs range queries without decrypting the ciphertexts [21] (returning a result in the database whose decryptions fall within a specified range, e.g., [a, b]).

Definition 2. Order-preserving function. For any $A, B \subset N$, $|A| < |B|$,function f: $A \longrightarrow B$ is order preserving, if for all $i, j \in A$, $f(i) > f(j)$, and only if $i > j$.

Definition 3. Order-preserving encryption. A deterministic encryption scheme $S\varepsilon = \{\mathcal{K}, E(\cdot), D(\cdot)\}$, with associated plaintext-space $\mathcal{D}$, ciphertext-space $\mathcal{R}$, and encryption key-space $\mathcal{K}$, for all $\mathcal{K}$, if $E(\cdot)$ is an order-preserving function from $\mathcal{D}$ to $\mathcal{R}$, the deterministic encryption scheme $S\varepsilon = \{\mathcal{K}, E(\cdot), D(\cdot)\}$ is order preserving.

The OPE-based scheme in our system consists of the following algorithms, where *EncL* and *EncR* are interactive, *KeyGen* is probabilistic, and the rest are deterministic:

Key Generation: KeyGen $(1^k) \longrightarrow SK$. The KeyGen runs on the user side, it takes the security parameters k as input and outputs a secret key *SK*.

Location Encryption: EncL$\{l, SK\} \longrightarrow l'$. Given a location point l, the user outputs the ciphertext l' with the secret key *SK*.

Range Encryption: EncR$\{R, SK\} \longrightarrow R'$. Given a 2-element vector R, $R = [l_1, l_2]$, the user outputs the ciphertext R′with the secret key *SK*.

Decryption: Dec$\{l', SK\} \longrightarrow l$. Dec is a decryption algorithm taking the ciphertext l' and secret key SK as inputs and outputs the plaintext l.

Detection: Det$R, l' \longrightarrow \{0, 1\}$. It takes the ciphertext R' and l' as inputs, performs the vicinity detection, and outputs 1 if l falls into the range R, otherwise, it outputs 0.

3. Overview of the System

The main idea of our scheme is to enable users to share location information and to find their nearby friends in a flexible and privacy-preserving way. Each user u in the system determines his current location (e.g., longitude and latitude) by a GPS-equipped device, which is a point in Euclidean space, presented by a 2-tuple value $(x_u, y_u) \in R$. The distance between the user u and other mobile users, e.g., v is d_{uv}, $d_{uv} = \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2}$. The user u in the system can specify a *vicinity region*, and we use the minimum bounding rectangle (MBR) to cover this region and determine it by the diagonal $\{(x_{V\min}, y_{V\min}), (x_{V\max}, y_{V\max})\}$.

The mobile user u can invite his friends v to join a *friend group* FG_u, and the invited user v may either accept or decline to join FG_u. If v accepts, the friendship relation F between u and v will be built as $\{(u, v)\} \in F$. In the same way, if user u is a friend of v, then users u and v are friends, and $\{(u, v), (v, u)\} \subseteq F$. The user u can manage the members of FG_u, and SP will maintain the friend list for him. The summary of notations used in the system is illustrated in Table 1.

3.1. System Model. The system architecture of the ULSS scheme is shown in Figure 2, which consists of the service provider, location-based social network server (LBSNS), and users. In the system model for our proposed protocol, the mobile user can be a request initiator or a participant (location provider). For ease of explanation, in the set of users, we assume Alice is the request initiator, and one of her friends Bob is the participant.

The mobile user Alice is an initiator who intends to share her real-time location in the MSNs and request the server for the proximity detection service. To preserve location privacy, Alice preprocesses the privacy information using the preloaded modules on her mobile device and then submits the encrypted location information to the server. Alice maintains different location policies according to the various contextual conditions, including the coarse granularity location privacy-preserving policy *Pa* and fine granularity location privacy-preserving policy *Pb*. With *Pa*, Alice can receive a user list from the server, and it shows the friends who are located over a large vicinity area (like a city) designated by her. Then, referring to some specified users on the received list, with *Pb*, Alice may query if they are in the nearby vicinity region (like around 2 km). The participant Bob is a friend of Alice. Bob follows the proposed protocol and the privacy policy of Alice. In the whole process, Bob only provides his encrypted location to the social network and preserves his real location information against to any other third party.

The service provider is a server in MSNs, which is an untrusted entity. Unlike the previous systems in [11, 22], which deploy the social network server and location server separately, in our architecture, the SP is an integrated server LBSNS. It provides users with two kinds of services, location-based services, and social network services. LBSNS achieves integrated service delivery, such as storing users' information and maintaining user friend list and offering dynamic user data update and adjustment. The LBSNS conducts proximity detection based on the encrypted locations of user and his friends and returns the results to mobile users based on their privacy-preserving policy.

3.2. Security Assumption. There are generally external attack and internal attack for the threat model [23]. The former is caused by unauthorized outsiders, and the latter is initiated by internal participant (for example, the users and LBSNS in our system). We assume the communication channels be secure by standard security schemes (such as SSL and SSH) or using cryptographic methods to resist internal attacks

Table 1: Summary of notations.

Notation	Description
(x_u, y_u)	Point coordinate of user u
$\langle X_u, Y_u \rangle$	Grid coordinate of user u
(x_u^o, y_u^o)	Origin of the grid user u located
(x_u^f, y_u^f)	Offset to the ordinate origin of user u
G_u	Grid identification number of the user u
$H(u)$	Hilbert value of the cell user u located
(X_o, Y_o)	Starting point of Hilbert curve

[24–26]. In the ULSS scheme, we consider only internal attack and assume conventional threat model in our system as follows: we consider the dishonest mobile users try to obtain other users' information outside the scope of the authorized access privileges. For example, Alice learns the exact location of Bob; Bob obtains the vicinity area or exact location of Alice. The untrusted mobile users may use some tools to forge their location, such attacks can be detected as presented in [27]. Besides, the LBSNS is assumed to be "honest but curious" by honestly following the proposed protocol in general but being curious about the content of queries, like the exact location and the real identity of users. In our system, LBSNS integrates location-based services and social network services on one server, which stores the information of all users, and social relationship network of them as well. In the ULSS scheme, all entities (including users and LBSNS) are assumed to be potential adversaries. The main security goal of our proposed scheme is to preserve user's location privacy against to all participants. In addition, the collusion between the malicious users and LBSNS is not allowed in our proposed ULSS scheme.

4. System Design

In this section, we first present the user registration and then describe the user location representation and update. In our system, according to user-defined location privacy-preserving policies, we use different methods to perform proximity detection: coarse-grained proximity detection and user-defined nearby friend detection as described in this section.

4.1. Registration. Before each user enjoys the location-sharing services in the social network, he should first register with his ID_U on the LBSNS, who verifies and maintains the uniqueness of the ID. Generally, users can use pseudonyms in the registration for safety considerations. The initiator Alice builds a *friend group* FG_A by inviting her friends to join in. The LBSNS will generate a group ID (like a random 256-bit-string) for the FG_A, storing the user list of it. Alice can classify the users on the list into several subgroups, like colleagues, families, classmates, and so on, defined as $\mathrm{C}_1, \mathrm{C}_2, \ldots \mathrm{C}_n$, $\mathrm{FG}_A = \{\mathrm{C}_1, \mathrm{C}_2, \ldots \mathrm{C}_n\}$ The LBSNS creates a friend relationship network for each user and updates dynamically and maintains the information of friend list as requested by users.

Each user should set a distance threshold D_{su} to preserve the location privacy during the application's initialization

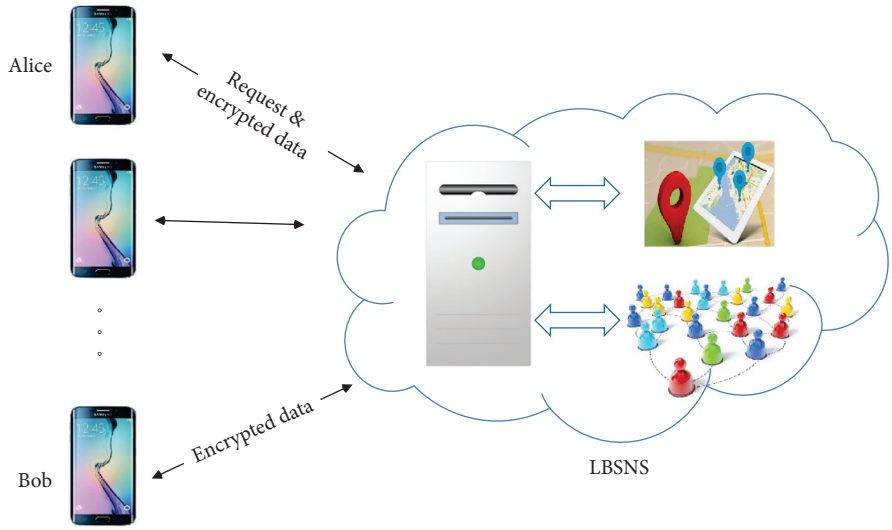

Figure 2: System architecture of ULSS scheme.

phase. D_{su} defines a minimum distance with which the user u allows his friends to search him. Otherwise, a user would be precisely located by the adversary continually reducing the range of vicinity detection. For example, Alice finds Bob somewhere around her by the vicinity detection service, and she can obtain the accurate location information of Bob according to continuous queries, such as by returning results when the range is set to 2 km, 1 km, 500 m, and so on. The SP will build a profile Pr for each user to keep these settings, e.g., for Alice, $Pr_A = (ID_A, PSW_A^*, FG_A, D_{SA}\}$, where ID_A is the ID of Alice, PSW_A^* is the ciphertext of the password, FG_A is the friend group of Alice, and D_{SA} is the distance threshold. The ULSS scheme allows users to customize and personalize their profiles to satisfy different users' needs.

4.2. Location Representation and Update. Each user u obtains his location in the forms of latitude-longitude by the GPS $(\text{lat}_u, \text{lon}_u)$. The ULSS scheme enables the user to transform his location by the Hilbert curve method as described in Section 2. Thus, the system first assigns a space range, where mapping the geolocation into a coordinate point denoted by $(x_u, y_u) \in \mathscr{R}$ in Euclidean space. For example, the system can define the square scale using the longitude and latitude of 4 points of east, west, north, and south. As shown in Figure 1, the given space range is divided into $2^N * 2^N$ equal-size cells and formed a grid system. Each grid in the space is numbered uniquely from $\mathbb{N}$. To protect the location privacy, each user in the ULSS scheme transforms his location coordinate (x_u, y_u) into two parts. The first part (x_u^o, y_u^o) is the origin coordinate of the grid where he is located in the grid system, the second part (x_u^f, y_u^f) is the offset coordinate in the grid, and $x_u = x_u^o + x_u^f$, $y_u = y_u^o + y_u^f$. We assume that the location coordinate range is separated into N scale levels, the coordinates of a user can be represented as $G_u \| (x_u^f, y_u^f)$. Here G_u is the grid identification number of the user u in the N-th scale level, and $0 \le G_u \le 2^{2N} - 1$, (x_u^f, y_u^f) is the offset coordinates in the corresponding grid. In the ULSS scheme, if the transformation parameter STP is defined, the H-values mapping to all cell are assigned, which means the G_u can be obtained by the H-value according to equation (1).

In general, if a user's motion or movement trajectory is in a small range (e.g., in a city), the origin coordinates of the grid in the given space are not changed, and the user can specify a time interval with which the user automatically updates his offset coordinate. While user traverses a large area, e.g., traverse different grids in the grid system, the changes in offset coordinates exceed a certain degree (e.g., greater than the unit length of a grid), and the user should recalculate his location $G_u \| (x_u^f, y_u^f)$ and submit related information to the server. In this way, we can reduce the computation and communication overheads for the user and improve the performance of the whole system.

4.3. Coarse-Grained Proximity Detection. In this phase, a user queries an LBSNS and gets the results showing friends in the coarse-grained vicinity area (such as in the same city). The main idea of coarse-grained proximity detection is to utilize the Hilbert curve method to transform user's location and the coordinates of vicinity. The scheme cloaks user and his vicinity by mapping 2-D points into identification numbers in corresponding scale level which can be presented by H-value under given STP. The LBSNS provides results by comparing the hash values of all values submitted by mobile users. The processes are shown in Figure 3, and the detailed steps are as follows:

Step 1. When the user Alice logs onto the system and wants to view the friends who are located in the coarse-grained vicinity area, she can submit a query $Q_A = \{ID_A, list_A, N\}$ to the SP, where ID_A is the login ID of Alice, which can be pseudonymous; $list_A$ is the user list in the friend group specified by Alice, which can be a classification of his friends. Actually, Alice can search

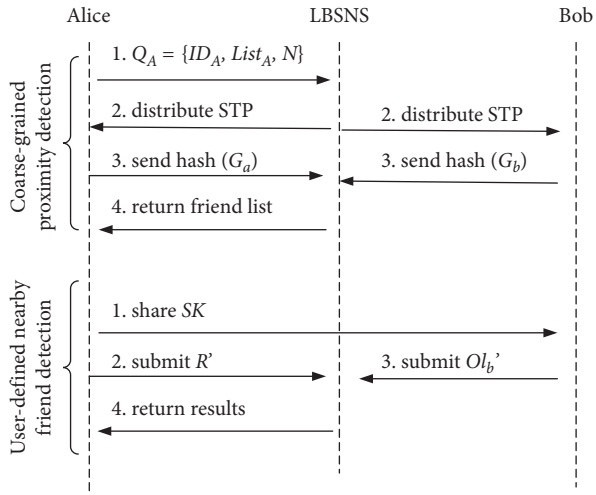

Figure 3: Processes in the ULSS scheme.

several classifications in his friend group FG_A, like $\{C_i, C_{i+1}, \ldots, C_j\}$, $1 < i, j < n$, for simplicity, and we denote it by $list_A$. According to the coarse granularity location privacy-preserving policy *Pa*, Alice sets the granularity *N*. For example, Alice wants to search the friends in the $list_A$ who are located over a large vicinity area, e.g., the same city she stays. In the given space range, Alice assigns the appropriate granularity *N* so that the vicinity can be presented by a grid number in the Hilbert transformation. In the Hilbert STP, *N* is the scale level $1 \le N \le 16$. Generally, the larger the *N*, the bigger the vicinity area.

Step 2. Receiving upon the request from an initiator Alice, the LBSNS generates the STP according to the Hilbert curve method for users. In the STP, the scale level *N* is defined by Alice. The system determines longitude and latitude coordinate ranges as curve scale factor Θ, mapping it to the points in the given space range. In this way, the given 2-D space can be transformed to be a grid system with the equal size of $2^N * 2^N$ grids. Then, the SP randomly selects the curve's starting point $<X_o, Y_o>$, and curve orientation Γ to build the STP and determine the Hilbert curve. Next, the LBSNS searches the friend list according to the $list_A$ submitted by Alice and distributes the generated STP to Alice and users on the list.

Step 3. When receiving STP, each user transforms his location point (x_u, y_u) to $G_u \| (x_u^f, y_u^f)$, and the process is as follows: Alice first maps her point location in Euclidean space (x_a, y_a) into the $2^N * 2^N$ grids system $<X, Y>$. For example, if the square space range Θ is determined by 4 points of east, west, north, and south, (x_E, y_E), (x_W, y_W), (x_N, y_N), and (x_S, y_S), the unit length *Unit* of each grid can be presented by $(x_W — x_E)/2^N$ or $(y_S — y_N)/2^N$. With the site (x_a, y_a), Alice can locate her grid coordinates $<X_a, Y_a>$ in the grid system by calculating $X_a = [(x_W — x_a)/Unit]$ and $Y_a = [(y_a — y_S)/Unit]$ and also gets the offsets (x_a^f, y_a^f) in the corresponding grid. Then, she transforms the grid coordinates $<X_a, Y_a>$ to a Hilbert value $H(a)$ using equation (1) by the determined STP. Now Alice obtains the $G(a)$ and can transform her location (x_a, y_a) into $G_a \| (x_a^f, y_a^f)$. In order to prevent other entity getting real location information, Alice submits the hash value of it, $hash(G_a)$, to the LBSNS, where $hash(\cdot)$ is a collision-resistant hash function. In this way, Alice cloaks her exact location into a region, presented by $hash(G_a)$. The friends of Alice, e.g., Bob also transforms his locations and sends the $hash(G_b)$ to the LBSNS in the same way.

Step 4. The LBSNS collects the transformed location information from Alice and her friends. Then, it searches the users who have the same $hash(G_u)$ as Alice and returns the neighboring users located in the vicinity to Alice under the coarse-grained setting *Pa*.

4.4. User-Defined Nearby Friend Detection. When Alice receives the results of the coarse-grained proximity detection from LBSNS, she can further select some specified users she has interest to perform nearby friend detection under the fine granularity policy *Pb*. For example, referring to some particular users on the received list, presented by $list_B$, Alice may query if they are in the nearby vicinity region (like around 2 km). Note that the neighborhood range in this phase can be customized arbitrarily by Alice under the constrain conditions of security assumption. We intend to use lightweight OPE technology to implement the user-defined nearby friend detection, and the processes are as follows:

Step 1. Alice generates an encryption key *SK* according to the OPE key generation algorithm $\text{KeyGen}(1^k) \longrightarrow SK$ and shares the key through the secure channel to the friends to be queried.

Step 2. Alice customizes a neighborhood range to retrieve the nearby friends. For example, the range around her *A* meters or the range with *A* and *B* meters in the *x*-axis and *y*-axis directions. The scheme generates a rectangle *R* to cover this range, $R = [r_l, r_u]$, where $r_l = (x_a - A, y_a - B)$ and $r_u = (x_a + A, y_a + B)$ denote the lower-left corner and upper-right corner of the vicinity range *R*, respectively. Notice that, here we use the offset coordinates in the corresponding grid to present the vicinity *R*. Then, Alice encrypts *R*, by $\text{EncR}\{R, SK\} \longrightarrow R'$.

Step 3. When a specific user Bob receives the request, he encrypts his own location before submitting. The location point of Bob l_b is presented by $G_b \| (x_b^f, y_b^f)$. Here, Bob only needs to send his encrypted offset coordinates to further ensure the safety of his data. In this way, even some malicious attacker can decrypt the ciphertext, he has no way to obtain the real l_b without G_b. If we denote the offset coordinates of Bob by Ol_b, Bob inputs it to the encryption function $\text{EncL}\{Ol_b, SK\} \longrightarrow Ol_b'$ and submits Ol_b' to the LBSNS to avoid potential security risks from trusted third party (TTP) free server.

Step 4. LBSNS performs the range query on the ciphertexts directly and returns the results whether l' falls into the associated range by $\mathrm{Det}(R', l') \longrightarrow \{0, 1\}$. LBSNS collects the information from all users of $list_B$ conducting the proximity detection and returns those who is in the nearby vicinity R to Alice.

5. Security Analysis

The ULSS scheme uses the Hilbert curve method to perform coarse-grained proximity detection and utilizes the OPE method to conduct user-defined nearby friend detection. We give the security analysis for these two technologies. The security assumption and requirements are described in Section 3.2.

5.1. Hilbert Curve Transformation. In general, the privacy protection methods in most literatures are public, and the technology of Hilbert curve in the ULSS scheme may be obtained by some adversaries. Once an attacker learns some background knowledge of the users' original locations or their transformed locations, they may guess the secret key (e.g., STP in the ULSS scheme) with a certain probability. However, some researchers in [15, 16, 28] stated that it is computably impossible to get the correct STP by comparing H values of all locations, and it can resist brute force attacks.

Brute-Force Attack. In STP, we represent curve's orientation Γ in terms of q bits, and it will generate 2^q values in the entire 360° space. The malicious adversary needs to make 2^q attempts to determine the right Γ. Refer to the curve's starting point (X_0, Y_0), and we represent the value with q bit data, respectively. To search for the correct (X_0, Y_0) over $2^q * 2^q$ elements, it requires the adversary to try $2^q * 2^q$ candidate coordinate values on the X and Y axes. In our scheme, the square space range Θ is fixed value, and the order of the Hilbert curve N is determined by users. Therefore, we have $(N * 2^q * 2^q * 2^q)$ solutions for the entire space, where q is the number of bits presenting each parameter of STP. N is the order of the Hilbert curve, and $1 \leq N < 16$. If the value of q is big enough, the possibility of getting the correct STP by trying combinations of parameters is negligible. For example, if $q = 32$, the complexity of exhaustive search for the STP would be $O(2^{3*32})$.

Therefore, for anyone without the STP, it is computationally impossible to reverse the process of the spatial transformation and get the users' original locations. The one-way mapping of the Hilbert curve makes the transformation function $\dot{f}(\langle X_c, Y_c \rangle)$ be a secure encryption function [15, 16]. Any adversary has no way to learn the user's real location without the encryption key STP. We can say that the Hilbert curve transformation is a very appropriate approach to preserve user location privacy in our ULSS scheme.

5.2. Security of Order-Preserving Encryption. The researchers in [29, 30] formalized a security requirement for OPE and proposed an efficient blockcipher-based scheme provably meeting their security definition. The OPE scheme has been proven [29] that it is *security under distinct chosen-plaintext attack* (IND-DCPA). Popa et al. [31] presented that the ideal security guarantee for order-preserving encryption is to reveal no additional information about the plaintext values besides their order. They proposed an ideal security protocol for OPE, which can achieve *indistinguishability under ordered chosen-plaintext attack* (IND-OCPA) security. The ULSS scheme utilizes the OPE method to conduct user-defined nearby friend detection. Alice and her friends hold the secret key SK, derived from the security of the OPE scheme, and any other entities including the LBSNS without the key cannot get the real location of users. Therefore, the security of user-defined nearby friend detection in the ULSS scheme can be achieved from that of the OPE scheme.

6. Evaluation

In this section, the efficiency and effectiveness of our ULSS scheme were evaluated by the extensive experiments. During the phase of coarse-grained proximity detection, we focus on the location transformation using the Hilbert curve method, which is mainly affected by the parameter of Hilbert order N. During the phase of user-defined nearby friend detection, we focus on the location encryption by OPE, which is mainly affected by the size of data. We also conducted experiments to evaluate the computational and communication overheads of each entity under different system settings, like the size of vicinity s or number of friends n.

All experiments were conducted on Java Development Kit (JDK).1.7 with the Intel (R) Core (TM) i7-7500U 2.70-GHz CPU and Win 10 OS. We used the 256-bit OPE symmetric encryption algorithm and 256-bit SHA hash function to ensure data confidentiality and integrity. The cryptographic algorithms were performed with JPBC library. In the experiments, we used the Web API of Amap [32] to get the GPS positioning by the function *GetLongitudeAndLatitude* and initialized the location of each user by calling function *GetCurrentPosition*. The studies utilize uniform data sets to randomly generate batches of 10,000 user locations and then map them into the points in a $65536 * 65536$ area in 2-D space. We assume each mobile user stores his friend list on the LBSNS, and each user has a maximum of 2,000 friends. Each mobile user can specify an arbitrary vicinity to search for his nearby friends. For the security and privacy issue, the minimum size of vicinity is set to $1\,\mathrm{km}^2$, and we got the average data from 20 experiment results.

6.1. Hilbert Curve Order. In the phase of coarse-grained proximity detection, Alice and her friends use the Hilbert curve to transform their exact locations into a cloaked region, presented by an H-value. The LBSNS gives the results of coarse-grained detection according to the hash values submitted by all users, and the computational overhead of the LBSNS is almost negligible. In this set of experiments, we mainly evaluated the processing time on the user side. When performing the Hilbert curve transformation, curve's

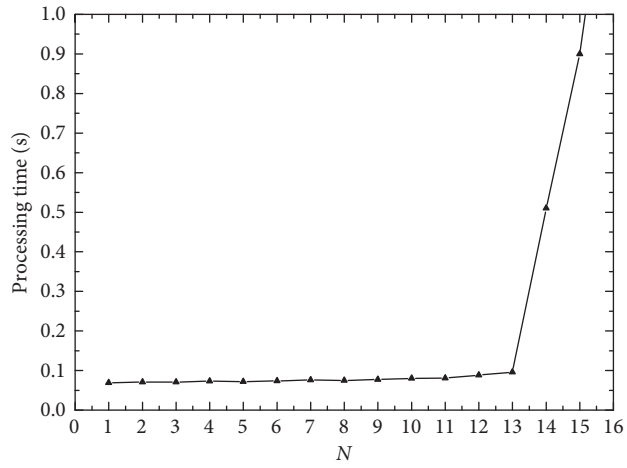

Figure 4: Computational overhead of location transformation for the user.

starting point is set to $\langle 0, 0\rangle$ and the curve orientation Γ is set to be clockwise, the square space range Θ is determined by 4 points of east, west, north, and south in China. Figure 4 shows the average computational cost during the location transformation with various Hilbert curve order N for the user. As shown by it, the computational processing time of the user slightly increases as $N \leq 13$, and the transformation time is no more than 0.2 second. In fact, in our ULSS scheme, if $N \geq 16$, the generated region is too small to cloak user's exact location. Generally, in the initialization phase, we assign the value of N is no bigger than 15. After transformation, each user submits the hash value of his transformed location to the LBSNS to prevent the information leakage on the third entity. Table 2 shows that this operation completes within several milliseconds. Thus, in practical application of the scheme, for the user, we can say the Hilbert curve transformation is computationally efficient.

6.2. Encryption and Decryption. In the phase of user-defined nearby friend detection, we used the 256-bit OPE symmetric encryption algorithm to ensure the data confidentiality for users. Table 2 shows the computational overheads of the encryption and decryption in various data sizes. In the ULSS scheme, the initiator Alice encrypts her vicinity range R, and her friend Bob encrypts his offset coordinates (x_b^f, y_b^f), the size of them is no more than 64 bytes. For each user, we can see, from Table 2, the average processing time for the encryption is about 13 ms. For the LBSNS, it conducts the nearby friend detection directly on the ciphertexts since the OPE preserves the sort order of plaintexts in the ciphertexts. It means that, without decrypting any ciphertext, the LBSNS can return a result in the list whose decryptions fall within a specified range R. This process can be completed by comparing the value of ciphertexts, which ensures minimal overhead on the LBSNS.

Table 2: Computational overhead of encryption and decryption.

Size (bytes)	32	64	128	1024	10 k	1 M
Hash (ms)	6	6	8	9	11	14
Encryption (ms)	11	13	21	122	136	2049
Decryption (ms)	10	11	29	130	522	43822

6.3. Number of Friends. We gave an experiment to evaluate the efficiency of the ULSS scheme in terms of various parameters, e.g., number of friends n and the size of vicinity s. We illustrated the experimental results by the computational time and traffic overheads from the Alice, Bob, and the LBSNS, respectively. We assumed Alice submits a coarse-grained proximity query to LBSNS, upon receiving the results, and she selected half of friends from the list to further put forward nearby friend query. In this process, we ignored the communication network transmission delay, as it was impacted by different network environments. Figure 5 shows the computational time of each entity in the scheme when the number of friends n changes from 20 to 2000, and we set the default vicinity was 2 km^2. The processing time is 92–98 ms for Alice and 80–85 ms for Bob. While it grows for LBSNS with number of friends, because the server should handle the proximity detection requirement from each participant, the greater the number of friends is, the longer the time it takes. Similarly, LBSNS needs to distribute STP to Alice and her friends, and its communication cost grows linearly with n (as shown in Figure 6). During the phase of user-defined nearby friend detection, Alice should share the encryption key to her friends, the traffic also grows, and the communication cost of Bob maintains at a very low level.

6.4. Size of Vicinity. In this set of experiments, the computational and communication cost of each entity in the ULSS scheme is evaluated when the size of vicinity s changes from 1 km^2 to 20 km^2. We assumed Alice first specified the Hilbert order to 5, when she got the results in the coarse-grained region, she defined a nearby vicinity (the size is s) to search for proximity friend. Figures 7 and 8 show that the computation and communication overheads for Alice, and SP slightly grow with increasing vicinity area, while it holds constant for Bob.

6.5. Comparison with Other Approaches and Discussion. We compared our proposed scheme with other similar approaches for a holistic evaluation of our system. Table 3 shows the results in terms of TTP, server, cryptography method, and efficiency characteristics when comparing with approaches like MLS in 2017 [11], VPPLS in 2017 [13], and UDPLS in 2017 [22].

From the table, we find that some approaches, e.g., the VPPLS scheme, need a fully trusted third party (TTP) to verify the user ID and run some cryptographic computation. In the TTP-based schemes, the third party stores user's sensitive information and some important information of the system, which may put the system in jeopardy if it is comprised. However, our scheme uses the location

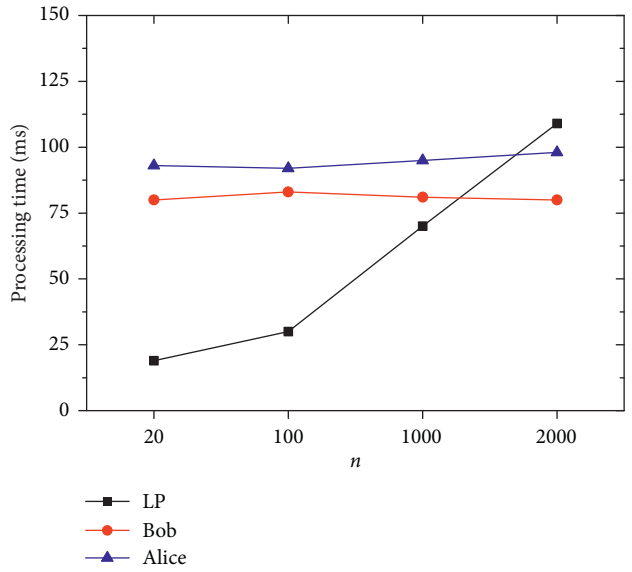

Figure 5: Computational overhead with various numbers of friends.

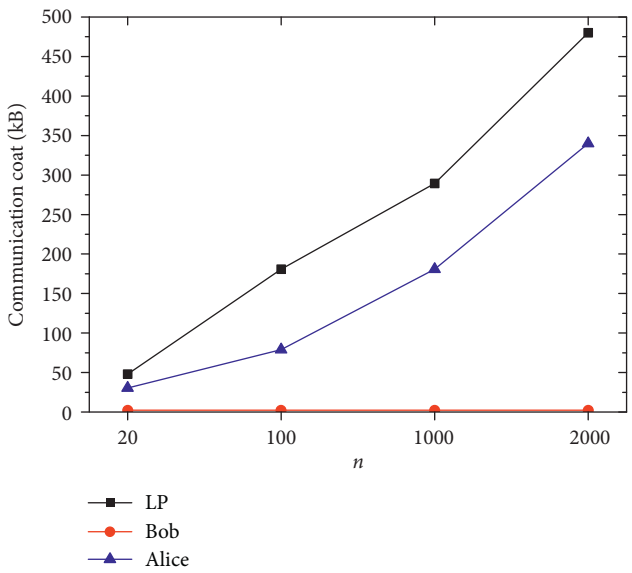

Figure 6: Communication cost with various number of friends.

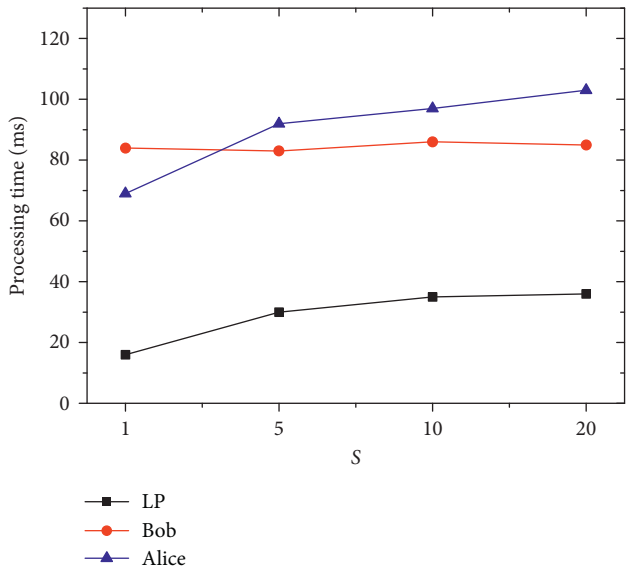

Figure 7: Computational cost with various sizes of vicinity.

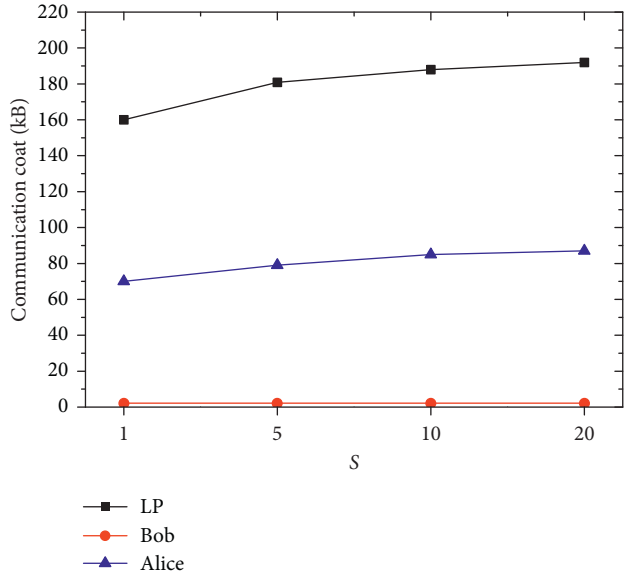

Figure 8: Communication cost with various sizes of vicinity.

transformation and OPE method to protect user privacy getting rid of the introduction of TTP.

Some exiting solutions, e.g., MLS and UPDLS, separately deploy two different servers including location server and social network server to store information and provide users two kinds of services, location-based services, and social network services. The main drawback of LBS/SNS separated architectures is the multiple-round interactions between the servers, which may incur higher communication cost and cause great computation and storage burden as well. In practice, mobile users can enjoy their online social network services by the wireless mobile network and, meanwhile, obtain their exact location information through the GPS equipped in the smart terminals. Nowadays, location sharing and services are important functions of MSNs, and integrating two kinds of services can bring more convenient and flexibility for users, which has become an indispensable tool of users' daily life. In security aspect, in the LBS/SNS separated architectures, the user's location information and user profiles are separately stored in two servers, and they are assumed to be no-collusion to guarantee the system safety. However, commonly, in some location-sharing scenarios, the location information can be easily captured by adversaries, if the server providers collude with them or collude with each other, the user privacy and system safety are insecure, which makes these distributed architectures unpractical in real life. In our model, these two kinds of servers

Table 3: Comparison of our proposed with other protocols.

Protocol	TTP	Server	Cryptographic methods
MLS [11]	No	Separately deployed LS and SNS	AES and RSA signature
VPPLS [13]	Yes	Integrated LBSNS	Homomorphic encryption
UDPLS [22]	No	Separately deployed LS and SNS	AES and RSA signature
Our scheme	No	Integrated LBSNS	OPE

are combined by one LBSNS, providing integrated service delivery.

In terms of cryptographic methods, the VPPLS protocol used the homomorphic encryption method to compute the distance between Alice and her friends without disclosing the exact position to other party, and MLS and UDPLS rely on symmetric encryption AES and asymmetric encryption RSA signature to ensure the data integrity and confidentiality. These privacy preservation approaches can provide user location sharing privately and securely. However, the main concern is the heavy computing burden both on user and server sides when running the encryption algorithms. Different with the traditional encryption method, our scheme utilizes efficient OPE algorithm to protect user data, and the ULSS only needs to encrypt very small amount of user data, in our evaluation result, Table 2 shows that user only takes several milliseconds to complete the encryption process. Besides, for the LBSNS, it can directly conduct the proximity detection by comparing the ciphertexts, without having to decrypt, which can greatly save the computing and communication resources. Thus, we can deduce that our scheme is more efficient.

Discussion. In our proposed scheme, we presented two methods, Hilbert curve-based proximity detection method and lightweight order-preserving encryption- (OPE-) based method, to provide the users with flexible privacy preservation in an efficient and friendly way. The evaluation results of Hilbert curve transformation show that the processing time on the user side and server side is very small. And from the experiments with respect to order-preserving encryption, we can also find that the average processing time of users is several milliseconds. Meanwhile, due to OPE, the server can directly perform range queries without decrypting the ciphertext, which minimizes the overhead on it. Compared with the homomorphic encryption- (HE-) based method [13] and AES&RSA-based method [11, 22], we can state that our ULSS is more efficient and practical. From the aspect of security, in some existing solution, the privacy of users is guaranteed based on the assumption of fully trust in the service provider or TTP. However, the TTP-based structure results in trust issues, like the single-point failure and the bottle-neck problem of communications, and if the TTP is comprised, the security of the entire system is at risk. Our scheme is TTP-free architecture, which can protect user location privacy with respect to SP, friends, and other adversaries. We also prove that the proposed scheme is secure under the stronger security model with enhanced privacy. Moreover, unlike the multiserver structure in [11, 22], in our scheme, the LBS and SNS are integrated by one server to provide entire and convenient services, and it also can reduce communication and computational overheads and security risks as well.

7. Related Work

7.1. Location-Based Services. With the popularity of LBS, the concern of privacy leakage in LBS raises, and many researches for privacy preservation have been proposed. K-anonymity [33] is one of the popular technologies to solve the privacy leakage issue in LBS, which employs a trusted third party (TTP) called *anonymizer* to replace the exact location of the user by a cloaked area including at least K users so that the user location is indistinguishable from $K-1$ other locations. Based on this fundamental idea, researchers [34–37] proposed efficient methods and models to construct K-anonymity spatial region (K-ASR) to protect user privacy. The authors in [36] proposed a location privacy-preserving K-anonymity method based on the credible chain, in which the optimal K value for the user is determined according to the user's environment and social attributes. The authors in [37] proposed a privacy scheme through caching and spatial K-anonymity (CSKA) and utilized the Markov model to predict the next query location according to the user mobility in continuous LBS. The TTP-based schemes, to some extent, solve the problem of privacy leakage in LBS. However, the introduction of the *anonymizer* in these schemes actually transfer users' trust from the SP to the intermediate entities. If it is compromised by the adversaries, it will pose a serious threat to users. Thus, these TTP-based schemes can only provide limited security assurance. In order to address the problem, in our previous work [38], we designed a privacy-preserving scheme based on location transformation getting rid of the fully trusted entities to provide enhanced security. The *anonymizer* can provide users accurate results without knowing any information about a user's real location.

The TTP-free schemes were adopted in the distributed peer to peer (P2P) environment to protect user privacy. Montazeri et al. in [39] introduced an information-theoretic notion for location privacy offering two models in both snapshot LBS and continuous LBS. Huang et al. [40] proposed a multimodal Bayesian embedding model (MMBE) for point-of-interest (POI) recommendation on location-based cyber-physical social networks. Sangaiah et al. used machine learning techniques to propose a method for conserving position confidentiality of roaming PBSs users in [41]. The authors in [42] considered the privacy and utility requirements of each user to propose an optimal user-centric location obfuscation mechanism. Sun et al. [43] introduced the location-label based (LLB) algorithm to distinguish locations of mobile users to sensitive and

ordinary locations and designed three protocols to protect user privacy for different user environments. Zhang et al. [9] proposed a deviation-based query exchange (DQE) scheme to obfuscate the users' query point to mitigate trajectory disclosure in MSNs. In our previous work [44], we presented a collaborative trajectory privacy-preserving (CTPP) scheme for continuous LBS queries, in which trajectory privacy is guaranteed by caching-aware collaboration between users. The main idea is to spatiotemporally break the correlations of continuous LBS queries to prevent the adversary from reconstructing a user's actual trajectory. The main drawback of TTP-free scheme is that multiple-round interaction between the servers and the user may cause a higher communication cost and may incur a higher computation overhead on the user side.

In addition, differential privacy technology [45–47], mix-zone method [48, 49], and encryption-based methods are also adopted to protect user privacy in LBS.

7.2. Location-Sharing Service. Our paper briefly focuses on a popular LBS location-sharing service, which enables users to share their current locations by the GPS-enabled devices to their friends on the MSNs or to find whether any friends are within a given vicinity area. It makes a large number of social network applications by the virtue of the smartphone and MSNs. Previous research studies [11, 12] discussed the issue of privacy preservation for the location-sharing service. In order to protect location privacy and social network privacy of mobile users, Li et al. [11] proposed a location-sharing construction with multiple location servers, in which the user's friend set was divided into multiple subsets randomly. Zheng et al. [2] presented a location tag construction method by environmental signals to provide an unforgeable location proof and used Bloom filters to efficiently represent users' location tags and vicinity regions. In the spatial-generalization-based method [50, 51], the user location space is divided into grids, and the precise position of user is replaced by the generalized grid prior to sending to SP. The authors in [50] proposed two protocols named "Hide and Seek" and "Hide and Crypt" based on spatial generalization to offer private proximity detection. Based on the "one degree" grid, Jing et al. in [51] presented a flexible and private proximity testing (FPODG) protocol. However, in the existing solution, the privacy of users can hardly be guaranteed without the assumption of fully trust in the service provider. Homomorphic encryption- (HE-) based privacy-preserving location-sharing scheme methods [13, 14] allow mobile users to compute distances between them and their friends without knowing the exact locations of each other. Based on HE, Tang and Cai proposed privacy-preserving location-sharing scheme (VPPLS) to build two models to enable users to query their friends' location in a secure way, which allowed user to make a classification of his friends and provide a verification for query result. The main drawback of these methods is the large computational and communication cost of the system, which makes the scheme less practical. Li and Jung [52] used the technology of ciphertext policy attribute-based encryption (CP-ABE) to design fine-grained privacy-preserving location query protocol (PLQP), and it satisfies different levels of location query and realize fine-grained and multileveled access control. These cryptographic-based methods can provide strong privacy guarantee for users at the high expense of computation and traffic. We proposed a lightweight privacy-aware friend locator (PAFL) in our previous work [3] to provide privacy guarantee for the user in an efficient way. However, the coarse-grained friend locator method cannot meet the personalized needs of mobile users. In this paper, we proposed a flexible ULSS scheme for private proximity detection, in which each user can maintain his own privacy-preserving control policy.

8. Conclusion

In this paper, we proposed user-defined location-sharing scheme in mobile social networks to protect user privacy in proximity detection service. We proposed two protocols, coarse-grained proximity detection and lightweight user-defined nearby friend detection to realize proximity query of users in different scales of vicinity regions. Our scheme is TTP-free architecture, in which the location-based services and social network services are integrated by one LBSNS server, making it more practical and convenient. Experimental results suggest that the ULSS scheme consumes low computational and communication overheads even for a large size of vicinity area and mount of friends. In our scheme, we used a minimum bounding rectangle to cover the user-specified vicinity region, which affects the accuracy of the results. In the future, we will propose an approach which can support the irregular shape vicinity to deliver services in more accurate ways.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant nos. 61802076, 61632009, and 61872097), Guangdong Provincial Natural Science Foundation (Grant no. 2017A030308006), High-Level Talents Program of Higher Education in Guangdong Province (Grant no. 2016ZJ01), and CERNET Innovation Project (Grant no. NGII20190408), and CERNET Innovation Project (Grant no. NGII20190408).

References

[1] H. Li, H. Zhu, S. Du, X. Liang, and X. Shen, "Privacy leakage of location sharing in mobile social networks: attacks and defense," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 646–660, 2018.

[2] Y. Zheng, M. Li, W. Lou, and T. Hou, "Location based handshake and private proximity test with location tags," *IEEE Transactions on Dependable & Secure Computing*, vol. 14, no. 4, pp. 406–419, 2017.

[3] T. Peng, Q. Liu, G. Wang, and J. Chen, "A lightweight privacy aware friend locator in mobile social networks," in *Proceedings of the 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)*, pp. 17–23, Guangzhou, China, December 2017.

[4] Q. Liu, P. Hou, G. Wang, T. Peng, and S. Zhang, "Intelligent route planning on large road networks with efficiency and privacy," *Journal of Parallel and Distributed Computing*, vol. 133, pp. 93–106, 2019.

[5] Y. Wu, H. Huang, Q. Wu, A. Liu, and T. Wang, "A risk defense method based on microscopic state prediction with partial information observations in social networks," *Journal of Parallel and Distributed Computing*, vol. 131, pp. 189–199, 2019.

[6] Y. Wu, H. Huang, N. Wu, Y. Wang, M. Z. Alam Bhuiyan, and T. Wang, "An incentive-based protection and recovery strategy for secure big data in social networks," *Information Sciences*, vol. 508, pp. 79–91, 2020.

[7] D. V. Medhane, A. K. Sangaiah, M. S. Hossain, G. Muhammad, and J. Wang, "Blockchain-enabled distributed security framework for next generation IoT: an edge-cloud and software defined network integrated approach," *IEEE Internet of Things Journal*, vol. 7, pp. 1–8, 2020.

[8] Q. Liu, G. Wang, F. Li, S. Yang, and J. Wu, "Preserving privacy with probabilistic indistinguishability in weighted social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 5, pp. 1417–1429, 2017.

[9] S. Zhang, G. Wang, Q. Liu, and J. H. Abawajy, "A trajectory privacy-preserving scheme based on query exchange in mobile social networks," *Soft Computing*, vol. 22, no. 18, pp. 6121–6133, 2018.

[10] E. Luo, K. Guo, Y. Tang, X. Ying, and W. Huang, "Hidden the true identity and dating characteristics based on quick private matching in mobile social networks," *Future Generation Computer Systems*, vol. 109, pp. 633–641, 2020.

[11] J. Li, H. Yan, Z. Liu, X. Chen, X. Huang, and D. S. Wong, "Location-sharing systems with enhanced privacy in mobile online social networks," *IEEE Systems Journal*, vol. 11, no. 2, pp. 439–448, 2017.

[12] Z. Liu, D. Luo, L. Jin, X. Chen, and C. Jia, "N-mobishare: new privacypreserving location-sharing system for mobile online social networks," *International Journal of Computer Mathematics*, vol. 93, no. 2, p. 384400, 2014.

[13] C. Tang and C. Cai, "Verifiable mobile online social network privacy preserving location sharing scheme," *Concurrency & Computation Practice & Experience*, vol. 29, no. 1, pp. 1–10, 2017.

[14] E. Novak and Q. Li, "Near-pri: private, proximity based location sharing," in *Proceedings of the IEEE Infocom-IEEE Conference on Computer Communications*, pp. 31–45, Toronto, Canada, April 2014.

[15] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Proceedings of the International Conference on Advances in Spatial & Temporal Databases*, Boston, MA, USA, July 2007.

[16] I. Kamel, A. M. Talha, and Z. A. Aghbari, "Dynamic spatial index for efficient query processing on the cloud," *Journal of Cloud Computing*, vol. 6, no. 1, pp. 1–16, 2017.

[17] P. Grubbs, K. Sekniqi, V. Bindschaedler, M. Naveed, and T. Ristenpart, "Leakage-abuse attacks against order-revealing encryption," in *Proceedings of the Security and Privacy (SP), 2017 IEEE Symposium on. IEEE*, pp. 655–672, San Jose, CA, USA, May 2017.

[18] D. S. Roche, D. Apon, S. G. Choi, and A. Yerukhimovich, "Pope: partial order preserving encoding," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM*, pp. 1131–1142, Vienna, Austri, October 2016.

[19] F. Kerschbaum, "Frequency-hiding order-preserving encryption," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM*, pp. 656–667, Denver, CO, USA, October 2015.

[20] F. Kerschbaum and A. Schropfer, "Optimal average-complexity idealsecurity order-preserving encryption," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM*, pp. 275–286, Scottsdale, AR, USA, November 2014.

[21] Q. Liu, Y. Tian, J. Wu, T. Peng, and G. Wang, "Enabling verifiable and dynamic ranked search over outsourced data," *Transactions on Services Computing*, 2019.

[22] S. Gang, Y. Xie, L. Dan, H. Yu, and V. Chang, "User-defined privacy location-sharing system in mobile online social networks," *Journal of Network & Computer Applications*, vol. 86, pp. 34–45, 2017.

[23] Y. Xu, Z. Quanrun, G. Wang, C. Zhang, J. Ren, and Y. Zhang, "An efficient privacy-enhanced attribute-based access control mechanism," *Concurrency and Computation: Practice and Experience*, vol. 32, pp. 1–10, 2020.

[24] Q. Liu, G. Wang, X. Liu, T. Peng, and J. Wu, "Achieving reliable and secure services in cloud computing environments," *Computers & Electrical Engineering*, vol. 59, pp. 153–164, 2017.

[25] T. Peng, Q. Liu, B. Hu, J. Liu, and J. Zhu, "Dynamic keyword search with hierarchical attributes in cloud computing," *IEEE Access*, vol. 6, 2018.

[26] A. K. Sangaiah, D. V. Medhane, G.-B. Bian, A. Ghoneim, M. Alrashoud, and M. S. Hossain, "Energy-aware green adversary model for cyber physical security in industrial system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3322–3329, 2020.

[27] W. He, X. Liu, and M. Ren, "Location cheating: a security challenge to location-based social network services," in *Proceedings of the International Conference on Distributed Computing Systems*, San Jose, CA, USA, June 2011.

[28] T. Peng, Q. Liu, G. Wang, Y. Xiang, and S. Chen, "Multidimensional privacy preservation in location-based services," *Future Generation Computer Systems*, vol. 93, pp. 312–326, 2019.

[29] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *Proceedings of the 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Cologne, Germany, April 2009.

[30] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: improved security analysis and alternative solutions," in *Advances in Cryptology-CRYPTO 2011*, pp. 578–595, Springer, Berlin, Germany, 2011.

[31] R. A. Popa, F. H. Li, and N. Zeldovich, "An ideal-security protocol for order-preserving encoding," in *Proceedings of the*

2013 IEEE Symposium on Security and Privacy, San Francisco, CA, USA, May 2013.

[32] http://https://lbs.amap.com/.

[33] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services. ACM*, pp. 31–42, San Francisco, CA, USA, May 2003.

[34] X. Gong, X. Chen, K. Xing, D.-H. Shin, M. Zhang, and J. Zhang, "From social group utility maximization to personalized location privacy in mobile networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 3, pp. 1703–1716, 2017.

[35] S. Zhang, G. Wang, M. Z. A. Bhuiyan, and Q. Liu, "A dual privacy preserving scheme in continuous location-based services," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 4191–4200, 2018.

[36] H. Wang, H. Huang, Y. Qin, Y. Wang, and M. Wu, "Efficient location privacy-preserving k-anonymity method based on the credible chain," *ISPRS International Journal of Geo-Information*, vol. 6, no. 6, p. 119, 2017.

[37] S. Zhang, X. Li, Z. Tan, T. Peng, and G. Wang, "A caching and spatial k-anonymity driven privacy enhancement scheme in continuous location based services," *Future Generation Computer Systems*, vol. 94, p. 4050, 2019.

[38] T. Peng, Q. Liu, and G. Wang, "Enhanced location privacy preserving scheme in location-based services," *IEEE Systems Journal*, vol. 11, no. 1, pp. 219–230, 2017.

[39] Z. Montazeri, A. Houmansadr, and H. Pishro-Nik, "Achieving perfect location privacy in wireless devices using anonymization," *IEEE Transactions on Information Forensics & Security*, vol. 12, no. 11, Article ID 26832698, 2017.

[40] L. Huang, Y. Ma, Y. Liu, and A. K. Sangaiah, "Multi-modal bayesian embedding for point-of-interest recommendation on location-based cyber-physical-social networks," *Future Generation Computer Systems*, vol. 108, pp. 1–10, 2020.

[41] A. K. Sangaiah, D. V. Medhane, T. Han, M. S. Hossain, and G. Muhammad, "Enforcing position-based confidentiality with machine learning paradigm through mobile edge computing in real-time industrial informatics," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4189–4196, 2019.

[42] R. Shokri, G. Theodorakopoulos, and C. Troncoso, "Privacy games along location traces," *ACM Transactions on Privacy and Security*, vol. 19, no. 4, pp. 1–31, 2017.

[43] G. Sun, D. Liao, H. Li, H. Yu, and V. Chang, "L2P2: a location-label based approach for privacy preserving in LBS," *Future Generation Computer Systems*, vol. 74, pp. 375–384, 2017.

[44] T. Peng, Q. Liu, D. Meng, and G. Wang, "Collaborative trajectory privacy preserving scheme in location-based services," *Information Sciences*, vol. 387, pp. 165–179, 2017.

[45] H. To, G. Ghinita, L. Fan, and C. Shahabi, "Differentially private location protection for worker datasets in spatial crowdsourcing," *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 934–949, 2016.

[46] T. Zhu, G. Li, W. Zhou, and P. S. Yu, "Differentially private data publishing and analysis: a survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 8, pp. 1619–1638, 2017.

[47] T. Wang, Y. Mei, W. Jia, X. Zheng, G. Wang, and M. Xie, "Edge-based differential privacy computing for sensor-cloud systems," *Journal of Parallel and Distributed Computing*, vol. 136, pp. 75–85, 2020.

[48] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan, "Practical approximate k nearest neighbor queries with location and query privacy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 6, pp. 1546–1559, 2016.

[49] I. Memon, Q. A. Arain, M. H. Memon, F. A. Mangi, and R. Akhtar, "Search me if you can: multiple mix zones with location privacy protection for mapping services," *International Journal of Communication Systems*, vol. 30, no. 16, pp. 1–23, 2017.

[50] S. Mascetti, C. Bettini, D. Freni, X. S. Wang, and S. Jajodia, "Privacy-aware proximity based services," in *Proceedings of the 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pp. 1–10, Taipei, Taiwan, May 2009.

[51] T. Jing, P. Lin, Y. Lu, C. Hu, and Y. Huo, "Fpodg: a flexible and private proximity testing based on "one degree" grid," *International Journal of Sensor Networks*, vol. 20, no. 3, pp. 199–207, 2016.

[52] X.-Y. Li and T. Jung, "Search me if you can: privacy-preserving location query service," in *2013 Proceedings of the IEEE INFOCOM*, pp. 2760–2768, Turin, Italy, April 2013.