Journal of Computer Engineering
ISSN 10003428

Original Research Article
Vol 8, Issue 11, 2019 | ISSN: 1000-3428

"Secure Geolocation Sharing in Mobile Online Communities: A Client-Server Framework"

"Zachary J. Miller, Sofia Rodriguez, Ethan M. Brooks, Maya K. Patel, Julian A. Sanchez"

"Zachary J. Miller and Sofia Rodriguez, Department of Computer Science, University of California, Berkeley;

***Abstract*—**Location sharing is a fundamental service in mobile
Online Social Networks (mOSNs), which raises significant privacy concerns in recent years. Now, most location-based service applications adopt client/server architecture. In this paper, a location sharing system, named CSLocShare, is presented to provide flexible privacy-preserving location sharing with client/server architecture in mOSNs. CSLocShare enables location sharing between both trusted social friends and untrusted strangers without the third-party server. In CSLocShare, Location-Storing Social Network Server (LSSNS) provides location-based services but do not know the users' real locations. The thorough analysis indicates that the users' location privacy is protected. Meanwhile, the storage and the communication cost are saved. CSLocShare is more suitable and effective in reality.

I. Introduction

W

ITH the development of mobile Internet, the social network enters the era of mOSNs. In a traditional social network that supports Location-Based Services (LBSs), such as Foursquare and Gowalla, users share their locations by means of "check-in", which helps the online website/application to record the time and the location of each user. The inefficient and inconvenient location sharing situation is not changed until the appearance of mobile devices, which can obtain the owner's location through Global Positioning System (GPS) or cellular geolocation anywhere and anytime. Since mobile devices can be online all the time and they can be everywhere, real-time interaction of users is realized in mOSNs. More excitingly, users can conveniently use various LBSs provided by mobile devices, such as recommendation of good friends or searching Points of Interests (POIs) including hotels, hospitals, and restaurants. However, behind the convenience brought by mOSNs, there comes an indispensable security risk of privacy. In the era of data, location data does not only mean the place of an individual, but also presents his/her home, interests, physical conditions, and so on. Therefore, location records should be regarded as unusually sensitive information and they deserve a high degree of attention.

In order to flexibly share privacy-preserving location in mOSNs, Wei et al. [1] introduced MobiShare in 2012. Since MobiShare enables location sharing between both trusted social relations and untrusted strangers, it is adaptable to support a
variety of location-based applications. Based on MobiShare, N-MobiShare [2], [3], MobiShare+ [4], and BMobiShare [5] have been successively put forward. N-MobiShare [2], [3] uses Social Network Server (SNS) instead of Cellular Tower (CT) to forward users' location updates and requests to Location Based Server (LBS) without leaking anything about the location information. Shortly afterwards, MobiShare+[4] employs dummy queries and the private set intersection protocol to prevent SNS and LBS from learning individual information from each other. In order to improve transmission efficiency, BMobiShare [5] replaces the private set intersection protocol in MobiShare+ with Bloom Filter. In fact, most of the LBS applications have client/server architecture [6]-[8]. Therefore, all these systems, which depend on the third-part LBS, are not suitable in the real world. Further, the third-part LBS generates additional communication overhead and economic costs. Moreover, if SNS and LBS work together with each other, they can easily get both users' individual profile and location information.

In order to overcome the above drawbacks, we present a new privacy-preserving location-sharing system for client/server based applications in mOSNs, named CSLocShare. Different from MobiShare, N-MobiShare, MobiShare+, and BMobiShare, in our system, SNS and LBS have been amalgamated into one single server, i.e., LSSNS. The efficiency in communication is improved, while the protection effect on the users' privacy is the same as the previous systems. Users' basic social information and location data are stored in LSSNS, and every user has one real location and k-1 dummy locations. So, LSSNS cannot know the users' real location and only CT is able to identify the real location. Meanwhile, CT no longer needs to store any users' related records and just only does some simply computation.

There are three main contributions in our work as follows:

1. We observe that, in real life, most of the mobile social applications have client/server architecture without the third-party server such as LBS. Therefore, we design a new location-sharing system named CSLocShare, which does not depend on the third-part

How to cite: "Zachary J. Miller, Sofia Rodriguez, Ethan M. Brooks, Maya K. Patel, Julian A. Sanchez" (2019). "Secure Geolocation Sharing in Mobile Online Communities: A Client-Server Framework". Journal of Computer Engineering, 8(11), 117–128. DOI: https://doi.org/10.5281/zenodo.19341122

server and can achieve the same protection goal on the users' privacy.

2. In our system, CT does not keep users' relevant information, thus the storage load has considerably decreased, while, in MobiShare, MobiShare+, and BMobiShare, CT has to store users' relevant information to ensure that SNS and LBS are able to cooperate with each other to complete users' location updates and requests.
3. The nearby friends' and strangers' locations are obtained in one entity, LSSNS, which does not require the cooperation between SNS and LBS. Thus, the encryption and transmission of the interactive data between SNS and LBS are eliminated.

This paper is organized as follows. In Section II, we state the system model and threat model for our mechanism. In Section III, we detail our improved privacy-preserving mechanism and system design. In Section IV, we present a comprehensive security analysis. In Section V, we briefly describe related work. Finally, we draw the conclusion in Section VI.

II. Problem Statement

In this section, we mainly introduce the privacy issues of location-sharing in mOSNs. Meanwhile, we simply analyze several main attack models in mOSNs.

A. System Model

In Fig. 1, we show the architecture of our location-sharing system in mOSNs, which consists of three entities.

(1) Mobile user U is an entity that intends to share U's real-time location and to request locations of nearby friends and strangers.

(2) LSSNS is an entity that not only stores U's social profile and location information, but also provides LBSs according to U's requests with nearby persons' locations.

(3) CT is an entity that helps U to communicate with LSSNS and executes some simple computation.

There are manifold challenges of location-sharing in mOSN. The first challenge is that how to update U's location in a privacy-preserving way when U arrives at a new place and the location has changed. When U wants to know the nearby friends' and strangers' locations, how to complete U's location request without leaking both U and the qualified users' individual privacy is the second challenge. We formalize these issues as follows.

We formalize the whole social network as a Graph $G = (V, E)$, where $V \in G$ is a set of identity vertices and $E \in G$ is a set of edges. Suppose that $V = \{ID_1, ID_2, \dots, ID_n\}$ is the identity set of all the users in the social network supporting location sharing. If there is a linked edge between ID_1 and ID_2, ID_1 and ID_2 own a trusted social relationship, i.e., friendship. If we do not think the privacy concerns, LSSNS maintains a location database, in which the location records are stored in the form of $\{ID, (x_{ID}, y_{ID}), df_{ID}, ds_{ID}\}_{ID \in V}$. In the record $\{ID, (x_{ID}, y_{ID}), df_{ID}, ds_{ID}\}_{ID \in V}$, ID is the user's identity, (x_{ID}, y_{ID}) is his/her current location, df_{ID} is the distance threshold within which the user is willing to share his/her location to the nearby friends, and ds_{ID} is the distance threshold within which the user agrees to share location with close-by strangers.

In a social network, the users' authorized location-based operations are real-location update and locations query of nearby friends/strangers. The first security issue is how to generate/update an appropriate record to LSSNS's location database so as to satisfy desired security goals. The second security issue is how to make an operation model according to friends'/strangers' location queries without violating privacy of both sides.

The locations query can be formalized in two sides:

(1) For friends' locations query, a user U with identity ID_U can send a query in the form of $(ID_U, \text{'f'}, qf_U)$ to acquire all his/her nearby friends' locations $\{(ID_i, (x_i, y_i))\}$ satisfying the constraint $(ID_U, ID_i) \in G.E$ and $dist((x_U, y_U), (x_i, y_i)) \leq min(qf_U, df_i)$, where 'f' presents the query of finding friends, (x_i, y_i) is the user ID_i's current location, $dist(.,.)$ is a function to compute Euclidean distance, $min(.,.)$ is a function to return minimum value in its input, qf_U is the distance threshold within which ID_U wants to find his/her friends, and df_i is the distance threshold within which ID_i allow his/her friend to find himself.

(2) For strangers' locations query, a user U with identity ID_U can submit a request in the form of $(ID_U, \text{'s'}, qs_U)$ to get all the nearby strangers' locations $\{(ID_i, (x_i, y_i))\}$ fitting in with $dist((x_U, y_U), (x_i, y_i)) \leq min(qs_U, ds_i)$, where 's' presents the query of finding strangers, qs_U is the distance threshold within which ID_U wants to find the nearby strangers, and ds_i is the distance threshold

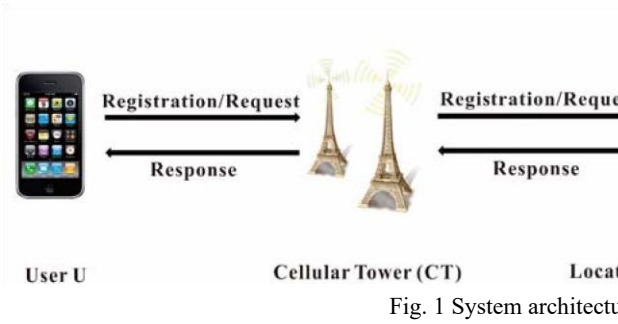

Fig. 1 System architectu

within which the user ID_i allows the others to find himself in strangers' locations query.

B. Adversary Model

The system consists of three entities including mobile users, CT and LSSNS. In our trusted model, CT is assumed as a trusted entity, which can obtain the users' complete information without rising privacy concerns. Here, we focus on security risk with respect to LSSNS and mobile users. There are three main attack models as follows.

(1) LSSNS. We consider that LSSNS is "honest-but-curious". This means that LSSNS is supposed to honestly follow our presented protocol in general but attempt to acquire the users' information as much as possible.

(2) Dishonest mobile users. The dishonest mobile users try to use authorized operation to acquire the information outside the scope of their access privileges.

(3) Collusion between LSSNS and Dishonest mobile users. LSSNS may collude with these malicious users. For example, a social network company's employee may register for the location sharing service and collude with the server to extract the users' complete social-location topology structure.

Different from the previous systems, our system merges SNS and LBS into one server, i.e. LSSNS so that there is no threat that SNS leaks social relationships to LBS or LBS leaks location information to SNS. In fact, LSSNS has the users' social relationship network. So, one of most important goals is to prevent LSSNS from obtaining the complete location topology structure. In fact, in our system, LSSNS cannot identify the target user's real location even though the target user's real location is stored in it.

III. System Design

In this section, we detail our practical system, which separates the issue of privacy-preserving location sharing into two cases, sharing locations between trusted friends and untrusted strangers. A summary of the notations used in this section is given in Table I.

TABLE I
SUMMARY OF NOTATIONS

Symbol	Description
ID_U	User U's social network identifier, regarded as his/her identity
(x_U, y_U)	User U's real location
df_U	User U's friend-case distance threshold
ds_U	User U's stranger-case distance threshold
qf_U	User U's Distance threshold in friends' locations query
qs_U	User U's Distance threshold in strangers' locations query
$Sess_U$	A symmetric key shared by user and his/her friends
G	A social network graph stored at LSSNS
Tag_U	An encrypted string that indicates the index of real location
Key_{LS}	Location-Storing Social Network Server's secret key, shared with cellular towers
Key_{CT}	Cellular tower's secret key
$Decrypt_{Key}(.)$	A decryption function using secret key Key
$dist(.,.)$	A function to compute Euclidean distance
$min(.,.)$	A function to return minimum value in its input

A. System Overview

Before we detail our proposed system, we first give an overview of location-sharing mechanism which mainly includes four processes as follows.

(1) Registration: Before using the location-sharing service in the social network, the user needs to register on LSSNS to get an account ID as his/her identifier. Then, the user sends his/her location-sharing preferences to LSSNS. And LSSNS stores the user's location-sharing preferences in the database and provides individual LBSs according to the user's preferences. In addition, the user also has to register on the local CT to obtain the permission of authorized operation.

(2) Location Updates: When the user reaches a new place, he/she must update his/her location in LSSNS's database to ensure that LSSNS knows the user's real-time locations.

(3) Friends' Locations Query: When a user wants to know the nearby friends' current locations, he/she can submit a friends' locations query to LSSNS. After receiving the user's query, LSSNS selects the user's friends in the database, who satisfy the distance restriction of the user, and returns these friends' present locations.

(4) Strangers' Locations Query: When a user intends to know the nearby strangers' current locations, he/she can send a strangers' locations request to LSSNS. After receiving a user's query, LSSNS selects the users in the database satisfying the distance restriction of the user and transmits these qualified users' locations to the requester.

B. Registration

There is no doubt that LSSNS distributes U a unique identifier in the social network and builds a friend-relationship network for U. Before mobile user U uses the location-sharing service, he/she needs to register the service in CT and LSSNS and shares location-sharing preferences with LSSNS. The main registration process is described as follows (Fig. 2 shows the message transmission in this stage).

Step 1. U sends a record in the form of {ID_U, 'reg', df_U, ds_U} to CT, where 'reg' presents the registration query, df_U is the distance threshold within which ID_U allow his friends to find himself, ds_U is the distance threshold within which the user ID_U allows the others to find himself in strangers' locations query.

Step 2. After receiving U's message, CT forwards {ID_U, 'reg', df_U, ds_U} to LSSNS.

Step 3. When LSSNS receives the registration query, LSSNS keeps a record as {ID_U, df_U, ds_U} in its local subscriber database. Then, LSSNS sends a reply (ID_U, 'ok') to CT.

Step 4. Finally, CT forwards LSSNS's message (ID_U, 'ok') to U in order to indicate registration success. Consequently, secure communication link is established between U and CT.

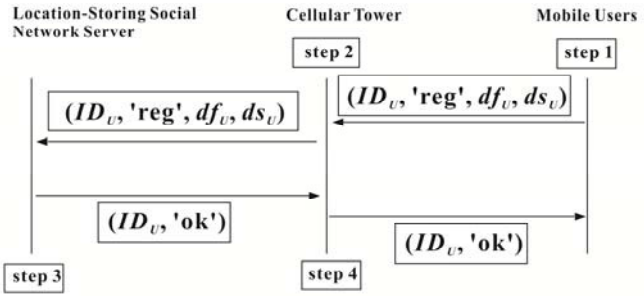

Fig. 2 Registration

Step 2. After receiving U's message, CT forwards {ID_U, 'reg', df_U, ds_U, Tag_U}, where the real location (x_U, y_U, str_U) is randomly put at the nth place,($1 \le n \le k$), and Tag_U is an encrypted string that indicates the index of the real location, encrypted by CT's secret key Key_{CT}.

Step 3. When LSSNS receives the update record, it stores the record in the database and sends a success message (ID_U, 'ok') to CT.

Step 4. After receiving the message from LSSNS, CT directly forwards the message to U.

In LSSNS, the database consists of a number of tables, and each table represents a geographic region. The updates of locations within a region are kept in the corresponding table, where the user's *ID* is the primary key. The purpose of storage measure is aimed at improving search efficiency and reducing computation overhead. For example, given one location, to find the strangers within a range, instead of searching all the stored location records, LSSNS only needs to check the tables of the regions that overlap the queried circular area. It is remarkable that the entries in the database expire after a certain period of time.

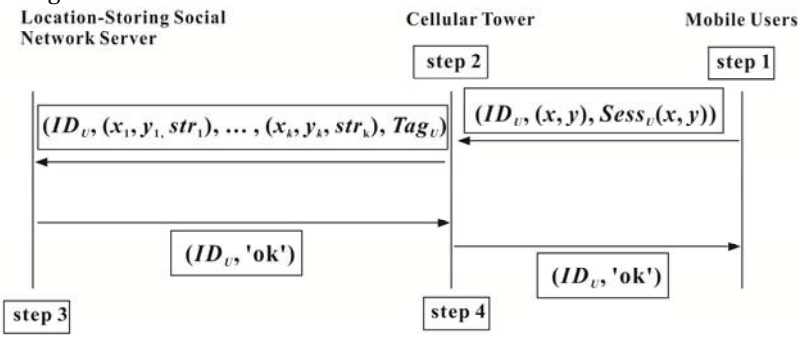

Fig. 3 Location Updates

C. Location Updates

Different from MobiShare, we no more employ the dummy techniques in LBS or store the mapping entries in SNS. We merge SNS and LBS into one server, i.e., LSSNS and use dummy techniques to generate k-1 dummy locations, which are stored together with user U's real location in LSSNS. The detailed steps are as follows (Fig. 3 shows the message transmission in this stage).

Step 1. When U intends to update his/her location, he/she uploads a record in the form of (ID_U, (x, y), $Sess_U(x, y)$) to CT, where (x, y) is U's current location, and $Sess_U(x, y)$ is the location encrypted with U's session key, which is shared with all his/her trusted social friends.

Step 2. CT randomly generates k-1 dummy locations (x_i' , y_i'), i=1,...,k-1, and k-1 random strings str_i, i = 1,...,k-1 to imitate the encrypted locations. To anonymize the location update from U, one real location and k-1 dummy locations are sent to LSSNS in the form of {ID_U, (x_1, y_1, str_1), ... , (x_k, y_k, str_k), Tag_U}, where the real location (x_U, y_U, str_U) is randomly put at the nth place,($1 \le n \le k$), and Tag_U is an encrypted string that indicates the index of the real location, encrypted by CT's secret key Key_{CT}.

Step 3. When LSSNS receives the update record, it stores the record in the database and sends a success message (ID_U, 'ok') to CT.

Step 4. After receiving the message from LSSNS, CT directly forwards the message to U.

In LSSNS, the database consists of a number of tables, and each table represents a geographic region. The updates of locations within a region are kept in the corresponding table, where the user's *ID* is the primary key. The purpose of storage measure is aimed at improving search efficiency and reducing computation overhead. For example, given one location, to find the strangers within a range, instead of searching all the stored location records, LSSNS only needs to check the tables of the regions that overlap the queried circular area. It is remarkable that the entries in the database expire after a certain period of time.

D. Querying Friends' Locations

When user U wants to query the nearby friends' locations, there are four steps as follows (Fig. 4 shows the message transmission in this stage).

Step 1. When U intends to query his/her friends' locations within a certain range, he/she can submit a query (ID_U, 'f', qf_U) to CT, where qf_U is U's specified distance, within which he/she wants to find his/her friends.

Step 2. After receiving U's query, CT appends a sequence number *seq*, which are encrypted by LSSNS's secret key Key_{LS}, and forwards the query (ID_U, 'f', qf_U, Key_{LS}(*seq*)) to LSSNS, where *seq* is used to resist the replay attack and the tampering attack and Key_{LS} is LSSNS's secret key, shared with the CT.

Step 3. Upon receiving the query (ID_U, 'f', qf_U, Key_{LS}(*seq*)), LSSNS finds U's friends set *FS* consisting of all the friend identifiers *ID*', with (*ID*, *ID*') ∈ *G.E*. For each entry (*ID*, *df*, *ds*, (x_1, y_1, str_1), (x_2, y_2, str_2),...,(x_k, y_k, str_k),

Tag) stored in the database, LSSNS checks whether $dist((x_i, y_i), (x_{Ut}, y_{Ut})) \leq min(qf_U, df_s)_{s \in FS}$, i=1,...,k, t=1,...,k, where (x_{Ut}, y_{Ut}) are the corresponding k locations of U. For all $F \in FS$, if one or more (x_{Fi}, y_{Fi}), i=1,...,k, satisfy the distance requirement, the corresponding record in the form of (ID_F, str_{Fi}, Tag_F) inserts into the result set. For example, if $A \in FS$ and (x_{A2}, y_{A2}), (x_{A3}, y_{A3}) and (x_{Ak}, y_{Ak}) meet the above restrictions, $(ID_A, (2, str_2), (3, str_3), (k, str_k), Tag_A)$ is recorded. And the result set is divided into k subsets Fq_i, i = 1,...,k, according to the corresponding center point

(x_{Ut}, y_{Ut}), t = 1,...,k. Finally, LSSNS replies $Key_{LS}(\{Fq_i\}_{i=1,...,k}, Tag_U, seq)$ to CT.

Step 4. Upon receiving $Key_{LS}(\{Fq_i\}_{i=1,...,k}, Tag_U, seq)$, CT decrypts the reply with Key_{LS} and checks whether *seq* corresponds to the sequence number that it previously sent, and finds the real center point (U's real location) with the real location index $r = Decrypt_{KeyCT}(Tag_U)$, where $Decrypt_{KeyCT}(.)$ is a decryption function using the secret key Key_{CT}. According to the real center point, CT chooses Fq_r where r is the real location index and discards the reminding set Fq_i, $i \neq r$. For each item in the set Fq_r, say $(ID_A, (2, str_2), (3, str_3), (k, str_k), Tag_A)$, if $Decrypt_{KeyCT}(Tag_A)$ indicates that the real location index is 3, then str_3 is $Sess_U(x_A, y_A)$ and $(ID_A, Sess_U(x_A, y_A))$ is added to the final result Ans_F. If the real location index is not 2, 3 or k, CT continues to handle the next record in Fq_r. Finally Ans_F is a set in the form of $\{ID_i, Sess_U(x_i, y_i)\}_{i=1,...,k'}$, $k' \leq k$, and CT sends (ID_U, Ans_F) to U.

When U receives (ID_U, Ans_F) and decrypts the encrypted friends' locations with U's session key, then U knows both the identities and the locations of the nearby friends who are willing to share their location information.

E. Querying Strangers' Locations

The strangers' locations query performs in a similar way with the friends' locations query, but does not require LSSNS to find friends and the scope of the candidates expands to all the nearby users. There are four steps in this stage as follows (Fig. 5 shows the message transmission in this stage).

Step 1. User U submits a strangers' locations query $(ID_U$, 's', $qs_U)$ to CT, where qs_U is U's specified distance, within which he/she wants to find the nearby strangers.

Step 2. After receiving U's query, CT appends a sequence number, which are encrypted by LSSNS's secret key, and forwards the query $(ID_U$, 's', qs_U, $Key_{LS}(seq))$ to LSSNS.

Step 3. Upon receiving the query $(ID_U$, 's', qs_U, $Key_{LS}(seq))$, for each entry $(ID, df, ds, (x_1, y_1, str_1), (x_2, y_2, str_2),...,(x_k, y_k, str_k), Tag)$ stored in the database, check whether $dist((x_i, y_i), (x_{Ut}, y_{Ut})) \leq min(qs_U, ds_i)$, i=1,...,k, t=1,...,k, where (x_{Ut}, y_{Ut}) are the corresponding k locations of U. For each user in the database, if one or more (x_i, y_i), i = 1,...,k, satisfy the distance requirement, the corresponding record in the form of $(ID, (x_i, y_i), Tag)$ inserts into the result set. For example, for user A, (x_{A2}, y_{A2}), (x_{A3}, y_{A3}) and (x_{Ak}, y_{Ak}) meet the above restrictions, $(ID_A, 2, (x_{A2}, y_{A2}), 3, (x_{A3}, y_{A3}), k, (x_{Ak}, y_{Ak}), Tag_A)$ are recorded. And the result set is divided into k subsets Sq_i, i = 1,...,k, according to the corresponding center point (x_{Ut}, y_{Ut}), t=1,...,k. Finally, LSSNS replies $Key_{LS}(\{Sq_i\}$ i = 1,...,k, Tag_U, *seq*) to CT.

Step 4. Upon receiving $Key_{LS}(\{Sq_i\}_{i=1,...,k}, Tag_U, seq)$, CT decrypts the message with Key_{LS} and checks whether *seq* corresponds to the sequence number it previously sent and finds the real center point (U's real location) by decrypting Tag_U. According to the real center point, CT chooses the real subset Sq_r and discards the reminding set Sq_i, $i \neq r$. For each item in the set Sq_r, say $(ID_A, 2, (x_{A2}, y_{A2}), 3, (x_{A3}, y_{A3}), k, (x_{Ak}, y_{Ak}), Tag_A)$, if $Decrypt_{KeyCT}(Tag_A)$ indicates that the real location index is 3, $(ID_A, (x_3, y_3))$ is added to the final result Ans_S and if the real location index is not 2, 3 or k, CT continues to handle the next record in Sq_r. Finally, Ans_S is a set in the form of $\{ID_i, (x_i, y_i)\}_{i=1,...,k'}$, $k' \leq k$, and CT sends (ID_U, Ans_S) to U.

When U receives (ID_U, Ans_S), U knows both the identities and the locations of the nearby strangers who are willing to share their location information.

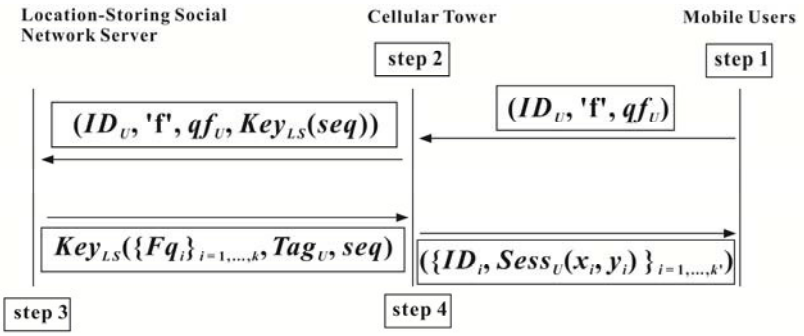

Fig. 4 Querying Friends' Locations

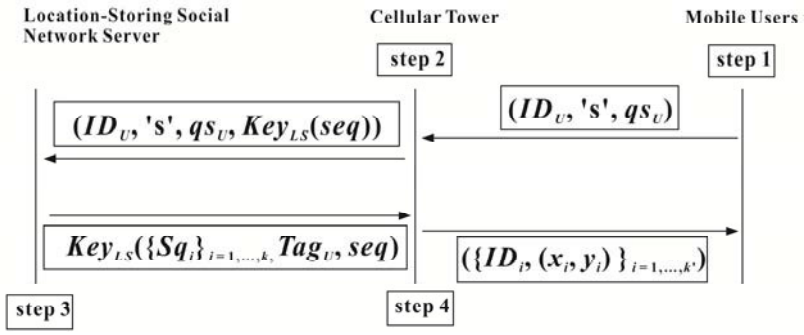

Fig. 5 Querying Strangers' Locations

IV. Security Analysis

In our system, we assume that CT is a trusted entity and attempts to find out as much privacy information as possible. LSSNS is 'honest-but-curious'. In the other words, this work does not concern social-location privacy rising by CT and users and intend to obtain the location of a target user. Through LSSNS is supposed to follow the agreement mechanism, but it Moreover, LSSNS may collude with the arbitrary malicious

analyzing the following two main threats, we can prove that our system is secure.

Threat 1: LSSNS intends to obtain the target user's location. Although all the users' real locations are stored in plain at LSSNS, K-anonymity technique is used so that LSSNS has at most the probability of $1/k$ to guess the user's real location. Though LSSNS deals with location update and query, LSSNS cannot distinguish the users' real locations because of K-anonymity technique. If LSSNS wants to get the user's real location, it must decrypt the encrypted tag, which is encrypted by CT. That is to say, as long as the encryption scheme using by CT is safe enough, our system is secure.

Threat 2: Malicious users, even colliding with LSSNS, want to obtain the target user's location. User B can fake as user A to query A's friends location, and CT sends back A's result set. B can still not obtain A's friends' locations because A's friends' locations in the result set are encrypted by the session key, which is only sharing with A's friends.

Additionally, each transmission message appends a sequence number, which is effectively withstanding the replay attack and the tampering attack.

V. Related Work

With the rise of OSNs, location sharing becomes an increasingly significant LBS, especially in mOSN. Simultaneously, the privacy issue caused by the location-sharing has developed into a devil of a tricky problem. In order to address the location privacy issue risen by location-sharing, many researches about information privacy [9] and location privacy protection [10], [11] have been done. There are a large number of researches focusing on preventing a location server from learning users' location when the users access the LBSs with their location information, e.g. the K-anonymity [12]-[14], the mix zones [15], [16], the pseudonym methods [17], [18], the m-unobservability [19], [20], and the location anonymity [21]-[23]. Except the methods above, in order to defend against various inference attacks, [24] present a systematic solution basing on differential privacy to preserve location privacy. In addition, [25]-[27] proposed privacy context obfuscation to obscure location information based on parameters, such as data requester, time of day, and so on.

In recent years, with the cell phones and tablets being exploding, most devices are considerably smart and capable of determining their locations through GPS or cellular geolocation. As a result, with the rapid fusion of OSNs with mobile computing, a new paradigm called mOSNs has emerged. A lot of works in mOSNs have been carried out. In 2007, SmokeScreen [28] provided a flexible and power-efficient mechanism to allow safely sharing location between both trusted social friends and untrusted strangers. Later, considering flexible privacy-preserving location sharing in mOSNs, [1] introduced MobiShare, which is adaptable to support a variety of location-based applications, in that it enables location-sharing between both trusted social relations and untrusted strangers. In MobiShare, users' individual profiles and location information are separately stored on SNS and LBS to secure the users' location privacy. After MobiShare [1], many improved systems were subsequently proposed, such as [2]-[5]. N-MobiShare [2], [3] assumes that CT should not be treated as a core component of the system, and instead of CT, SNS is used to forward users' location update requests to the Location-Based Server without knowing anything about the location information. Meanwhile, N-MobiShare employs Identity- Based Broadcast Encryption (IBBE) to realize sharing key off-line to all users' friends. Shortly afterwards, Li et al. [4] observed that users' real fake identities would be potentially leaked to location service provider in MobiShare and then they proposed a security improved mechanism, i.e. MobiShare+ which applies dummy queries and the private set intersection protocol to prevent SNS and LBS from learning individual information from each other. In order to improve transmission efficiency, MobiShare [5] uses Bloom Filter, which can filter invalid information with masking the sensitive data, to replace private set intersection protocol in MobiShare+. In 2015, aiming at achieving enhanced privacy against the insider attack launched by the service providers in mOSNs, Li et al. [29] introduced a new architecture with multiple location servers and proposed a secure solution supporting sharing location between friends and strangers in location-based applications.

All the above mechanisms have to depend on the third-party location server, which greatly increases the risks of privacy leakage and the economic burden. Different from the previous systems [1-5], considering the popular client/server structure, our system amalgamates SNS and LBS into one single server, i.e. LSSNS, which has the same protection effect on the users' privacy and the less resource consumption.

VI. Conclusion

In this paper, taking into account the current popular client/server architecture of location-based applications in mOSNs, we present CSLocShare, a system that provides flexible privacy-preserving location sharing with client/server architecture in mOSNs. CSLocShare does not depend on the third-party server. It achieves a higher efficiency in communication and has the same protection effect on the users' privacy as the previous systems. CSLocShare can be applied more widely in mOSNs.

REFERENCES

[1] Wei W, Xu F, Li Q. Mobishare: Flexible privacy-preserving location sharing in mobile online social networks (C)//INFOCOM, 2012 Proceedings IEEE. IEEE, 2012: 2616-2620.

[2] Liu Z, Luo D, Li J, et al. N-Mobishare: new privacy-preserving location-sharing system for mobile online social networks (J). International Journal of Computer Mathematics, 2016, 93(2): 384-400.

[3] Liu Z, Li J, Chen X, et al. New privacy-preserving location sharing system for mobile online social networks (C)//P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on. IEEE, 2013: 214-218.

[4] Li J, Li J, Chen X, et al. MobiShare+: Security improved system for location sharing in mobile online social networks (J). Journal of Internet Services and Information Security, 2014, 4(1): 25-36.

[5] Shen N, Yang J, Yuan K, et al. An efficient and privacy-preserving location sharing mechanism (J). Computer Standards & Interfaces, 2016, 44: 102-109.

[6] Pontikakos C, Glezakos T, Tsiligiridis T. Location-based services: architecture overview (C)//Proceedings of the International Congress on Information Technology in Agriculture, Food and Environment (ITAFE'05). 2005.

[7] Chow C Y, Mokbel M F. Privacy in location-based services: a system architecture perspective (J). Sigspatial Special, 2009, 1(2): 23-27.

[8] Jensen C S, Lu H, Yiu M L. Location privacy techniques in client-server architectures (M)//Privacy in location-based applications. Springer Berlin Heidelberg, 2009: 31-58.

[9] Qian Z, Chen C, You I, et al. ACSP: A novel security protocol against counting attack for UHF RFID systems (J). Computers & Mathematics with Applications, 2012, 63(2): 492-500.

[10] Ju X, Shin K G. Location Privacy Protection for Smartphone Users Using Quadtree Entropy Maps (J). Journal of Information Privacy and Security, 2015, 11(2): 62-79.

[11] Rao U P, Girme H. A novel framework for privacy preserving in location based services (C)//2015 Fifth International Conference on Advanced Computing & Communication Technologies. IEEE, 2015: 272-277.

[12] Sweeney L. k-anonymity: A model for protecting privacy (J). International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2002, 10(05): 557-570.

[13] Gruteser M, Grunwald D. Anonymous usage of location-based services through spatial and temporal cloaking (C)//Proceedings of the 1st international conference on Mobile systems, applications and services. ACM, 2003: 31-42.

[14] Meyerowitz J, Roy Choudhury R. Hiding stars with fireworks: location privacy through camouflage (C)//Proceedings of the 15th annual international conference on Mobile computing and networking. ACM, 2009: 345-356.

[15] Beresford A R, Stajano F. Location privacy in pervasive computing (J). IEEE Pervasive computing, 2003, 2(1): 46-55.

[16] Beresford A R, Stajano F. Mix Zones: User Privacy in Location-aware Services (C)//PerCom Workshops. 2004: 127-131.

[17] Ouyang Y, Le Z, Xu Y, et al. Providing Anonymity in Wireless Sensor Networks (C)//ICPS. 2007: 145-148.

[18] Rahman M, Mambo M, Inomata A, et al. An anonymous on-demand position-based routing in mobile ad hoc networks (C)//International Symposium on Applications and the Internet (SAINT'06). IEEE, 2006: 7 pp.-306.

[19] Chen Z, Hu X, Ju X, et al. LISA: Location information scrambler for privacy protection on smartphones (C)//Communications and Network Security (CNS), 2013 IEEE Conference on. IEEE, 2013: 296-304.

[20] Chen Z. Energy-efficient information collection and dissemination in wireless sensor networks (D). The University of Michigan, 2009.

[21] J. Li and K. Kim, Hidden attribute based signature without revocation, Inform. Sci. 180 (2010), pp. 1681–1689.

[22] J. Li and Y. Wang, Universal designated verifier ring signature (proof) without random oracles. Emerging Directions in Embedded and Ubiquitous Computing, Springer Berlin Heidelberg, 2006: 332–341.

[23] Rass S, Wigoutschnigg R, Schartner P. Doubly-Anonymous Crowds: Using Secret-Sharing to achieve Sender-and Receiver-Anonymity (J). JoWUA, 2011, 2(4): 27-41.

[24] Xiao Y, Xiong L. Protecting locations with differential privacy under temporal correlations (C)//Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM, 2015: 1298-1309.

[25] Li J, Kim K, Zhang F, et al. Aggregate proxy signature and verifiably encrypted proxy signature (C)//International Conference on Provable Security. Springer Berlin Heidelberg, 2007: 208-217.

[26] Li J, Zhang F, Wang Y. A new hierarchical ID-based cryptosystem and CCA-secure PKE (C) //International Conference on Embedded and Ubiquitous Computing. Springer Berlin Heidelberg, 2006: 362-371.

[27] Rahman F, Hoque M E, Kawsar F A, et al. Preserve your privacy with pco: A privacy sensitive architecture for context obfuscation for pervasive e-community based applications (C) //Social Computing (SocialCom), 2010 IEEE Second International Conference on. IEEE, 2010: 41-48.

[28] Cox L P, Dalton A, Marupadi V. Smokescreen: flexible privacy controls for presence-sharing (C) //Proceedings of the 5th international conference on Mobile systems, applications and services. ACM, 2007: 233-245.

[29] Li J, Yan H, Liu Z, et al. Location-Sharing Systems with Enhanced Privacy in Mobile Online Social Networks (J).

[30] Zhou B, Pei J. Preserving privacy in social networks against neighborhood attacks (C)//2008 IEEE 24th International Conference on Data Engineering. IEEE, 2008: 506-515.

[31] Narayanan A, Shmatikov V. De-anonymizing social networks (C)//2009 30th IEEE symposium on security and privacy. IEEE, 2009: 173-187.

[32] Wang G, Liu Q, Li F, et al. Outsourcing privacy-preserving social networks to a cloud (C)//INFOCOM, 2013 Proceedings IEEE. IEEE, 2013: 2886-2894.