Hindawi
Security and Communication Networks
Volume 2018, Article ID 4724815, 11 pages
https://doi.org/10.1155/2018/4724815

Research Article

An Efficient and Privacy-Preserving Multiuser Cloud-Based LBS Query Scheme

Lu Ou,[1] **Hui Yin**,[1,2] **Zheng Qin**,[1] **Sheng Xiao**,[1] **Guangyi Yang**,[3,4] **and Yupeng Hu**[1]

[1]*College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan 410082, China*
[2]*College of Computer Engineering and Applied Mathematics, Changsha University, Changsha, Hunan 410022, China*
[3]*School of Information Science and Engineering, Central South University, Changsha, Hunan 410083, China*
[4]*Hunan Institute of Metrology and Test, Changsha, Hunan 410014, China*

Correspondence should be addressed to Hui Yin; yhui@ccsu.edu.cn and Zheng Qin; zqin@hnu.edu.cn

Received 24 September 2017; Accepted 31 December 2017; Published 8 March 2018

Academic Editor: Zhenyu Li

Location-based services (LBSs) are increasingly popular in today's society. People reveal their location information to LBS providers to obtain personalized services such as map directions, restaurant recommendations, and taxi reservations. Usually, LBS providers offer user privacy protection statement to assure users that their private location information would not be given away. However, many LBSs run on third-party cloud infrastructures. It is challenging to guarantee user location privacy against curious cloud operators while still permitting users to query their own location information data. In this paper, we propose an efficient privacy-preserving cloud-based LBS query scheme for the multiuser setting. We encrypt LBS data and LBS queries with a hybrid encryption mechanism, which can efficiently implement privacy-preserving search over encrypted LBS data and is very suitable for the multiuser setting with secure and effective user enrollment and user revocation. This paper contains security analysis and performance experiments to demonstrate the privacy-preserving properties and efficiency of our proposed scheme.

1. Introduction

Location-based services (LBSs) are increasingly popular in today's society. It is reported that up to 150 million people have enjoyed LBSs as early as 2014 [1]. People reveal their location information to LBS providers to obtain personalized services such as map directions, restaurant recommendations, and taxi reservations.

The most common and most important service in an LBS system is a location query service. In LBS applications, a user is able to use his powerful smartphone equipped with GPS modules to obtain accurate location information anytime and anywhere by submitting a query keyword of interest (e.g., hotel) to the LBS system. Upon receiving an location query request from the user, an LBS provider rapidly returns back a target location list, in which all locations are ranked in an ascending order based on distances between the query user and these target locations.

Every coin has two sides: although the LBS greatly facilitates people's life nowadays, the user privacy disclosure problems for LBS applications are more and more serious. The LBS providers can mine LBS users' privacy by analyzing LBS queries or recovering the spatial correlated data [2]. For example, LBS providers can easily obtain user's mobility trace or even infer user's real identity, healthy status, hobbies, and so on [3, 4]. To address the privacy challenge in the LBS system, many solutions have been proposed such as the pseudoidentity technique [5], location fuzzy [6–9], and private information retrieval in the trusted third party (TTP) [5, 7]. These schemes significantly promote the further development of LBS applications.

With the rapid development of the cloud computing, more and more LBS providers are beginning to consider to outsource their location data and services to the cloud server for enjoying the numerous advantages brought by the cloud computing such as economic savings, great flexibility, quick deployment, excellent computation performance, and abundant bandwidth resources. However, the cloud server is not fully trusted usually by the LBS providers due to being

operated by the remote commercial organizations. Once the location data is outsourced to the cloud server in the plaintext form, the data security would not be guaranteed. For example, a corrupted administrator of the cloud server may sell the location data outsourced by the LBS provider to other one for obtaining illegal profit. Presently, the most effective way to protect the confidentiality of outsourced location data is to encrypt data before outsourcing it to the cloud server [10]. On the other hand, the bare user query requests also provide more opportunities for the cloud server to mine user's privacy just like a traditional LBS system. Therefore, the user requests should also be encrypted before being submitted to the cloud server. However, data encryption makes the available location query service a challenging task, since the ciphertext no longer bears the natures of numerical computation and character match in the plaintext field. Therefore, there are two essential problems that need to be solved in a cloud-based LBS application over the encrypted outsourced location data: (1) how to find out all target locations over the encrypted location data according to the encrypted user request; (2) how to compute or compare distances between these target locations and user current location over the encrypted outsourced location data.

A recent work in [11] sets out to explore the challenging issue that how to implement the cloud-based LBS system over encrypted location data and proposes a privacy-preserving cloud-based LBS query scheme, called "EPQ." The scheme enables the cloud server to perform LBS query over the encrypted LBS data without divulging users' location information. However, the scheme only can enforce a user location coordinate query according to a user's current location. In a practical LBS application, a goal-oriented keyword query is necessary for the user to accurately locate locations of interest (e.g., the user may need to accurately search hotels near to him/her). Compared with the existing work, in this paper, we propose an efficient and secure keyword-based query scheme that allows the user to be able to first accurately locate desirable locations according to the encrypted query request and then rank distances between these target locations and the user's current location, which greatly improves the user's location server experiences. Moreover, our scheme is very suitable for a multiuser cloud environment by equipping with flexible users enrollment and users revocation mechanisms.

In this paper, we make the following three key contributions:

(i) First, we propose an efficient and privacy-preserving cloud-based LBS query scheme. For protecting the security of location data and user requests against the curious cloud server, we adopt a hybrid encryption to encrypt the outsourced location data and user requests while the cloud server can still provide accurately LBS query services for users by performing privacy-preserving and efficient search over the encrypted locations data. In addition, our scheme is very suitable for the multiuser setting by equipping with flexible user enrollment and user revocation mechanisms.

(ii) Second, we provide detailed correction analysis and security analysis. The analyses show that our scheme is correct and can achieve user privacy preservation and confidentiality of LBS data, simultaneously.

(iii) Lastly, we implement our scheme in Java and evaluate the performance on a real data set. Experimental results demonstrate that our proposed scheme is efficient and practical.

The rest of our paper is organized as follows. In Section 2, we review some related literatures. In Section 3, we recall a bilinear pairing map, secure kNN, and a difficulty assumption of discrete logarithm problem as the preliminaries. Then, we formalize a system model and a threat model and depict problem statements in Section 4. We present our approach in Section 5. What is more, some analyses and performance evaluations are conducted in Sections 6 and 7, respectively. Finally, we draw our conclusions in Section 8.

2. Related Work

In this section, we review some related works about privacy protection in traditional LBSs and cloud-based LBSs, respectively.

2.1. Traditional LBS Privacy Protection. Privacy leakage problem in traditional LBSs has drawn much attention of researchers, and we review mainly related literatures.

Firstly, a location k-anonymity model is introduced, which guarantees that an individual cannot be identified from $k - 1$ other individuals [12]. What is more, in a distributed environment, an anonymous approach based on homomorphic encryption [13] is proposed to protect location privacy. However, when the anonymous region is sensitive or k individuals are the same place, the sensitive location will still be leaked. Thus the third party (TTP) is proposed to manage the location information centrally [14–16]. To achieve an accurate query, a method is proposed to convert original locations of LBS data and query, maintaining a spatial relationship between the LBS data and query [16]. However, because of many users' sensitive information in the TTP, attackers would aim at attacking it easily. Then a scheme without the TTP is proposed, which protects the locations through private information retrieval [7]. Recently, considering mobile nodes, a distributed anonymizing protocol based on peer-to-peer architecture is proposed [17]. A specific zone is responded by each mobile node. Besides, an information-theoretic notion was introduced to protect privacy in LBS systems [18]. An approach is proposed to protect both client's location privacy and the server's database security by improving the oblivious transfer protocol [19]. For providing privacy-preserving map services, a new multiple mix zones location privacy protection is proposed. By using this method, users are able to query a route between two endpoints on the map, without revealing any confidential location and query information [20].

2.2. Cloud-Based LBS Privacy Protection. Considering the low computation and communication cost, the LBS providers

APSE Scheme
Key: a $(d+1)\times(d+1)$ invertible matrix M.
Tuple Encryption Function E_T**:** Consider an LBS data p which will be stored in a cloud server.
(1) Create a $(d+1)$-dimensional point $\widehat{p} = (p^T, -0.5\|p\|^2)^T$.
(2) The encrypted data $p' = M^T\widehat{p}$.
Query Encryption Function E_Q**:** Consider an LBS query q.
(1) Generate a random number $r > 0$.
(2) Create a $(d+1)$-dimensional point $\widehat{q} = r(q^T, 1)^T$.
(3) The encrypted query $q' = M^{-1}\widehat{q}$.
Distance Comparison Operator A_e**:**
Let p_1' and p_2' be the encrypted data of p_1 and p_2 respectively. To determine whether (p_1) is nearer to a query q than p_2 is, the system checks whether $(p_1' - p_2')\cdot q' > 0$, where q' is the encrypted point of q.
Decryption Function D**:** Consider an encrypted data p'.
The original data $p = \pi_d M^{T^{-1}} p'$, where π_d is a $d\times(d+1)$ matrix which projects on the first d dimensions and $\pi_d = (I_d, 0)$ where I_d is the $d\times d$ identity matrix.

Algorithm 1: APSE scheme.

outsource the LBS data to the cloud server to compute accurate LBS queries, whereas the cloud server is semitrusted. Hence the privacy problem is still a challenge in the cloud-based LBS. There are some literatures about this problem. Firstly, a spatiotemporal predicate-based encryption is proposed for fine-grained access control [21]. Then an improved homomorphic encryption [11] is proposed to protect users' privacy and LBS data privacy. A privacy extension in crowdsourced LBS [22] is proposed. To handle the long-term privacy protection and fake path generation for LBS, a dummy-based long-term location privacy protection [23] is proposed. Recently, two-tier lightweight network coding based on pseudonym scheme in a group LBS [24] is proposed to protect privacy. What is more, a query scheme by using Bloom filter and bilinear pairing is proposed [25]. However, the literatures above did not consider the multiusers condition (i.e., joining of registered users and revocation of expired users). But unregistered users and expired users access for cloud-based LBSs is a typical scenario. Therefore, providing an efficient and privacy-preserving cloud-based LBS in multiuser environments is an unnegligible issue.

3. Preliminaries

In this section, we introduce several necessary tools used in our scheme, including a bilinear pairing map, secure kNN computation techniques, and the difficulty assumption of discrete logarithm problem.

3.1. Bilinear Pairing Map. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplication cyclic groups with large prime order q. A bilinear pairing map [26, 27] $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ satisfies the following properties:

(i) Bilinear: for all $x, y \in \mathbb{Z}_q^*$ and $P, Q \in \mathbb{G}_1$, $e(P^x, Q^y) = e(P^y, Q^x) = e(P, Q)^{xy}$.
(ii) Nondegenerate: if $P, Q \in \mathbb{G}_1$, then $e(P, G) \neq 1 \in \mathbb{G}_2$.
(iii) Computable: for any element $P, Q \in \mathbb{G}_1$, there exists a polynomial time algorithm to compute $e(P, Q)$.

3.2. Secure kNN. Secure kNN [28] enables an efficient kNN computation over encrypted data points. It adopts an asymmetric scalar-product-preserving encryption (ASPE) to achieve a distance comparison between two encrypted data vectors. We synoptically introduce the principle of this technique as follows.

Definition 1 (asymmetric scalar-product-preserving encryption). Let E be an encryption function and $E(p, K)$ be an encryption of a point p under a key K. E is an ASPE if and only if E just preserves the scalar product between encrypted data points and an encrypted query points; that is,

(1) $p_i \cdot q = E(p_i, K)\cdot E(q, K)$, where p_i is one encrypted data point and q is one encrypted query point;
(2) $p_i \cdot p_j \neq E(p_i, K)\cdot E(p_j, K)$, where p_i and p_j are two encrypted data points.

For ease of understanding, we describe the APSE scheme in Algorithm 1. As shown in Algorithm 1, this scheme includes five parts, that is, a key, a tuple encryption function, a query encryption function, a distance comparison operator, and a decryption function.

3.3. Difficulty Assumption of Discrete Logarithm Problem. Given a multiplication group $\mathbb{G}$ with the prime order q, g is its generator. An element x is selected from $\mathbb{Z}_q^*$ randomly, computing $Q = g^x \in \mathbb{G}$. The definition of the difficulty assumption of discrete logarithm problem (DLP) is as follows.

Definition 2 (difficulty assumption of discrete logarithm problem). Given $\mathbb{G}$ and g, it is difficult to compute the correct value of x. In other words, given a tuple $(\mathbb{G}, q, g^x, g)$, there is not an efficient polynomial time algorithm to output x.

4. Background

In this section, we formally introduce our system model and threat model and then state our proposed problem.

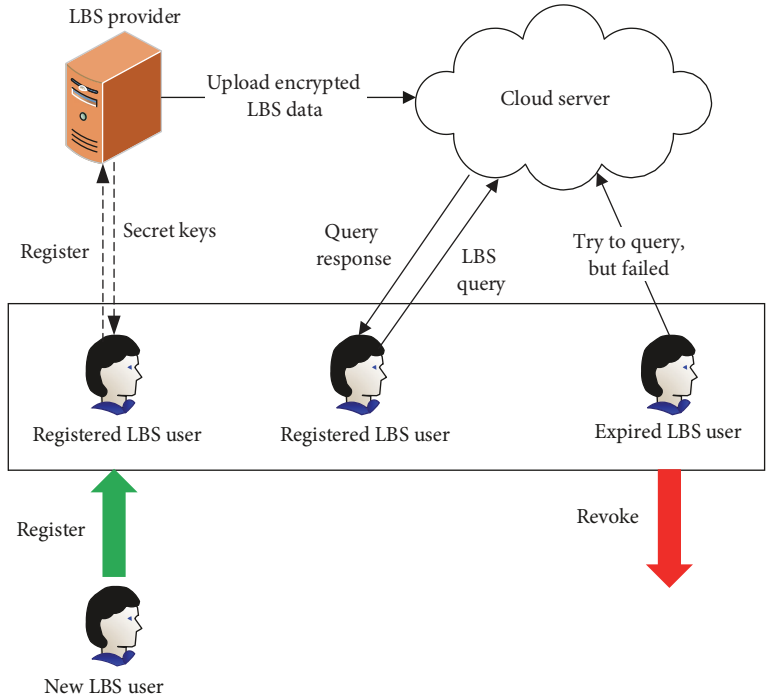

Figure 1: System model.

4.1. System Model. In our system model, there are three entities: an LBS provider, a cloud server, and a group of LBS users, as shown in Figure 1. Next, we introduce each entity of our model as follows.

(i) LBS Provider. An LBS provider is a location data owner. It outsources large-scale location data to the cloud server for enjoying the low-cost storage services and powerful computation services. To ensure the confidentiality of location data, all location data is uploaded to the cloud server after being encrypted by the LBS provider. In addition, when an LBS user wants to join the system, the LBS provider provides authentication and registration service for the LBS user. Once the LBS user passes authentication, the LBS provider sends some important security parameters to the user via secure communication channels. Correspondingly, the LBS provider is also able to revoke any expired LBS user, who no longer has the query capabilities for the outsourced location data when being revoked by the LBS provider.

(ii) A Group of LBS Users. A group of LBS users are the location data users, who enjoy convenient LBSs by submitting LBS query requests to the LBS provider anywhere and anytime. To hide query requests of LBS users for protecting privacy, LBS users first encrypt their query requests and then submit the encrypted query requests to the cloud server. Note that the LBS users are usually referred to the legal registered users and unregistered users and revoked users from the provider cannot enjoy LBSs.

(iii) Cloud Server. Upon receiving the encrypted LBS query request submitted by a legal LBS user, the cloud server is responsible for performing the query over encrypted outsourced location data on behalf of the LBS user and returning the satisfied query results to the LBS user. In the whole query processes, the cloud server does not know any contents about outsourced location data, the user's query request, and the current location of the LBS user.

4.2. Problem Statements. In a conventional LBS system, the LBS data construction usually is organized as the category set and the location data set, as shown in Table 1(a). A *CATEGORY* denotes the general name of location data, which contains multiple concrete location data. Each concrete location data is a four-tuple $\{ID, TITLE, COORDINATE, DESCRIPTION\}$, which describes the detailed information of a certain location. When a registered user searches an interested location, he/she submits the specified CATEGORY and his current location coordinates to the LBS system. The LBS system first searches over the category set according to the submitted CATEGORY to obtain all target locations and then sorts target locations in an ascending order based on the distances between the user's current location and these target locations, which are easily computed according to the user's coordinate and each target location's coordinate, and finally returns the first k nearest locations to the query user, if the query user sends an optional parameter k to the LBS system. It means that the LBS system can analyze what are the LBS user interested in and his/her real-time location, when receiving an LBS query.

To ensure the confidentiality of LBS data and enable registered users to enforce efficient location query in the privacy-preserving manner when the LBS provider outsources LBS data and query services to the cloud server, in this paper,

Table 1: LBS data creation.

(a)

{*CATEGORY*}	Location Data Set {*ID*, *TITLE*, *COORDINATE*, *DESCRIPTION*}
{School:	{
	{1, Hunan University, (x_1, y_1), {211, 985}},
	⋮
	{15, Center South University, (x_{15}, y_{15}), {211, 985}}}.
Hospital:	{
	{16, the Second Xiangya Hospital, (x_{16}, y_{16}), {grade III-A hospital}},
	⋮
	{31, the Union Hospital, (x_{31}, y_{31}), {grade III-A hospital}}}.
⋮	⋮
Hotel:	{
	{450, Huatian Hotel, (x_{450}, y_{450}), {5 stars}},
	⋮
	{500, Westin Hotel, (x_{500}, y_{500}), {4 stars}}}.
}	

(b)

{*CATEGORY*}	Location Data Set {*ID*, *TITLE*, *COORDINATE*, *DESCRIPTION*}
{E_1(School):	{
	{1, E_3(Hunan University), $E_2((x_1, y_1))$, $E_3(\{211, 985\})$},
	⋮
	{15, E_3(Center South University), $E_2((x_{15}, y_{15}))$, $E_3(\{211, 985\})$}}.
E_1(Hospital):	{
	{16, E_3(the Xiangya Hospital), $E_2((x_{16}, y_{16}))$, E_3({grade III-A hospital})},
	⋮
	{31, E_3(the Union Hospital), $E_2((x_{31}, y_{31}))$, E_3({grade III-A hospital})}}
⋮	⋮
E_1(Hotel):	{
	{450, E_3(Huatian Hotel), $E_2((x_{450}, y_{450}))$, E_3({5 stars})},
	⋮
	{500, E_3(Westin Hotel), $E_2((x_{500}, y_{500}))$, E_3({4 stars})}}.
}	

we adopt a hybrid encryption mechanism to encrypt the LBS data; the encryption version of LBS data is shown in Table 1(b), where E_1, E_2, and E_3 denote different encryption scheme, respectively, whose constructions will be formally proposed in the next section. By encrypting different fields of LBS data using the different encryptions E_1, E_2, and E_3, our scheme allows the cloud server to provide totally the same query service over encrypted location data as the plaintext environment aforementioned while information about the location data and user's query request is exposed to the cloud server.

From the point of view of LBS users, compared with the LBS system in the plaintext environment, the only difference in our scheme is that an LBS user needs to encrypt the interested *GATEGORY* and his/her location coordinates to generate a query trapdoor. The cloud server performs the LBS query over encrypted outsourced location data according to the query trapdoor. Of course, the necessary decryption operations need to be involved for the LBS user once the encrypted LBS query results are received; however, this is not our concerned problem in this paper.

In addition, the LBS system is a typical multiuser application, our scheme designs efficient and flexible user registration and user revocation mechanisms to guarantee that only registered users are able to use the LBS system and unregistered users or revoked users have not access to it.

5. A Privacy-Preserving Multiuser LBS Query Scheme Based on Hybrid Encryption

In this section, we describe the implementation details of our privacy-preserving multiuser LBS query scheme. From the system-level point of view, our scheme includes six key modules: system initialization, new user grant, location data encryption, query trapdoor generation, search over encrypted location data, and user revocation. Each module is operated by one entity independently or multiple entities interactively and all modules integrally constitute our privacy-preserving multiuser LBS query system.

5.1. System Initialization. The system initialization operation is executed by the LBS provider to set up the system running environment. The LBS provider takes a large security parameter l as input and first generates two multiplication cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with the large prime order q equipping the bilinear pairing map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Let g be a generator of $\mathbb{G}_1$. Then, the algorithm defines a cryptographical hash function $h_1 : \{0,1\}^* \to \mathbb{G}_1$, which maps a message of arbitrary length to an element in $\mathbb{G}_1$. Lastly, the algorithm chooses a random value $x \in \mathbb{Z}_q^*$ and generates a 3×3 invertible matrix M that are kept secretly by the LBS provider and opens the public parameter $PK = \{\mathbb{G}_1, \mathbb{G}_2, e, q, g, h_1\}$.

5.2. New User Grant. When a new LBS user u wants to join the system, the LBS provider registers the new user in this phase. At first, the LBS provider selects a random value $x_u \in \mathbb{Z}_q^*$ for u and computes g^{x/x_u} and the inverse matrix M^{-1} of M. Then, g^{x/x_u}, M^{-1}, and x_u are sent to the user u by secure communication channels. When u receives g^{x/x_u}, M^{-1}, and x_u, he/she randomly selects $r_u \in \mathbb{Z}_q^*$ and keeps r_u, M^{-1} secretly and then further computes g^{r_u}.

According to the received value g^{x/x_u}, u computes his/her register secret key $Enrk_u$:

$$Enrk_u = g^{x/x_u} \times g^{r_u} = g^{x/x_u + r_u}. \tag{1}$$

At last, $(u, Enrk_u)$ is sent to the cloud server and the cloud server stores this tuple into a user list *U-Enrkey*.

5.3. Location Data Encryption. To guarantee the security of location data, the LBS provider needs to encrypt all location information before outsourcing them to the cloud server. In this paper, to enable an efficient and privacy-preserving LBS query, we use different encryption mechanisms to encrypt the different attributes of the location data. Without loss of generality, we use $\{ID : ID_d, TITLE : T_d; COORDINATE : (x_d, y_d); DESCRIPTION : D_d\}$ to denote any location data d belongs to *CATEGORY* C_d in category set. The LBS provider takes the following three steps to encrypt the location data d.

First, the LBS provider uses its secret value x to code C_d as $h_1(C_d)^x$ and further employs the bilinear pairing map e to calculate $e(h_1(C_d)^x, g)$. We use E_1 to denote the code operation of *CATEGORY* attribute of the location data such that

$$E_1(C_d) = e\left(h_1(C_d)^x, g\right). \tag{2}$$

Second, the LBS provider uses the secretly preserved invertible matrix M to encrypt d's coordinate (x_d, y_d) as

$$E_2(x_d, y_d) = M^T\left(x_d, y_d, -0.5\left(x_d^2 + y_d^2\right)\right)^T. \tag{3}$$

Here, correspondingly, we use E_2 to denote this encryption operation.

Third, for the remaining other attributes TITLE and DESCRIPTION, the LBS provider adopts any semantically secure symmetric encryption such as AES under a given key sk to encrypt them, denoted as E_3 in our paper such that

$$\begin{aligned} E_3(T_d) &= AES_{sk}(T_d), \\ E_3(D_d) &= AES_{sk}(D_d). \end{aligned} \tag{4}$$

5.4. Query Trapdoor Generation. To preserve user's query privacy and enable correct search over encrypted location data, a query user u with the current location coordinate (x_u, y_u) needs to encrypt his/her query request before it is submitted to the cloud server. In this paper, a query trapdoor generation is conducted in two steps.

First, the user u chooses a desired query objective denoted as q (e.g., $q = Hotel$) and uses the secret value x_u granted by the LBS provider and the secret value r_u randomly chosen by himself/herself in the user grant phase to encrypt q as $h_1(q)^{x_u}$ and $h_1(q)^{x_u r_u}$.

Second, the user u generates a random number $r > 0$ and uses the matrix M^{-1} granted by the LBS provider to encrypt the current location coordinate (x_u, y_u) as $M^{-1}(rx_u, ry_u, r)^T$. Ultimately, the query trapdoor of q can be denoted as follows:

$$\mathcal{T}_u(q) = \left(h_1(q)^{x_u}, h_1(q)^{x_u r_u}, M^{-1}(rx_u, ry_u, r)^T\right). \tag{5}$$

5.5. Search over Encrypted Location Data. After the query user u generates a query trapdoor $\mathcal{T}_u(q)$, he/she submits $\mathcal{T}_u(q)$ and a parameter k to the cloud server. Upon receiving the query trapdoor $\mathcal{T}_u(q)$ and k, the powerful cloud server is responsible for searching over encrypted outsourced location data on behalf of the query use, without knowing any plaintext information of the outsourced location data and the user query request. If the user u is a legal user, the cloud server returns back the first k encrypted target locations that satisfy the query and are nearest to the query user. Therefore, in the whole query process, the cloud server must perform two key operations under the encrypted environment: (1) searching the encrypted category set according to the query trapdoor to obtain all target locations; (2) sorting the distances between target locations and user's current location in an ascending order. To achieve the above goal, the cloud sever processes the search in two steps.

First, the cloud server looks up the query user u's registration information $\langle u, Enrk_u \rangle$ from the user list *U-Enrkey*. If the user information does not exist, the cloud server rejects the query; otherwise it linearly scans the encrypted category set and obtains all encrypted target location data if it finds out

an encrypted *CATEGORY* $E_1(C)$ in the encrypted category set that satisfies the following equation:

$$E_1(C) = \frac{e\left(Enrk_u, h_1(q)^{x_u}\right)}{e\left(g, h_1(q)^{x_u r_u}\right)}. \quad (6)$$

Second, upon obtaining all target locations, the cloud server sorts the distances between target locations and user's query location by evaluating

$$\left(M^T\left(x_i, y_i, -0.5\left(x_i^2 + y_i^2\right)\right)^T - M^T\left(x_j, y_j, -0.5\left(x_j^2 + y_j^2\right)\right)^T\right) \cdot \left(M^{-1}\left(rx_u, ry_u, r\right)^T\right) > 0, \quad (7)$$

where i and j are any two locations satisfying the query with the encrypted coordinate $E_2(x_i, y_i)$ and $E_2(x_j, y_j)$, respectively.

If the above inequality holds, then this indicates that the target location i is closer to the query user u than the target location j. Hence, i is sorted in the front of j. Finally, the cloud server returns the first k encrypted locations to the query user u.

5.6. User Revocation. User revocation is an essential yet challenging task in a practical multiuser application such as an LBS system. In some related literatures supporting user revocation, to prevent revoked users from continuing to access outsourced cloud data, the data provider usually has to rebuild data index or reencrypt large amounts of data and reuploads them to the cloud server and issues new keys to the remaining users. It unavoidably brings heavy computation and communication cost for the data provider because of the high of dynamic of users in the cloud environment. In this paper, we propose an efficient user revocation mechanism without any data reencryption and keys update operations while being able to effectively prevent the revoked user from searching outsourced location data. More concretely, for a user u' who will be revoked by the LBS provider, the LBS provider first sends the user information about u' to the cloud server. Then, the cloud server scans user information in the user list *U-Enrkey* to find out the information of u' and deletes $(u', Enrk_{u'})$. Once $(u', Enrk_{u'})$ is deleted from *U-Enrkey*, u' no longer has the capability to search location data stored at the cloud server. Since without $Enrk_{u'}$, the cloud server cannot complete matching between the trapdoor and encrypted *CATEGORY* according to the query scheme proposed in Section 5.5, u' can still generate a legal query trapdoor.

6. Analysis

In this section, we analyze the search correctness and security to prove that our proposed scheme is correct and secure.

6.1. Search Correctness Analysis. When an authorized query user u submits his/her query trapdoor $\mathcal{T}_u(q)$ to the cloud server, the cloud server firstly obtains the all encrypted locations satisfying the query q by performing a matching operation between an encrypted *CATEGORY* and $\mathcal{T}_u(q)$. Specifically, the cloud server judges whether $E_1(C) = e(Enrk_u, h_1(q)^{x_u})/e(g, h_1(q)^{x_u y_u})$ holds or not for an encryption *CATEGORY* $E_1(C)$. If the equation holds, then this indicates that the query q correctly matches $E_1(C)$ and the cloud server obtains all target locations belong to *CATEGORY* C. The correctness can be easily verified as follows:

$$\begin{aligned}
\frac{e\left(Enrk_u, h_1(q)^{x_u}\right)}{e\left(g, h_1(q)^{x_u r_u}\right)} &= \frac{e\left(g^{x/x_u + r_u}, h_1(q)^{x_u}\right)}{e\left(g, h_1(q)^{x_u r_u}\right)} \\
&= \frac{e\left(g^{x/x_u}, h_1(q)^{x_u}\right) e\left(g^{r_u}, h_1(q)^{x_u}\right)}{e\left(g, h_1(q)^{x_u r_u}\right)} \\
&= \frac{e\left(g, h_1(q)^{x_u \cdot (x/x_u)}\right) e\left(g, h_1(q)^{x_u \cdot r_u}\right)}{e\left(g, h_1(q)^{x_u r_u}\right)} \\
&= e\left(h_1(q)^x, g\right) \\
&= e\left(h_1(C)^x, g\right) \quad (C = q) \\
&= E_1(C).
\end{aligned} \quad (8)$$

Then, for any two target locations i and j, the cloud server is able to determine whether i is closer to the query current location than j by evaluating

$$\left(M^T\left(x_i, y_i, -0.5\left(x_i^2 + y_i^2\right)\right)^T - M^T\left(x_j, y_j, -0.5\left(x_j^2 + y_j^2\right)\right)^T\right) \cdot \left(M^{-1}\left(rx_u, ry_u, r\right)^T\right) > 0. \quad (9)$$

This is because that

$$\begin{aligned}
&\left(M^T\left(x_i, y_i, -0.5\left(x_i^2 + y_i^2\right)\right)^T - M^T\left(x_j, y_j, -0.5\left(x_j^2 + y_j^2\right)\right)^T\right) \cdot \left(M^{-1}\left(rx_u, ry_u, r\right)^T\right) \\
&= \left(M^T\left(x_i, y_i, -0.5\left(x_i^2 + y_i^2\right)\right)^T - M^T\left(x_j, y_j, -0.5\left(x_j^2 + y_j^2\right)\right)^T\right)^T \\
&\cdot \left(M^{-1}\left(rx_u, ry_u, r\right)^T\right) = \left(\left(x_i, y_i, -0.5\left(x_i^2 + y_i^2\right)\right)^T - \left(x_j, y_j, -0.5\left(x_j^2 + y_j^2\right)\right)^T\right)^T \\
&\cdot M^T M^{-1}\left(\left(rx_u, ry_u, r\right)^T\right) = 0.5r\left(\left(x_j^2 + y_j^2\right)\right.
\end{aligned}$$

$$
\begin{aligned}
&-\left(x_i^2+y_i^2\right)+2\left(x_ix_u+y_iy_u-x_jx_u-y_jy_u\right)\Big)\\
&=0.5r\left(d\left(\left(x_j,y_j\right),\left(x_u,y_u\right)\right)\right.\\
&\left.-d\left(\left(x_i,y_i\right),\left(x_u,y_u\right)\right)\right),
\end{aligned}
\tag{10}
$$

where $d((x_i, y_i), (x_u, y_u))$ ($d((x_j, y_j), (x_u, y_u))$, resp.) denotes the Euclidean distance between the location i (j, resp.) and the user's current location. So,

$$
\begin{aligned}
&\left(M^T\left(x_i,y_i,-0.5\left(x_i^2+y_i^2\right)\right)^T\right.\\
&\left.-M^T\left(x_j,y_j,-0.5\left(x_j^2+y_j^2\right)\right)^T\right)\\
&\cdot\left(M^{-1}\left(rx_u,ry_u,r\right)^T\right)>0\\
&\Longleftrightarrow 0.5r\left(d\left(\left(x_j,y_j\right),\left(x_u,y_u\right)\right)\right.\\
&\left.-d\left(\left(x_i,y_i\right),\left(x_u,y_u\right)\right)\right)>0\\
&d\left(\left(x_j,y_j\right),\left(x_u,y_u\right)\right)>d\left(\left(x_i,y_i\right),\left(x_u,y_u\right)\right).
\end{aligned}
\tag{11}
$$

Thus, the cloud server is able to sort all target locations according to the above distance comparisons in an ascending order and returns back the first k nearest locations to the query user.

6.2. Security Analysis. In our proposed scheme, three encryption schemes E_1, E_2, and E_3 are employed to protect the confidentiality of LBS data. In this section, we will analyze the security of our scheme against the "honest-but-curious" cloud server in the multiuser environment.

E_2 is a semantically secure symmetric encryption such as AES that encrypts the TITLE and DESCRIPTION fields of LBS data. The semantic security of AES guarantees the security of TITLE and DESCRIPTION fields of LBS data. We use secure kNN encryption technique denoted as E_2 to encrypt the *COORDINATE* attribute of LBS data and query user's coordinate to enable a secure and effective distance comparison. The security of the *COORDINATE* attribute of LBS data and the query user's coordinate mainly relies on the security of secure kNN scheme. For the *CATEGORY* attribute of LBS data, we use E_3 to encrypt it to enable secure and flexible search over encrypted location data. Specifically, given a location data with *CATEGORY* attribute C, the ciphertext can be denoted as $E_3(C) = e(h_1(C)^x, g) = e(h_1(C), g)^x$. Since $e(h_1(C), g)$ is a group element in $\mathbb{G}_2$ with a large prime order, the secret key x is acknowledgedly intractable from $E_3(C)$ in the large number field due to the well-known DLP assumption. Without the secret key x kept secretly by the LBS provider, the cloud server cannot recover the message C from encryption E_3.

In addition, in the multiuser environment, the system should prevent an illegal user from requesting for query LBS data stored in the cloud server. In our scheme, when a registered user u wants to query the LBS location data using q, u uses secret query keys x_u and r_u to generate the query trapdoor of q, $trap_u(q) = (h_1(q)^{x_u}, h_1(q)^{x_u r_u})$. Under the assumption of DLP, attackers cannot compute out x_u and r_u according to $trap_u(q)$. Without the correct x_u and r_u, an illegal user u_I cannot generate the correct query trapdoor such that u_I cannot perform the correct query over encrypted location data. For a revoked user u_R, although u_R can generate the trapdoor $trap_{u_R}(q)$ for q, u_R still cannot let the cloud server perform a correct query on behalf of him/her due to lacking of the necessary query parameter $Enrk_{u_R}$ that has been deleted from the list *U-Enrkey* by the cloud server in the phase of user revocation.

7. Evaluation

In this section, we evaluate the performance of our proposed scheme from the perspective of the LBS provider, the LBS user, and the cloud server, respectively. The software and hardware configurations of the LBS provider and LBS user side are Windows 7 desktop systems with Intel Core 2 Duo CPU 2.26 GHZ, 3 GB memory, and 320 GHZ hard driver and the cloud sever side is a virtual machine with Core 2 Duo CPU 4 × 2.394 GHZ, 8 GB memory on the Dell blade sever M610, and the Linux Centos 5.8 OS.

All programs are developed using Java language and the JPBC library [29] is employed to implement group operations. We execute all experiments in a real data set collected from the *open street map* [30] with 50 categories and the number of concrete location data being 1000 by extracting the location data belonging to *Yuelu* District, Changsha, China.

7.1. LBS Data Encryption. Figure 2(a) shows that the time cost of encrypting LBS data for the LBS provider increases linearly with the increasing size of category set when the total number of location data remains unchanged ($n = 1000$). Figure 2(b) shows the number of concrete location data has little influence on the time cost of encrypting LBS data when fixing the size of category set ($c = 50$). Recall that, in our scheme, E_1 is used to encrypt categories in category set and E_2 and E_3 are used to encrypt location data. Experimental results from Figure 2 illustrate that the time cost of encrypting LBS data is closely related to the encryption E_1 while not being almost affected by E_2 and E_3. It is reasonable that the E_1 consists of the time-consuming pair operation and exponent operation over the ellipse curve group while E_2 and E_3 almost do not consume time when an extremely small message and 3-dimensional vector are encrypted by them, respectively.

7.2. Query Request Encryption. According to the query trapdoor generation proposed in the Section 5.4, in the whole processes of the query request encryption, three key operations are involved to encrypt an interested query keyword and current location coordinate for a registered LBS user (i.e., the hash operation, the exponentiation operation on group, and the matrix multiplication operation between a 3 × 3 matrix and a 3-dimensional column vector). The time cost of each operation based on our software/hardware setting is shown in Table 2. Therefore, the total time cost of generating a query

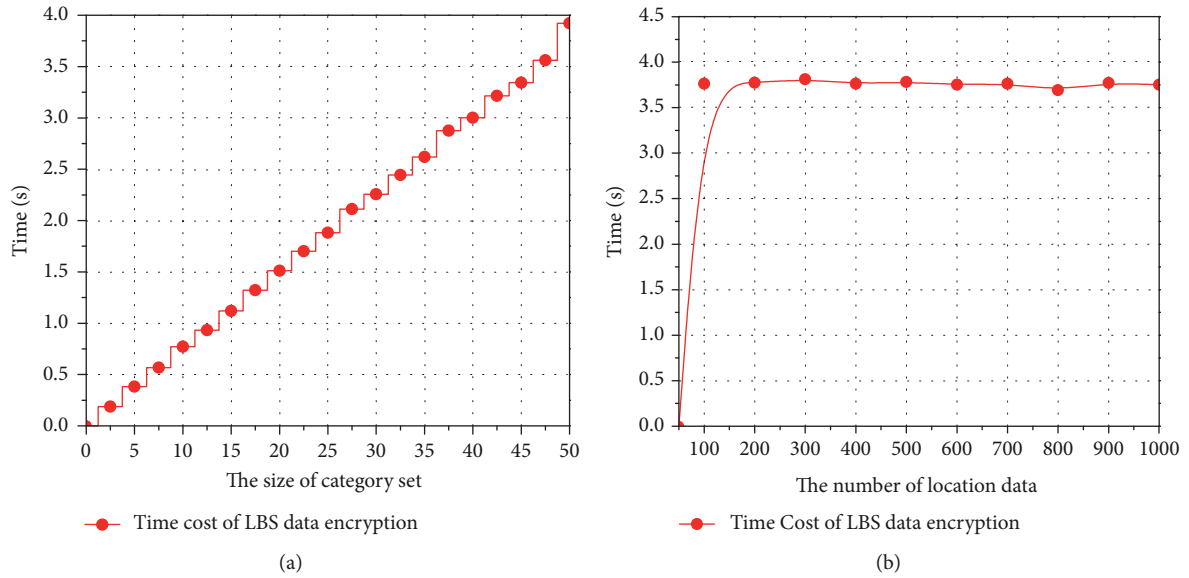

Figure 2: (a) The time cost of encrypting LBS data for LBS provider for different size of category set with fixed number of location data, $n = 1000$; (b) the time cost of encrypting location data for LBS provider for different number of location data with fixed size of category set, $c = 50$.

Table 2: Time cost of operation.

Notations	Descriptions	Time (ms)
h_1	Hash function $h_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$	≈93
$E_{\mathbb{G}_1}$	Exponentiation operation on group $\mathbb{G}_1$	≈34
M	Matrix multiplication operation	<1

request for an LBS user can be denoted as $h_1 + 2 * E_{\mathbb{G}_1} + M \approx$ 161 ms; it is extremely efficient in practice.

7.3. Query over Encrypted LBS Data. Figures 3(a) and 3(b) show the time cost of search over encrypted location data for the cloud server. We can observe that the number of categories and encrypted location data have little influence on the overhead of search for the cloud server. This is because that the main time cost for the search is to only enforce two relatively time-consuming pairing operations while linear search over 50 categories according to the query trapdoor for target location data and 3-dimensional vector computation for distance comparison almost does not consume time.

Figure 3(c) shows the average response time of our query scheme for different sizes of query users. We can see that the response time grows linearly with the increasing number of query users. When the number of query users achieves 100, the response time is about 6.82 s, and this is extremely efficient in practical application.

8. Conclusion

In this paper, we propose a privacy-preserving multiuser LBS query scheme based on the hybrid encryption in the cloud environment. Adopting different encryptions on different attributes of LBS data, our proposed scheme can achieve users' location privacy protection and the confidentiality of LBS data. In particular, the LBS query is performed in the cipher environment, thus the LBS users can get the accurate LBS query results without disclosing their private information. Besides, we consider LBS user accountability and LBS user dynamics, for preventing the unregistered users and expired users accessing. And extensive experiments show that our proposed scheme is highly efficient. In the future, we will consider collusion attacks in the cloud-based LBSs.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Authors' Contributions

Hui Yin and Zheng Qin equally contributed to this work.

Acknowledgments

This work is partially supported by the National Science Foundation of China under Grants nos. 61472131 and 61772191; the Science and Technology Key Projects of Hunan Province under Grants nos. 2015TP1004, 2015SK2087, 2015JC1001, and 2016JC2012; the Natural Science Foundation of Hunan Province under Grant no. 2017JJ2292; Outstanding Youth Research Project of Provincial Education Department of Hunan under Grant no. 17B030; the Science and Technology Planning Project of Changsha under Grant no. k1705018.

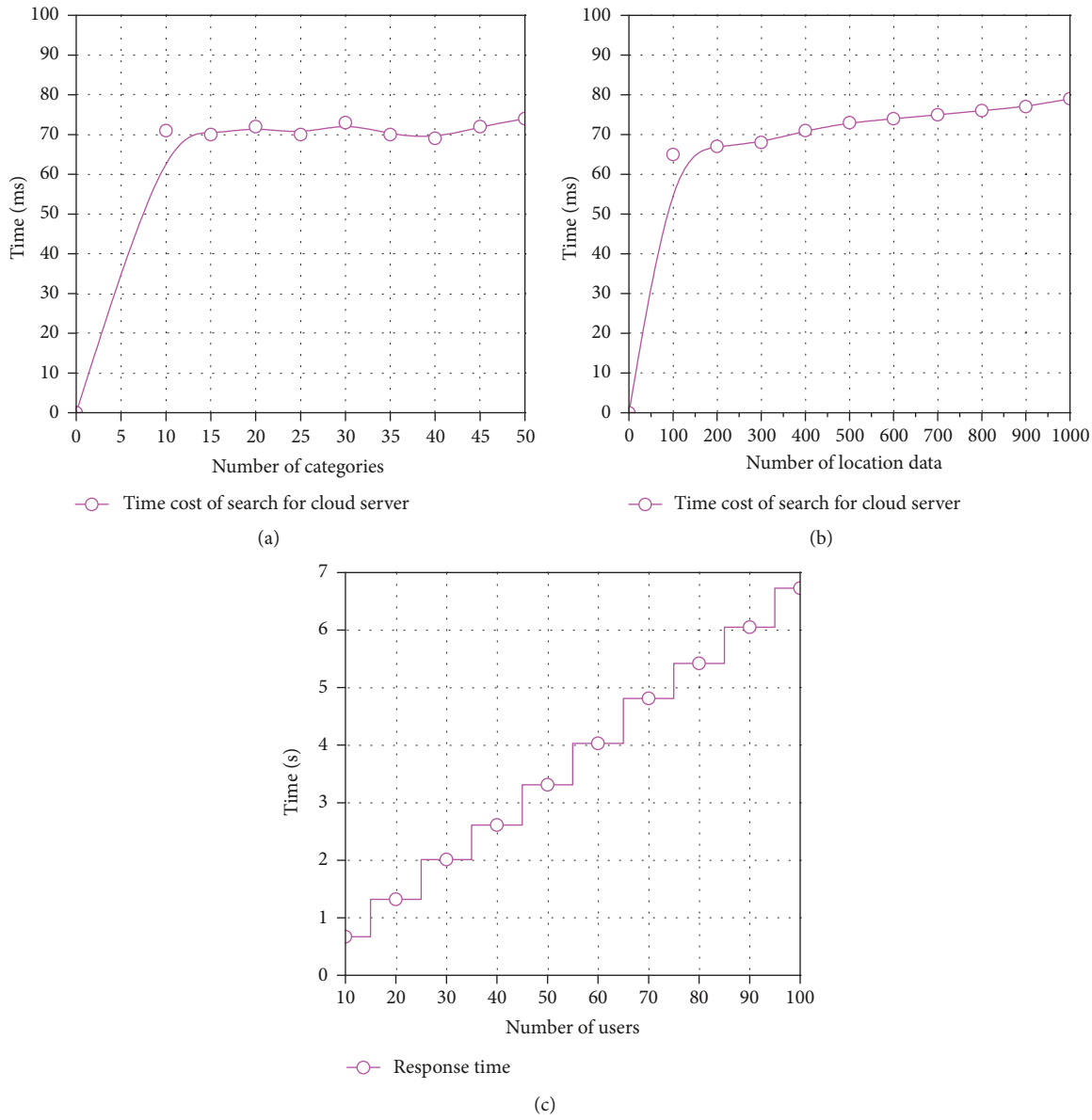

FIGURE 3: (a) The time cost of search for cloud server for different number of categories with fixed number of location data, $n = 1000$; (b) the time cost for search for cloud server for different number of location data with fixed number of categories; (c) the system response time for different number of search users with fixed number of categories and location data, $n = 1000$, $c = 50$.

References

[1] Statista, "Number of location-based service users in the United S tates from 2013 to 2018 (in millions)," Statista; 2017. https://www .statista.com/statistics/436071/location-based-service-users-usa/.

[2] K. Xie, X. Ning, X. Wang et al., "Recover corrupted data in sensor networks: a matrix completion solution," *IEEE Transactions on Mobile Computing*, vol. 16, no. 5, pp. 1434–1448, 2017.

[3] M. Li, H. Zhu, Z. Gao et al., "All your Location are belong to us: breaking mobile social networks for automated user location tracking," in *Proceedings of the 15th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2014*, pp. 43–52, USA, August 2014.

[4] J. Shao, R. Lu, and X. Lin, "FINE: a fine-grained privacy-preserving location-based service framework for mobile devices," in *Proceedings of the IEEE INFOCOM*, pp. 244–252, IEEE, Ontario, Canada, May 2014.

[5] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '08)*, pp. 121–132, ACM, 2008.

[6] B. Gedik and L. Liu, "Protecting location privacy with personalized k-anonymity: architecture and algorithms," *IEEE Transactions on Mobile Computing*, vol. 7, no. 1, pp. 1–18, 2008.

[7] R. Paulet, M. G. Kaosar, X. Yi, and E. Bertino, "Privacy-preserving and content-protecting location based queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1200–1210, 2014.

[8] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 571–588, 2002.

[9] Q. Wang, C. Xu, and M. Sun, "Multi-dimensional K-anonymity based on mapping for protecting privacy," *Journal of Software*, vol. 6, no. 10, pp. 1937–1944, 2011.

[10] K. Xie, X. Ning, X. Wang et al., "An efficient privacy-preserving compressive data gathering scheme in WSNs," *Information Sciences*, vol. 390, pp. 82–94, 2017.

[11] H. Zhu, R. Lu, C. Huang, L. Chen, and H. Li, "An efficient privacy-preserving location-based services query scheme in outsourced cloud," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 9, pp. 7729–7739, 2016.

[12] 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*, pp. 31–42, ACM, San Francisco, Calif, USA, May 2003.

[13] G. Zhong and U. Hengartner, "Toward a distributed k-anonymity protocol for location privacy," in *Proceedings of the 7th ACM Workshop on Privacy in the Electronic Society (WPES '08)*, pp. 33–37, ACM, Alexandria, VA, USA, October 2008.

[14] C.-Y. Chow, M. F. Mokbel, and W. G. Aref, "Casper*: query processing for location services without compromising privacy," *ACM Transactions on Database Systems (TODS)*, vol. 34, no. 4, Article ID 24, 2009.

[15] B. Gedik and L. Liu, "Location privacy in mobile systems: a personalized anonymization model," in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS '05)*, pp. 620–629, IEEE, June 2005.

[16] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Proceedings of the Advances in Spatial and Temporal Databases*, pp. 239–257, Springer, 2007.

[17] M. Ghaffari, N. Ghadiri, M. H. Manshaei, and M. S. Lahijani, "P^4QS: a peer-to-peer privacy preserving query service for location-based mobile applications," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 10, pp. 9458–9469, 2017.

[18] Z. Montazeri, A. Houmansadr, and H. Pishro-Nik, "Achieving perfect location privacy in wireless devices using anonymization," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2683–2698, 2017.

[19] H. Jannati and B. Bahrak, "An oblivious transfer protocol based on elgamal encryption for preserving location privacy," *Wireless Personal Communications*, vol. 97, no. 2, pp. 1–11, 2017.

[20] 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, Article ID e3312, 2017.

[21] Y. Zhu, D. Ma, D. Huang, and C. Hu, "Enabling secure location-based services in mobile cloud computing," in *Proceedings of the 2013 2nd ACM SIGCOMM Workshop on Mobile Cloud Computing, MCC 2013*, pp. 27–32, China, August 2013.

[22] J. B. Abdo, T. Bourgeau, J. Demerjian, and H. Chaouchi, "Extended privacy in crowdsourced location-based services using mobile cloud computing," *Mobile Information Systems*, vol. 2016, Article ID 7867206, 13 pages, 2016.

[23] F. Tang, J. Li, I. You, and M. Guo, "Long-term location privacy protection for location-based services in mobile cloud computing," *Soft Computing*, vol. 20, no. 5, pp. 1735–1747, 2016.

[24] J. Y. Chen and C. L. Wang, "Privacy protection for mobile cloud data: a network coding approach," https://arxiv.org/abs/1701.07075.

[25] H. Yin, Z. Qin, L. Ou, and K. Li, "A query privacy-enhanced and secure search scheme over encrypted data in cloud computing," *Journal of Computer and System Sciences*, vol. 90, pp. 14–27, 2017.

[26] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 586–615, 2003.

[27] H. Yin, Z. Qin, J. Zhang, and K. Li, "Achieving secure, universal, and fine-grained query results verification for secure search scheme over encrypted cloud data," *IEEE Transactions on Cloud Computing*, no. 99, 2017.

[28] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure kNN computation on encrypted databases," in *Proceedings of the International Conference on Management of Data and 28th Symposium on Principles of Database Systems (SIGMOD-PODS '09)*, pp. 139–152, Providence, RI, USA, July 2009.

[29] A. de Caro and V. Iovino, "jPBC: java pairing based cryptography," in *Proceedings of the 16th IEEE Symposium on Computers and Communications (ISCC '11)*, pp. 850–855, July 2011.

[30] Openstreetmap Foundation UK West Midlands, Openstreetmap. Openstreetmap Foundation West Midlands, U, K; 2017, http://www.openstreetmap.org/#map=10/1.3375/103.9732.