Search Me If You Can: Privacy-preserving Location Query Service

Xiang-Yang Li[‡] and Taeho Jung[†]
Department of Computer Science, Illinois Institute of Technology, Chicago, IL
[‡]xli@cs.iit.edu, [†]tjung@hawk.iit.edu

***Abstract*—Location-Based Service (LBS) becomes increasingly popular with the dramatic growth of smartphones and social network services (SNS), and its context-rich functionalities attract considerable users. Many LBS providers use users' location information to offer them convenience and useful functions. However, the LBS could greatly breach personal privacy because location itself contains much information. Hence, preserving location privacy while achieving utility from it is still an challenging question now. This paper tackles this non-trivial challenge by designing a suite of novel fine-grained Privacy-preserving Location Query Protocol (PLQP). Our protocol allows different levels of location query on encrypted location information for different users, and it is efficient enough to be applied in mobile platforms.**

I. Introduction

Location Based Service (LBS) has become one of the most popular mobile applications due to the wide use of smartphones. The smartphones, equipped with GPS modules, have powerful computation ability to process holders' location information, and this brought the flood of LBS applications in the smartphone ecosystem. A good example is the smartphone camera: if one takes a photo with a smartphone camera, the location where the photo is taken is embedded in the picture automatically, which helps one's remembrance. Furthermore, the explosive growth of social network services (SNS) also assisted its growth by constructing connections between location information and social network. When a picture taken by a smartphone (location embedded) is uploaded to the Facebook album, the system automatically shows the location of the picture on the map, and this is shared with the owner's friends in the Facebook (unless the privacy setting specifies otherwise).

Many similar applications exploit both LBS and SNS. They offer several attractive functions, but location information contains much more information than barely the location itself, which could lead to unwanted information leakage. For example, when Alice and Bob both use check-in application in Facebook (which leaves a location record in one's webpage) in a nice restaurant, it is inferable that they are having a date and that they could be in a relationship. This inference might be an unintended information leakage from Alice's and Bob's perspective. Therefore, a privacy-preserving protocol is needed to prevent significant privacy breach resulted from the combination of LBS and SNS.

The simplest way, which most of applications adopted, is to exert group based access control on published locations: specify a group of user who can or cannot see them. Social photo sharing website Flickr only let users choose all users, neighbours, friends or family to allow the access to the locations, and SNS websites Facebook and Google+ additionally support custom groups to specify the accessible user groups. Mobile applications are much worse. Many mobile applications (e.g., Circle, Who's around and Foursquare) even do not offer group choices to the users, instead, they only ask users whether they want to disclose the location or not. Obviously, this is too simple to achieve what users need. First of all, from users' perspective, it is hard to explicitly determine a user group such that their locations are visible only to them. It is more natural to find a condition such that friends who satisfy it can or cannot see the location. Secondly, binary access control (can or cannot) is far beyond enough to properly configure the privacy setting. In the previous example of the two lovers Alice and Bob, Alice might want to share her date at the restaurant with her best friends and discloses the exact location to them. Besides, Alice might also want other friends to know that she is having a good time in downtown, but not detailed location. In this case, approximate settings between 'can' and 'cannot' are needed to fulfil her requirements.

As discussed above, existing privacy control settings in LBS are 'coarse' in the sense that: 1) users can only explicitly specify a group of users who can or cannot access the location information; 2) access control policy supports binary choices only, which means users can only choose to enable or disable the information disclosure. The existing control strategies also suffer from privacy leakage in terms of the server storage. Even if one disables all of the location disclosure, his location is still open to the server, which in fact is users' top concern. Therefore, a fine-grained privacy control executable on encrypted location data is needed to further foster the LBS and its related business market.

A. Contributions

This paper proposes a fine-grained Privacy-preserving Location Query Protocol (PLQP) which enables queries to get location information (e.g., Searching a friend's approximate location, Finding nearest friends) without violating users location privacy. This is not a trivial job since simple anonymiza-

[1]The research of Xiang-Yang Li is partially supported by NSF CNS-0832120, NSF CNS-1035894, NSF ECCS-1247944, National Natural Science Foundation of China under Grant No. 61170216, No. 61228202, China 973 Program under Grant No.2011CB302705.

tion makes it impossible to utilize them for queries. Also, if one directly applies queries or functions on the raw location information, privacy leakage is inevitable. Main contributions of our work are three-fold.

- *Fine-Grained Access Control*: Our protocol allows users to specify a condition instead of a group and exert access control over the users who satisfy this condition. This is more scalable since users can simply add a new condition for new privacy setting instead of hand-picking hundreds of users to form a new group. Also, this is more user-friendly because users themselves do not clearly know which of their friends should or should not access the information most of time.
- *Multi-leveled Access Control*: The protocol also supports semi-functional encryption. That is, the protocol enables users to control to what extent (or level) others can learn his location. The lowest level corresponds to nothing, and the highest level corresponds to one's exact location. Levels between them correspond to indirect information about one's location.
- *Privacy-Preserving Protocol*: In our protocol, every location information is encrypted and queries are processed upon ciphertexts. Therefore, a location publisher's friends learn nothing but the result of the location query, which is under the location publisher's control. In addition, since every location is encrypted, even the server who stores location information does not learn anything from the ciphertext.

II. Related Work

There are several works achieving privacy-preserving location query [1]–[4], which are based on k-anonymity model. The k-anonymity model [5] has been widely used to protect data privacy. The basic idea is to remove some features such that each item is not distinguishable among other k items. However, relevant techniques which achieve k-anonymity of data cannot be used in our case for the following four reasons: 1) Those techniques protect the privacy of the data stored in servers. In our PLQP, we do not store the data at all. 2) In LBS, location data is frequently updated, and this dynamic behaviour introduces huge overhead to keep the data k-anonymous. 3) As analyzed in Zang *et al.* [6], achieving k-anonymity in location dataset significantly violate the utility of it even for small k, so it is not suitable for our location query protocol. 4) k is generally a system-wide parameter which determines the privacy level of all data in the system, but our goal is to leave the decision of privacy level to each user.

Kido *et al.* [7] proposed a scheme which appends multiple false locations to a true one. The LBS responds to all the reports, and the client only collects the response corresponding to the true location. They examined this dummy-based technique and predicted how to make plausible dummy locations and how to reduce the extra communication cost. However, their technique protects the users' location privacy against LBS provider. We are also interested in a user's location privacy against other users.

In the mix zone model proposed by Beresford *et al.* [8], users are assigned different pseudonyms every time he enters the mix zone, and users' paths are hidden by doing so. Several works [9]–[11] are based on this model, but they guarantee the privacy only when the user density is high and user behaviour pattern is unpredictable. Also, most of them require trusted servers.

There are also works related to CR (cloaking region) [12]–[15]. In these works, the LBS receives a cloaking region instead of actual users' locations. Gedit *et al.* proposed spatial cloaking and temporal cloaking in [12]. Each query specifies a temporal interval, and queries within the same interval, whose sources are in the vicinity of the first query's source, are merged to a single query. Otherwise, the query is rejected because it has no anonymity. Kalnis *et al.* [13] used the Hilbert space filling curve to map the two dimensional locations to one dimensional values, which are then indexed by a B+ tree. Then, they partition the one dimensional sorted list into groups of n users, which is the CR of their scheme. Since this Hilbert Cloaking is not based on geometric space, it guarantees privacy for any location distribution. However, a certain range, where the user is located, is disclosed in CR-based approaches, and this is out of users' control. It is more desirable to allow users themselves to configure it.

III. System Model and Problem Formulation

A. System Model

We denote every person engaged in the protocol as a user $\mathcal{U}_i$ (we do not differentiate smartphone users and PC users), the user who publishes his location as a publisher $\mathcal{P}_i$ and the user who queries the location information of other user as a querier $\mathcal{Q}_i$. Note that a user can be a querier and a publisher at the same time. When he queries on others, he acts as a querier and when he is queried, he acts as a publisher. That is, $\mathcal{U}_i = \mathcal{P}_i = \mathcal{Q}_i$ for the same i.

Also, mobile applications or SNS applications which support LBS are denoted as service providers $\mathcal{SP}$. $\mathcal{Q}$ and $\mathcal{P}$ retrieves keys from $\mathcal{SP}$, which are used for access control. For simplicity, we consider only one $\mathcal{SP}$ here.

We assume an independent semi-honest model for users and service providers. That is, they all behave independently and will try to extract useful information from the ciphertexts, but they will follow the protocol in general and will not collude with each other. We further assume that every user communicate with each other via an anonymized network (e.g., Tor: https://www.torproject.org) or other anonymized protocol ([16]) such that the privacy is not compromised by the underlying network protocol. We assume the origin of a packet is successfully hidden, which is out of this paper's scope (otherwise any attacker can achieve the location based on the origin of the packet).

B. Location Assumption

For simplicity, we assume the ground surface is a plane, and every user's location is mapped to an Euclidean space

with integer coordinates (with meter as unit). That is, everyone's location can be expressed as a tuple of coordinates representing a point in a grid partition of the space. This does not affect the generality since there exists a bijection between spherical locations and Euclidean locations. By approximating the coordinates in the Euclidean space to the nearest grid point, we can show that it results in errors of the Euclidean distance between two locations at most $\sqrt{2}$ meters when the space is partitioned using grid of side-length 1 meter.

The Euclidean distance between two users with locations $\mathbf{x}_1 = (x_{11}, x_{12}, x_{13})$ and $\mathbf{x}_2 = (x_{21}, x_{22}, x_{23})$ is $dist(\mathcal{U}_1, \mathcal{U}_2) = |\mathbf{x}_1 - \mathbf{x}_2| = \sqrt{\sum_{i=1}^{3} (x_{1i} - x_{2i})^2}$. Given a real location on the surface of the earth, we need to compute the surface distance, denoted as $SD(\mathcal{U}_i, \mathcal{U}_j)$, between these two points. By assuming that the earth is a sphere with radius R meters, it is easy to show that $SD(\mathcal{U}_i, \mathcal{U}_j) = 2\arcsin(\frac{dist(\mathcal{U}_i, \mathcal{U}_j)}{2R}) \cdot R$. Then the surface distance can be quickly computed from the Euclidean distance. To check if the surface distance satisfies certain conditions, we can convert it to check if the Euclidean distance satisfying corresponding conditions. For example, $dist(\mathcal{U}_1, \mathcal{U}_2) \leq D$ is equivalent as $SD(\mathcal{U}_i, \mathcal{U}_j) \leq 2R\arcsin(D/2R)$. For simplicity and convenience of presentation, in this paper, we will focus on the Euclidean distance instead of the surface distance. Notice that although we consider only Euclidean space here, our protocol works for any system where distance is a polynomial of location points $\mathbf{x}$'s, where $\mathbf{x}$ is a vector.

C. Problem Statement

Each user $\mathcal{U}_i$ has his location information $\mathbf{x}_i = (x_{i1}, x_{i2}, x_{i3})$ which determines his current location. He also has an attribute set S_i which determines his identity (e.g., University:I.I.T, Degree:Ph.D, Major:Computer Science). Then, a querier $\mathcal{Q}_i$ uses his current location information and attribute set to execute a query (function) f on a publisher $\mathcal{P}_j$'s location information $\mathbf{x}_j$. According to $\mathcal{Q}_i$'s location information $\mathbf{x}_i$ and his attribute set S_i, he obtains the corresponding query result $f(\mathbf{x}_i, S_i, \mathbf{x}_j)$. Note that different $\mathbf{x}_i$ and S_i leads to different level of query result. During the whole protocol, $\mathcal{Q}_i$ or $\mathcal{P}_j$ cannot learn any useful extra information about each other's location information.

In this paper, we propose novel protocols such that the location publisher exerts a fine-grained access control on who can access what location information. For example, a publisher could specify the following access control policies: (1) a user can know which city I am in if s/he is in my friend list; or (2) a user can check whether the distance between him and me is less than 100 meters if s/he is my classmate; or (3) a user can compute the exact distance between us if we both went to the same university. We generally assume that a user $\mathcal{U}_i$ has a set of attributes A_i, and that an access control policy of the publisher is specified by a boolean function (specified as an access tree T) on all possible attributes of users.

According to the location information disclosed to the querier, we define four different levels of queries.

Definition 1. Level 1 Query*: When the query ends, $\mathcal{Q}$ learns whether $dist(\mathcal{Q}, \mathcal{P}) \leq \tau$ or not if the attributes of the querier satisfy a certain condition specified by the publisher, where τ is a threshold value determined by $\mathcal{P}$. The querier knows nothing else about the location of the publisher.*

Definition 2. Level 2 Query*: When the query ends, $\mathcal{Q}$ learns whether $dist(\mathcal{Q}, \mathcal{P}) \leq \tau$ when the attributes of the querier satisfy a certain condition specified by the publisher, where τ is a threshold value determined by $\mathcal{Q}$. The querier knows nothing else about the location of the publisher.*

Definition 3. Level 3 Query*: When the query ends, $\mathcal{Q}$ learns the $dist(\mathcal{Q}, \mathcal{P})$ if the attributes of the querier satisfy a certain condition specified by the publisher. The querier knows nothing else about the location of the publisher.*

Definition 4. Level 4 Query*: When the query ends, $\mathcal{Q}$ learns the function $F(\mathbf{x})$ of the location $\mathbf{x}$ of $\mathcal{P}$ if the attributes of the querier satisfy a certain condition specified by the publisher. Here function F is defined by the publisher. The querier knows nothing else about the location of the publisher.*

It is easy to show that the level i query provides better privacy protection than level $i+1$ query, for $i = 1, 2$. Level 4 query provides most information in general. In level 4 query, the function F could be used by the publisher to exert fine-grained access control on his location information. For example $F(\mathbf{x})$ could return the city of the location, the zip-code of the location or the exact location information.

IV. Background

In our Privacy-preserving Location Query Protocol (PLQP), various cryptographic concepts are used. We introduce each of them in this section.

A. Attribute-Based Encryption (ABE)

As Jung *et al.* discussed in detail in their work [17], in the Attribute-Based Encryption (ABE) [18], the identity of a person is viewed as a set of attributes. This enables the encrypter to specify a boolean function to do access control. There are two types of ABE system: Goyal *et al.*'s Key-Policy Attribute-Based Encryption [19] and Bethencourt *et al.*'s Ciphertext-Policy Attribute-Based Encryption [20]. The KP-ABE specifies the encryption policy in the decryption key, and the CP-ABE specifies the policy in the ciphertext. Due to many reasons discussed in [17], we will employ CP-ABE as a component of access control.

1) Access Tree T: In most of previous ABE works (e.g., [19] [20] [21]), encryption policy is described with an access tree. Each non-leaf node of the tree is a threshold gate by a threshold value θ, and each leaf node x is described by an attribute. A leaf node is satisfied if a key contains the corresponding attribute, and a non-leaf threshold gate is satisfied if at least θ children are satisfied.

Note that this threshold-gate based access tree is able to express arbitrary condition, which makes the privacy control in our protocol flexible and scalable.

2) Definition: With the access tree defined as above, the CP-ABE scheme is defined as follows:

Setup $\rightarrow \mathbf{PK}, \mathbf{MK}$. The setup algorithm takes nothing as input other than the implicit security parameter. It outputs the public parameter $\mathbf{PK}$ and a master key $\mathbf{MK}$. The master key belongs to the key issuer and is kept secret.

Encrypt($\mathbf{PK}$, M, T) $\rightarrow \mathcal{E}_T(M)$. The encryption algorithm takes as input the public key $\mathbf{PK}$, a message M, and an access tree T. It will encrypt the message M and returns a ciphertext $\mathbf{CT}$ such that only a user with key satisfying the access tree T can decrypt it.

KeyGenerate($\mathbf{PK}$, $\mathbf{MK}$, S) $\rightarrow \mathbf{SK}$. The Key Generation algorithm takes as input the public key $\mathbf{PK}$, the master key $\mathbf{MK}$ and a set of attributes S. It outputs a private key $\mathbf{SK}$ which contains the attributes in S.

Decrypt($\mathbf{PK}$, $\mathbf{SK}$, $\mathcal{E}_T(M)$) $\rightarrow M$. The decryption algorithm takes as input the public parameter $\mathbf{PK}$, a private key $\mathbf{SK}$ whose attribute set is S, and a ciphertext $\mathbf{CT}$ which contains an access tree T. It outputs the original message M if and only if the set S satisfies the access tree T.

We direct the readers to [20] for detailed construction.

B. Homomorphic Encryption (HE)

Homomorphic Encryption (HE) allows direct addition and multiplication on ciphertexts while preserving decryptability. That is, following equations are satisfied (Note that this is only an example of a HE, and detailed operations vary for different HE system).

$$\text{Enc}(m_1) \cdot \text{Enc}(m_2) = \text{Enc}(m_1 + m_2)$$

$$\text{Enc}(m_1)^{\text{Enc}(m_2)} = \text{Enc}(m_1 \cdot m_2)$$

where $\text{Enc}(m)$ stands for the ciphertext of m.

In general, there are two types of HE: Partially Homomorphic Encryption (PHE) and Fully Homomorphic Encryption (FHE). PHE supports constant number of additions and multiplications, and FHE supports unlimited additions and multiplications but it is much less efficient than PHE. As discussed by Lauter *et al.* in [22], the decryption time of FHE system is too high to be used in a real application, and in most of cases one only needs a few number of multiplications or additions. Therefore, Pallier's system, which is much simpler and thus efficient, is our choice: it involves only one multiplication for each homomorphic addition and one exponentiation for each homomorphic multiplication.

1) Definition of Paillier's Cryptosystem: Paillier's cryptosystem is composed of three algorithms – KeyGenerate, Encrypt and Decrypt.

KeyGenerate $\rightarrow EK, DK$. An entity randomly chooses two large prime numbers p and q of same bit length. He then computes $n = pq$ and $\lambda = (p-1)(q-1)$. Next, he sets $g = (n+1)$ and $\mu = (\lambda \bmod n^2)^{-1} \bmod n$. Then, the encryption key is $EK = (n, g)$ and the decryption key is $DK = (\lambda, \mu)$.

Encrypt$(EK, m) \rightarrow \mathbb{E}(m, r)$. The encrypter selects a random integer $r \in \mathbb{Z}_n$ and computes the ciphertext

$$\mathbb{E}(m, r) = g^m \cdot r^n \bmod n^2$$

and publishes it.

Decrypt$(\mathbb{E}(m, r), DK) \rightarrow m$. The holder of $DK = (\lambda, \mu)$ can decrypt the ciphertext $\mathbb{E}(m, r)$. He computes the following to recover the message:

$$m = L((\mathbb{E}(m, r))^\lambda \bmod n^2) \cdot \mu \bmod n$$

where $L(a) = (a-1)/n \bmod n$.

The Paillier's cryptosystem satisfies the following homomorphic properties:

$$\mathbb{E}(m_1, r_1) \cdot \mathbb{E}(m_2, r_2) = \mathbb{E}(m_1 + m_2, r_1 r_2) \bmod n^2$$

$$\mathbb{E}(m_1, r_1)^{m_2} = \mathbb{E}(m_1 \cdot m_2, r_1^{m_2}) \bmod n^2$$

Note that DK can decrypt only the ciphertexts encrypted with EK which pairs with it. Also, the random number r in a ciphertext $\mathbb{E}(m, r)$ does not contribute to decryption or other homomorphic operation. It only prevents the dictionary attack by randomizing the ciphertext. For sake of simplicity, we use $\mathbb{E}(m)$ instead of $\mathbb{E}(m, r)$ in the remaining paper.

C. Functional Encryption (FE)

Functional Encryption (FE) is a new encryption scheme recently proposed after the Attribute-Based Encryption (ABE). To the best of our knowledge, the concept is first proposed by Boneh *et al.* in [23]. In the open direction of their work, they proposed the terminology 'Functional Encryption' and its general concept, and later in 2011, Boneh *et al.* formally defined it and discussed its challenge [24]. According to their study, the FE is defined as follows: FE is an encryption scheme such that a key holder can learn a specific function of the data based on the ciphertext, but nothing else about the data. This is totally different from the traditional encryption scheme in terms of the differentiated decryption. In traditional encryption schemes (e.g., PKI, ABE), decryption result of a ciphertext for every authorized users is same: the plaintext. In FE, encrypter can specify a function for each key such that each decryption result is the corresponding function of the plaintext.

There are a few recent works related to FE ([25], [26]). However, they mainly focus on hiding encryption policy from ordinary users. To the best of our knowledge, there is no formal construction of FE which satisfies the definition of FE [24].

V. Preliminary Design

In our PLQP, we require that a publisher could specify several access control structures for all potential location queriers. Different access trees will allow access to different level of knowledge about the location information, which is achieved by using FE in our protocol. However, strictly speaking, the encryption in our protocol is not a formal FE because we only support a constant number of functions of the data, so we refer to it as semi-functional encryption. To allow a set of possible queries by all users, we first present

distance computation and comparison algorithms which will be used to provide four levels of functions over location data in our semi-functional PLQP.

1) Privacy Preserving Distance Computation: Let $\mathbf{x} = (x_1, x_2, x_3)$ and $\mathbf{y} = (y_1, y_2, y_3)$ be a publisher $\mathcal{P}$'s and a querier $\mathcal{Q}$'s 3-dimensional location respectively. We use Algorithm 1 to let $\mathcal{Q}$ securely compute $dist(\mathcal{P}, \mathcal{Q})$ without knowing $\mathcal{P}$'s coordinates or disclosing his own one.

Algorithm 1 Privacy Preserving Distance Computation

1: $\mathcal{Q}$ generates a pair of encryption and decryption keys of Paillier's cryptosystem: $EK = (n, g)$, $DK = (\lambda, \mu)$. We assume n is of 1024-bit length. $\mathbb{E}_{\mathcal{Q}}$ denotes the encryption done by $\mathcal{Q}$ using his encryption keys.

2: $\mathcal{Q}$ generates the following ciphertexts and sends them to $\mathcal{P}$ at $\mathbf{x}$.

$$\mathbb{E}_{\mathcal{Q}}(1), \mathbb{E}_{\mathcal{Q}}(\sum_{i=1}^{3} y_i^2), \{\mathbb{E}_{\mathcal{Q}}(y_i) \mid i = 1, 2, 3\},$$

3: $\mathcal{P}$, after receiving the ciphertexts, executes the following homomorphic operations:

$$\begin{cases} \{\mathbb{E}_{\mathcal{Q}}(y_i)^{-2x_i}\} = \{\mathbb{E}_{\mathcal{Q}}(-2x_i y_i)\}, \text{ for } i = 1, 2, 3 \\ \mathbb{E}_{\mathcal{Q}}(1)^{\sum_{i=1}^{3} x_i^2} = \mathbb{E}_{\mathcal{Q}}(\sum_{i=1}^{3} x_i^2) \end{cases}$$

4: $\mathcal{P}$ computes and sends the following to the querier $\mathcal{Q}$:

$$\mathbb{E}_{\mathcal{Q}}(\sum_{i=1}^{3} x_i^2) \cdot \mathbb{E}_{\mathcal{Q}}(\sum_{i=1}^{3} y_i^2) \cdot \prod_{i=1}^{3} (\mathbb{E}_{\mathcal{Q}}(-2x_i y_i))$$
$$= \mathbb{E}_{\mathcal{Q}}(\sum_{i=1}^{3} (x_i - y_i)^2) = \mathbb{E}_{\mathcal{Q}}(|\mathbf{x} - \mathbf{y}|^2)$$

5: $\mathcal{Q}$ uses the private key DK to decrypt the $\mathbb{E}_{\mathcal{Q}}(|\mathbf{x} - \mathbf{y}|^2)$ to get the distance.

Note that the location $\mathbf{y}$ is kept secret to $\mathcal{P}$ during the whole protocol, since he does not know the private key; on the other hand, the location $\mathbf{x}$ is also kept secret since $\mathcal{Q}$ only achieves $\mathbb{E}(|\mathbf{x} - \mathbf{y}|^2)$. However, the location $\mathbf{x}$ is inferred if $\mathcal{Q}$ runs the same protocol at different places for four times in Euclidean space (three times in Euclidean plane). This will be discussed in detail in Theorem VI.1.

2) Privacy Preserving Distance Comparison: Let $\mathbf{x} = (x_1, x_2, x_3)$ and $\mathbf{y} = (y_1, y_2, y_3)$ be publisher $\mathcal{P}$'s and querier $\mathcal{Q}$'s 3-dimensional location respectively. We use Algorithm 2 to let $\mathcal{Q}$ learn whether $dist(\mathcal{P}, \mathcal{Q})$ is less than, equal to or greater than a threshold value τ, which is determined by the publisher $\mathcal{P}$.

The reason δ and δ' are chosen from $\mathbb{Z}_{2^{972}}$ and $\mathbb{Z}_{2^{1022}}$ is because otherwise the comparison is not correct due to the modular operations. This will be further discussed in Section VI-F.

On the other hand, if $\mathcal{Q}$ wants to determine the threshold value τ, he can send another ciphertext $\mathbb{E}(\tau^2)$ at the Step 2. Then, $\mathcal{P}$ computes $\mathbb{E}(\tau^2)^{\delta} \cdot \mathbb{E}(1)^{\delta'} = \mathbb{E}(\delta\tau^2 + \delta')$ at the Step 4 and proceeds same as Algorithm 2.

Algorithm 2 Privacy Preserving Distance Comparison

1: $\mathcal{Q}$ generates encryption and decryption key pair of Paillier's cryptosystem: $EK = (n, g)$, $DK = (\lambda, \mu)$.

2: $\mathcal{Q}$ generates the following ciphertexts and sends them to the user $\mathcal{P}$ with location $\mathbf{x}$.

$$\mathbb{E}_{\mathcal{Q}}(1), \mathbb{E}_{\mathcal{Q}}(\sum_{i=1}^{3} y_i^2), \{\mathbb{E}_{\mathcal{Q}}(-2y_i) \mid i = 1, 2, 3\}$$

3: $\mathcal{P}$, after receiving the ciphertexts, randomly picks two integers $\delta \in \mathbb{Z}_{2^{972}}, \delta' \in \mathbb{Z}_{2^{1022}}$ and executes the following homomorphic operations:

$$\begin{cases} \{\mathbb{E}_{\mathcal{Q}}(-2y_i)^{\delta x_i} = \mathbb{E}_{\mathcal{Q}}(-2\delta x_i y_i) \mid i = 1, 2, 3\} \\ \mathbb{E}_{\mathcal{Q}}(\sum_{i=1}^{3} y_i^2)^{\delta} = \mathbb{E}_{\mathcal{Q}}(\delta(y_1^2 + y_2^2 + y_3^2)) \\ \mathbb{E}_{\mathcal{Q}}(1)^{\delta \sum_{i=1}^{3} x_i^2} = \mathbb{E}_{\mathcal{Q}}(\delta \sum_{i=1}^{3} x_i^2)) \\ \mathbb{E}_{\mathcal{Q}}(1)^{\delta'} = \mathbb{E}_{\mathcal{Q}}(\delta') \\ \mathbb{E}_{\mathcal{Q}}(\delta \sum_{i=1}^{3} x_i^2) \cdot \mathbb{E}_{\mathcal{Q}}(\delta') = \mathbb{E}_{\mathcal{Q}}(\delta \sum_{i=1}^{3} x_i^2 + \delta') \end{cases}$$

4: $\mathcal{P}$ computes the followings and sends them back to the other user at $\mathbf{y}$.

$$\mathbb{E}_{\mathcal{Q}}(\delta \sum_{i=1}^{3} x_i^2 + \delta') \cdot \mathbb{E}_{\mathcal{Q}}(\delta \sum_{i=1}^{3} y_i^2) \prod_{i=1}^{3} (\mathbb{E}_{\mathcal{Q}}(-2\delta x_i y_i))$$
$$= \mathbb{E}_{\mathcal{Q}}((\delta \sum_{i=1}^{3} (x_i - y_i)^2) + \delta') = \mathbb{E}_{\mathcal{Q}}(\delta |\mathbf{x} - \mathbf{y}|^2 + \delta')$$
$$\mathbb{E}_{\mathcal{Q}}(1)^{\delta\tau^2 + \delta'} = \mathbb{E}_{\mathcal{Q}}(\delta\tau^2 + \delta')$$

5: $\mathcal{Q}$ uses the private key $DK(\lambda, \mu)$ to decrypt the ciphertexts and gets $\delta|\mathbf{x} - \mathbf{y}|^2 + \delta'$ and $\delta\tau^2 + \delta'$. If, without modular operations, both of them are less than the modulo n, we have:

$$\delta|\mathbf{x} - \mathbf{y}|^2 + \delta' < \delta\tau^2 + \delta' \quad \Leftrightarrow \quad |\mathbf{x} - \mathbf{y}| < \tau$$

VI. Privacy Preserving Location Services

In this section, we propose the construction of Privacy-preserving Location Query Protocol (PLQP). First of all, we define a group for CP-ABE.

Let $\mathbb{G}_0$ be a multiplicative cyclic group of prime order m and g be its generator. The bilinear map e used in CP-ABE is defined as follows: $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$, where $\mathbb{G}_T$ is the codomain of the map e. The bilinear map e has the following properties:

1) **Bilinearity**: for all $u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_q$, $e(u^a, v^b) = e(u, v)^{ab}$
2) **Symmetry**: for all $u, v \in \mathbb{G}_0$, $e(u, v) = e(v, u)$
3) **Non-degeneracy**: $e(g, g) \neq 1$

Definition 5. *The Decisional Diffie-Hellman (DDH) problem in an integer group with generator g is defined as follows: on input $g, g^a, g^b, g^c = g^{ab} \in \mathbb{Z}$, where $a, b, c \in \mathbb{Z}$, decide whether $c = ab$ or c is a random element.*

Definition 6. *The Decisional Bilinear Diffie-Hellman (DBDH) problem in group $\mathbb{G}_0$ of prime order p with generator g is defined as follows: on input $g, g^a, g^b, g^c \in \mathbb{G}_0$ and $e(g, g)^z =$*

$e(g,g)^{abc} \in \mathbb{G}_T$, *where* $a, b, c \in \mathbb{Z}_q$, *decide whether* $z = abc$ *or* z *is a random element.*

The security of our construction relies on the assumption that no probabilistic polynomial-time algorithms can solve the DDH problem or DBDH problem with non-negligible advantage. This is a widely made assumption in various cryptographic works ([17], [26], [27]), which is reasonable since discrete logarithm problems in large number fields are widely considered to be intractable ([17], [28], [29]).

The reason we introduce CP-ABE here is to exert fine-grained access control over the location queries. Even if one's location satisfies a certain condition, one cannot gain any information from the query if his identity attributes do not satisfy the pre-defined encryption policy.

A. Initialize

The service provider $\mathcal{SP}$ initializes the system by following the instructions:

Algorithm 3 Initialize

1: Executes Setup (CP-ABE) to generate public and master key pairs:

$$\begin{cases} \mathbf{PK} = \langle \mathbb{G}_0, g, h = g^{\beta}, f = g^{1/\beta}, e(g,g)^{\alpha} \rangle \\ \mathbf{MK} \langle \beta, g^{\alpha} \rangle \end{cases}$$

2: Executes KeyGenerate (CP-ABE) for all users within the system to issue them private keys corresponding to their attributes.

$$\mathbf{SK} = \langle D = g^{(\alpha+r)/\beta}, \forall j \in S: \ D_j = g^r H(j)^{r_j}, D'_j = g^{r_j} \rangle$$

Here we assume secure channels exist between users and service providers $\mathcal{SP}$ such that private keys are securely delivered to each user.

B. Protocol for Level 4 Query

After the query ends, $\mathcal{Q}_j$ learns $\mathcal{P}_i$'s exact location $\mathbf{x}_i$.

Algorithm 4 Level 4 Query Protocol

1: A publisher $\mathcal{P}_i$ creates an access tree T_{i4} which specifies the access authority for the level 4 query.
2: When a querier $\mathcal{Q}_j$ sends a level 4 query to $\mathcal{P}_i$, $\mathcal{P}_i$ encrypts his location using the CP-ABE algorithm Encrypt:

$$\mathcal{E}_{T_{i4}}(x_{i,1}), \mathcal{E}_{T_{i4}}(x_{i,2}), \mathcal{E}_{T_{i4}}(x_{i,3})$$

3: These are sent to $\mathcal{Q}_j$, and $\mathcal{Q}_j$ decrypts it with his private key $\mathbf{SK}$ if it satisfies the access tree T_{i4}, and achieves $\mathcal{P}_i$'s locoation.

C. Protocol for Level 3 Query

After the query ends, $\mathcal{Q}_j$ learns the $dist(\mathcal{Q}_j, \mathcal{P}_i)$.

Algorithm 5 Level 3 Query Protocol

1: A publisher $\mathcal{P}_i$ creates an access tree T_{i3} which specifies the access authority for the level 3 query.
2: When a querier $\mathcal{Q}_j$ wants to send a level 3 query to $\mathcal{P}_i$, he initiates the Secure Distance Computation protocol (Section V-1) by generating encryption and decryption Paillier key pair $EK_j = (n_j, g_j), DK_j = (\lambda_j, \mu_j)$.
3: Then, he calculates the following ciphertexts and sends to $\mathcal{P}_i$:

$$\mathbb{E}(1), \mathbb{E}(x_{j1}^2 + x_{j2}^2 + x_{j3}^2), \{\mathbb{E}(-2x_{ji})\}_{i=1,2,3}$$

4: $\mathcal{P}_i$, after receiving them, calculates the ciphertext below:

$$\mathbb{E}(|\mathbf{x}_i - \mathbf{x}_j|^2)$$

5: The ciphertext above is encrypted again with the access tree T_{i3} using the CP-ABE algorithm Encrypt, which we refer to doubly nested ciphertexts:

$$\mathcal{E}_{T_{i3}}(\mathbb{E}(|\mathbf{x}_1 - \mathbf{x}_2|^2)$$

6: The doubly nested ciphertext is sent back to $\mathcal{Q}_j$, and if $\mathcal{Q}_j$'s private key $\mathbf{SK}$ satisfies the access tree T_{i3}, he can decrypt it and use his Paillier key pair to decrypt the ciphertext again to achieve $|\mathbf{x}_1 - \mathbf{x}_2|^2$. Then, he obtains the $dist(\mathcal{Q}_j, \mathcal{P}_i)$.

Theorem VI.1. *If* $\mathcal{Q}$ *executes the level 3 query for more than three times at different places in Euclidean space, level 3 query is equivalent to level 4 query.*

Proof: This is also mentioned in the Section V-2. If $\mathcal{Q}$ executes the level 3 query for four times at different locations, he achieves 4 distances: $\{|\mathbf{x}_i - \mathbf{y}|\}_{i=1,2,3,4}$, where $\mathbf{x}_i$'s are $\mathcal{Q}$'s 4 different locations and $\mathbf{y}$ is $\mathcal{P}$'s location. These are essentially four equations with three variables y_1, y_2 and y_3:

$$(x_{i1}-y_1)^2+(x_{i2}-y_2)^2+(x_{i3}-y_3)^2 = |\mathbf{x}_i - \mathbf{y}|^2 \ \ (i = 1,2,3,4)$$

which can be solved. Therefore, $\mathcal{P}$'s location $\mathbf{y}$ can be computed in this case. ■

Similarly, it can be proved that if $\mathcal{Q}$ executes the level 3 query for more than two times at different places in Euclidean plane, level 3 query is equivalent to level 4 query.

D. Protocol for Level 2 Query

After the query ends, $\mathcal{Q}_j$ learns whether $dist(\mathcal{Q}_j, \mathcal{P})$ is less than, equal to or greather than τ, where τ is a threshold value determined by $\mathcal{Q}_j$.

Theorem VI.2. *Suppose* D *is the greatest possible distance in the location space, if* $\mathcal{Q}$ *executes the level 2 query for* $\Theta(\log D)$ *times, level 2 query is equivalent to level 3 query.*

Proof: Since $\mathcal{Q}$ can control the threshold value τ, he can first execute a level 2 query with $\tau = D$. Then, he uses binary search to execute level 2 queries with different τ's until he finds the τ such that $\tau = |\mathbf{x} - \mathbf{y}|$, where $\mathbf{x}$ and $\mathbf{y}$ are $\mathcal{Q}$'s and $\mathcal{P}$'s locations respectively. Then, he finds the distance. ■

Algorithm 6 Level 2 Query Protocol

1: A publisher $\mathcal{P}_i$ creates an access tree T_{i2} which specifies the access authority for the level 2 query.
2: When a querier $\mathcal{Q}_j$ wants to send a level 2 query to $\mathcal{P}_i$, he initiates the Secure Distance Comparison protocol (Section V-2) by picking two large prime numbers p_j, q_j of the same length, then $n_j = p_j q_j$, $g_j = n_j + 1$, $\lambda_j = (p_j - 1)(q_j - 1)$ and $\mu_j = \lambda_j^{-1} \bmod n_j$, which form Paillier key pair $EK_j = (n_j, g_j), DK_j = (\lambda_j, \mu_j)$ (The subscriptions indicate that these keys are used by $\mathcal{Q}_j$).
3: Then, he calculates the following ciphertexts and sends them to $\mathcal{P}_i$:

$$\mathbb{E}(1), \mathbb{E}(x_{j1}^2 + x_{j2}^2 + x_{j3}^2), \{\mathbb{E}(-2x_{ji})\}_{i=1,2,3}, \mathbb{E}(\tau^2)$$

4: $\mathcal{P}_i$, after receiving them, picks two random integers $\delta \in \mathbb{Z}_{2^{970}}, \delta' \in \mathbb{Z}_{2^{1022}}$ and calculates the ciphertexts:

$$\begin{cases} \mathbb{E}(\sum_{k=1}^{3} x_{j,k}^2)^{\delta} \cdot \mathbb{E}(1)^{\delta \sum_{k=1}^{3} x_{i,k}^2 + \delta'} \cdot \prod_{k=1}^{3} \mathbb{E}(-2x_{jk})^{\delta x_{ik}} \\ \mathbb{E}(\tau^2)^{\delta} \cdot \mathbb{E}(1)^{\delta'} = \mathbb{E}(\delta\tau^2 + \delta') \end{cases}$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ refer to $\mathcal{P}_i$'s and $\mathcal{Q}_j$'s locations respectively.
5: These ciphertexts are encrypted again with the access tree T_{i1} using the CP-ABE algorithm Encrypt:

$$\mathcal{E}_{T_{i1}}(\mathbb{E}(\delta|\mathbf{x}_i - \mathbf{x}_j|^2 + \delta')), \mathcal{E}_{T_{i1}}(\mathbb{E}(\delta\tau^2 + \delta'))$$

6: The doubly nested ciphertexts are sent back to $\mathcal{Q}_j$, and if $\mathcal{Q}_j$'s private key $\mathbf{SK}$ satisfies the access tree T_{i1}, he can decrypt them and uses his Paillier key pair to decrypt the ciphertext again to achieve $\delta|\mathbf{x}_i - \mathbf{x}_j|^2 + \delta'$ and $\delta\tau^2 + \delta'$. Then he is able to compare two values to learn whether $dist(\mathcal{P}_i, \mathcal{Q}_i)$ is less than, equal to or greater than τ.

Algorithm 7 Level 1 Query Protocol

1: A publisher $\mathcal{P}_i$ creates an access tree T_{i1} which specifies the access authority for the level 1 query.
2: When a querier $\mathcal{Q}_j$ wants to send a level 1 query to $\mathcal{P}_i$, he initiates the Secure Distance Comparison protocol (Section V-2) by picking two large prime numbers p_j, q_j of the same length, then $n_j = p_j q_j$, $g_j = n_j + 1$, $\lambda_j = (p_j - 1)(q_j - 1)$ and $\mu_j = \lambda_j^{-1} \bmod n_j$, which form Paillier key pair $EK_j = (n_j, g_j), DK_j = (\lambda_j, \mu_j)$ (The subscriptions indicate that these keys are used by $\mathcal{Q}_j$).
3: Then, he calculates the following ciphertexts and sends them to $\mathcal{P}_i$:

$$\mathbb{E}(1), \mathbb{E}(x_{j1}^2 + x_{j2}^2 + x_{j3}^2), \{\mathbb{E}(-2x_{ji})\}_{i=1,2,3}$$

4: $\mathcal{P}_i$, after receiving them, picks two random integers $\delta \in \mathbb{Z}_{2^{970}}, \delta' \in \mathbb{Z}_{2^{1022}}$ and calculates the ciphertexts:

$$\mathbb{E}(\delta|\mathbf{x}_i - \mathbf{x}_j|^2 + \delta'), \mathbb{E}(\delta\tau^2 + \delta')$$

where $\mathbf{x}_i$ and $\mathbf{x}_j$ refer to $\mathcal{P}_i$'s and $\mathbf{x}_j$'s locations respectively.
5: These ciphertexts are encrypted again with the access tree T_{i1} using the CP-ABE algorithm Encrypt:

$$\mathcal{E}_{T_{i1}}(\mathbb{E}(\delta|\mathbf{x}_i - \mathbf{x}_j|^2 + \delta')), \mathcal{E}_{T_{i1}}(\mathbb{E}(\delta\tau^2 + \delta'))$$

6: The doubly nested ciphertexts are sent back to $\mathcal{Q}_j$, and if $\mathcal{Q}_j$'s private key $\mathbf{SK}$ satisfies the access tree T_{i1}, he can decrypt them and use his Paillier key pair to decrypt the ciphertext again to achieve $\delta|\mathbf{x}_i - \mathbf{x}_j|^2 + \delta'$ and $\delta\tau^2 + \delta'$. Then he is able to compare two values to learn whether $dist(\mathcal{P}_i, \mathcal{Q}_i)$ is less than, equal to or greater than τ.

E. Protocol for Level 1 Query

After the query ends, $\mathcal{Q}_j$ learns whether $dist(\mathcal{Q}_j, \mathcal{P})$ is less than, equal to or greather than τ, where τ is a threshold value determined by $\mathcal{P}_i$.

Theorem VI.3. *If $\mathcal{Q}$'s distance to $\mathcal{P}$ is less than τ, 1evel 1 query is equivalent to level 4 query after $O(\log \tau)$ tries.*

Proof: For sake of visualization, we prove the theorem in Euclidean plane, but the proof also holds in Euclidean space.

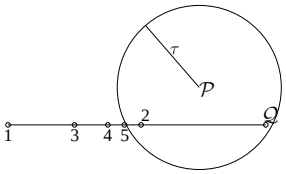

Fig. 1. $\hat{\mathbf{x}}$ being inferred by binary search

First draw a circle whose center is $\mathcal{P}$'s location and the radius is τ. Then, if $\mathcal{Q}$ is inside this circle, his level 1 query result is '$<$'; if he is outside the circle, the result is '$>$'; if he is just on the circle, the result is '$=$'.

$\mathcal{Q}$ executes level 1 queries at another random place $\mathbf{x}'$ which is 2τ apart from his current location $\mathbf{x}$ (i.e., $|\mathbf{x} - \mathbf{x}'| = 2\tau$). Since the radius is τ, $\mathbf{x}'$ must be outside the circle. Then, he uses binary search on the line $(\mathbf{x}', \mathbf{x})$ to find the point $\hat{\mathbf{x}}$ such that $|\hat{\mathbf{x}} - \mathbf{y}| = \tau$ (i.e., the intersection point with the circle). Figure 1 illustrates this process, where point with number i represents the location where the i-th query is executed, and the point $\mathcal{Q}$ is his initial location.

The querier repeat the above process by randomly selecting two more different points $\mathbf{x}'$. We then found three points on the circle. Consequently the location $\mathbf{y}$ is successfully found. The querier needs at most $\log_2(2\tau)$ tries to find a point on the circle, and three such points are needed to locate $\mathbf{y}$, so $\mathbf{y}$ can be calculated after at most $3\log_2(2\tau)$ times for level 1 query. ■

Theorem VI.4. *Suppose D is the greatest possible distance in the location space, if $\mathcal{Q}$'s distance to $\mathcal{P}$ is greater than τ, the expected number of level 1 queries after which $\mathcal{Q}$ achieves $\mathcal{P}$'s location is $\Omega((D/\tau)^d \log D)$, where d is 2 for Euclidean plane and 3 for Euclidean space.*

For the simplicity, we only prove for Euclidean plane, but same proof also holds for Euclidean space.

Proof: $\mathcal{Q}$ is outside the circle (the one drawn above), so if he finds another location inside the circle, he can determine the location of $\mathcal{P}$ as proved. Since $\mathcal{Q}$ does not know where is the circle, he can only randomly choose any location in the location space to execute the level 1 query. The probability of first guess being inside the circle is approximately

$$\text{(Size of circle / Size of Euclidean plane)} \approx (\pi\tau^2)/(XY-1)$$

where X is the number coordinates in x-axis and Y is the number of coordinates in y-axis in the Euclidean plane. The approximation comes from the reason that our location system is discrete system with integer coordinates, and $\mathcal{Q}$'s current location will not be chosen. We can further deduce that at each time, the probability of i-th guess being inside the circle is approximately $\frac{(\pi\tau^2)}{XY-i}$. Therefore, the probability that the point inside the circle will be found at k-th try is approximately $(1-\frac{\pi\tau^2}{XY-k})^{k-1}\cdot\frac{\pi\tau^2}{XY-k}$. Then, the expected number of tries until the first success is approximately $\sum_{i=1}^{\infty} k(1-\frac{\pi\tau^2}{XY-k})^{k-1}\cdot\frac{\pi\tau^2}{XY-k}$. Then, we have

$$\begin{aligned}
&\sum_{i=1}^{\infty} k(1-\frac{\pi\tau^2}{XY-k})^{k-1}\cdot\frac{\pi\tau^2}{XY-k}\\
&> \sum_{i=1}^{\infty} k(1-\frac{\pi\tau^2}{XY})^{k-1}\cdot\frac{\pi\tau^2}{XY}\\
&= \frac{XY}{\pi\tau^2} = \Theta((D/\tau)^2)
\end{aligned}$$

Therefore, expected number of level 1 queries after which a point inside the circle is guessed is $\Omega((D/\tau)^2)$. After this point is found, the point on the circle can be found using binary search, which leads to $\Theta(logD)$. With three this kind of points, $\mathcal{P}$'s location can be calculated. Therefore, total expected number of level 1 queries needed to correctly locate $\mathcal{P}$'s location is $\Omega((D/\tau)^d \log D)$.

Similarly, it can be proved that the expected total number in Euclidean space is $\Omega((D/\tau)^3 \log D)$. ■

So far, 4 different levels of query protocols are constructed. However, note that level 1-3 queries are equivalent to the level 4 query unless some restrictions are applied, which is proved above. Hence, some restrictions should be applied to protect user's location privacy.

According to Theorem VI.1, during the time period when $\mathcal{P}$'s location does not change, level 3 query is equivalent to level 4 query unless level 3 queries are limited to three times (two times in Euclidean plane) in this period. Thus, the $\mathcal{P}$ can choose to discard the query requests after three times of queries.

According to Theorem VI.2, in the level 2 query, information is leaked when one query returns that distance is greater than τ and another one returns that the distance is less than τ. So, $\mathcal{P}$ can choose to discard the query requests when the comparison result changes (e.g., from $|\mathbf{x}-\mathbf{y}|<\tau$ to $|\mathbf{x}'-\mathbf{y}|>\tau$). Although not responding also leaks some information, this let $\mathcal{Q}$ learn only that the distance is between two pre-calculated two values.

Similar actions can be taken by $\mathcal{P}$ in the level 1 query. He responds to queries until the comparison result changes, and not responding to queries let $\mathcal{Q}$ learn only that the point on the circle is somewhere between two points, and thus protecting $\mathcal{P}$'s location.

F. Restrictions for δ, δ'

As mentioned in Section V-2, $\delta|\mathbf{x}-\mathbf{y}|^2+\delta'$ and $\delta\tau^2+\delta'$ should be less than the modulo n, where n is one of the parameters in Paillier's cryptosystem (Section IV-B1). Otherwise, due to the modular operations, the two parameters cannot be compared.

Normally $n=pq$ is a 1024-bit number, which indicates $n >= 2^{1023}$. In Euclidean plane, the greatest possible distance in a map of the world is $\sqrt{2}C$, where C is the circumference of the earth (approximately 40000km). This value is approximately equal to $6\cdot 10^7 \approx 2^{26}$. Therefore, $|\mathbf{x}-\mathbf{y}|^2 \leq 2^{52}$, so it is sufficient to let $\delta \in \mathbb{Z}_{2^{970}}$ and $\delta' \in \mathbb{Z}_{2^{1022}}$. Then, $\delta|\mathbf{x}-\mathbf{y}|^2+\delta' < 2^{1023} < n$. In Euclidean space, the greatest possible distance is the above distance in a map of the world plus atmosphere height (vector addition). This value is approximately equal to the largest distance above (We estimate the atmosphere height as 32km since 99% of the air is within it, which is too small when compared with the circumference of the earth). Therefore, the restrictions to δ, δ' remain same.

VII. Performance Evaluation

In this section, we evaluate the extra communication and computation overhead introduced in our Privacy-preserving Location Query Protocol (PLQP).

Large Number Arithmetic library for smartphone is unavailable currently, so we implemented our protocol in a computer with only one CPU underclocked to 900MHz, whose computation ability is similar to a smartphone. We used GMP library and CP-ABE toolkit [20] to implement the protocol in Ubuntu 11.04.

Every parameter's length is same as the construction, and we randomly picked two locations for a querier $\mathcal{Q}$ and a publisher $\mathcal{P}$. Then, we executed each level query for 1000 times and measured the average running time for each. Since the purpose of the evaluation is to evaluate the computation performance, so we issued a decryption key (of CP-ABE) containing all attributes, which satisfies any access tree, to the querier. In addition, it is well studied in previous works ([17], [20], [21]) that encryption and decryption time is proportional to the number of attributes (leaf nodes) in the access tree, so we fixed the attributes in each access tree to ten in every query and did not further analyzed its impact on run time.

Table I shows the average run time of each query at the querier's and the publisher's side. We found the run time is dominated by the encryption and decryption algorithms of CP-ABE, and the total run time of each query is less than 1.5 seconds. Also, Table II shows that the communication overhead is less than 10 Kilobytes. Note that we only listed

TABLE I
COMPUTATION OVERHEAD

Query Level	$\mathcal{Q}$'s Run Time (ms)	$\mathcal{P}$'s Run Time (ms)
1	577.49	919.24
2	588.02	909.53
3	492.89	704.85
4	413.05	702.71

TABLE II
COMMUNICATION OVERHEAD

Query Level	$\mathcal{Q} \rightarrow \mathcal{P}$ (Bytes)	$\mathcal{P} \rightarrow \mathcal{Q}$ (Bytes)
1	1280	6592
2	1536	6592
3	1280	3296
4	0	3052

the extra overhead in the tables. The total overhead should include other regular overhead (control messages, ACKs etc.). In conclusion, the computation and communication overhead of our protocol is low enough to be used in a real mobile network.

VIII. CONCLUSION

In this paper, we proposed a fine-grained Privacy-preserving Location Query Protocol (PLQP), which successfully solves the privacy issues in existing LBS applications and provides various location based queries. The PLQP uses our novel distance computation and comparison protocol to implement semi-functional encryption, which supports multi-levelled access control, and used CP-ABE as subsidiary encryption scheme to make access control be more fine-grained. Also, during the whole protocol, unless intended by the location publisher, the location information is kept secret to anyone else. We also conducted experiment evaluation to show that the performance of our protocol is applicable in a real mobile network.

REFERENCES

[1] T. Hashem and L. Kulik, "Safeguarding location privacy in wireless ad-hoc networks," *Ubicomp 2007: Ubiquitous Computing*, pp. 372–390, 2007.

[2] C. Bettini, X. Wang, and S. Jajodia, "Protecting privacy against location-based personal identification," *Secure Data Management*, pp. 185–199, 2005.

[3] M. Mokbel, C. Chow, and W. Aref, "The new casper: query processing for location services without compromising privacy," in *Proceedings of the 32nd international conference on Very large data bases*, VLDB Endowment, 2006, pp. 763–774.

[4] K. Vu, R. Zheng, and J. Gao, "Efficient algorithms for k-anonymous location privacy in participatory sensing." in *IEEE INFOCOM*, 2012.

[5] L. Sweeney *et al.*, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty Fuzziness and Knowledge Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[6] H. Zang and J. Bolot, "Anonymization of location data does not work: A large-scale measurement study," in *Proceedings of the 17th annual international conference on Mobile computing and networking*, 2011, pp. 145–156.

[7] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *21st International Conference on Data Engineering Workshops, 2005*, pp. 1248–1248.

[8] A. Beresford and F. Stajano, "Mix zones: User privacy in location-aware services," in *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004*, pp. 127–131.

[9] B. Hoh, M. Gruteser, R. Herring, J. Ban, D. Work, J. Herrera, A. Bayen, M. Annavaram, and Q. Jacobson, "Virtual trip lines for distributed privacy-preserving traffic monitoring," in *Proceeding of the 6th international conference on Mobile systems, applications, and services*, ACM, 2008, pp. 15–28.

[10] M. Li, K. Sampigethaya, L. Huang, and R. Poovendran, "Swing & swap: user-centric approaches towards maximizing location privacy," in *Proceedings of the 5th ACM workshop on Privacy in electronic society*, 2006, pp. 19–28.

[11] X. Liu, H. Zhao, M. Pan, H. Yue, X. Li, and Y. Fang, "Traffic-aware multiple mix zone placement for protecting location privacy." in *IEEE INFOCOM 2012*.

[12] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," in *Proceedings of 25th IEEE International Conference on Distributed Computing Systems, 2005*, pp. 620–629.

[13] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, "Preventing location-based identity inference in anonymous spatial queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 12, pp. 1719–1733, 2007.

[14] Y. Wang, D. Xu, X. He, C. Zhang, F. Li, and B. Xu, "L2p2: Location-aware location privacy protection for location-based services." in *IEEE INFOCOM 2012*.

[15] C. Chow, M. Mokbel, and X. Liu, "A peer-to-peer spatial cloaking algorithm for anonymous location-based service," in *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*, 2006, pp. 171–178.

[16] Y. Liu, J. Han, and J. Wang, "Rumor riding: anonymizing unstructured peer-to-peer systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 3, pp. 464–475, 2011.

[17] T. Jung, X. Li, Z. Wan, and M. Wan, "Privacy preserving cloud data access with multi-authorities," in *IEEE INFOCOM*, 2013.

[18] A. Sahai and B. Waters, "Fuzzy identity-based encryption," *Advances in Cryptology–EUROCRYPT 2005*, pp. 557–557, 2005.

[19] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 89–98.

[20] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE Symposium on Security and Privacy, 2007*, pp. 321–334.

[21] J. Liu, Z. Wan, and M. Gu, "Hierarchical attribute-set based encryption for scalable, flexible and fine-grained access control in cloud computing," *Information Security Practice and Experience*, pp. 98–107, 2011.

[22] K. Lauter, M. Naehrig, and V. Vaikuntanathan, "Can homomorphic encryption be practical," *Preprint*, 2011.

[23] D. Boneh, P. Papakonstantinou, C. Rackoff, Y. Vahlis, and B. Waters, "On the impossibility of basing identity based encryption on trapdoor permutations," in *IEEE 49th Annual IEEE Symposium on Foundations of Computer Science*, 2008, pp. 283–292.

[24] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," *Theory of Cryptography*, pp. 253–273, 2011.

[25] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," *Advances in Cryptology–EUROCRYPT 2010*, pp. 62–91, 2010.

[26] A. Sahai and H. Seyalioglu, "Worry-free encryption: functional encryption with public keys," in *Proceedings of the 17th ACM conference on Computer and communications security*, 2010, pp. 463–472.

[27] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology–EURYPTO 2001*, pp. 213–229.

[28] T. Jung, X. Mao, X. Li, S. Tang, W. Gong, and L. Zhang, "Privacy-preserving data aggregation without secure channel: multivariate polynomial evaluation," in *IEEE INFOCOM*, 2013.

[29] L. Zhang, X. Li, Y. Liu, and T. Jung, "Verifiable private multi-party computation: ranging and ranking," in *IEEE INFOCOM Mini-Conference*, 2013.