Practical k Nearest Neighbor Queries with Location Privacy

Xun Yi*, Russell Paulet*, Elisa Bertino†, Vijay Varadharajan‡
*College of Engineering and Science, Victoria University, Melbourne, VIC 8001, Australia
†Department of Computer Sciences, Purdue University, West Lafayette, IN 47907, USA
‡Department of Computing, Faculty of Science, Macquarie University, NSW 2109 Australia

***Abstract*—In mobile communication, spatial queries pose a serious threat to user location privacy because the location of a query may reveal sensitive information about the mobile user. In this paper, we study k nearest neighbor (kNN) queries where the mobile user queries the location-based service (LBS) provider about k nearest points of interest (POIs) on the basis of his current location. We propose a solution for the mobile user to preserve his location privacy in kNN queries. The proposed solution is built on the Paillier public-key cryptosystem and can provide both location privacy and data privacy. In particular, our solution allows the mobile user to retrieve one type of POIs, for example, k nearest car parks, without revealing to the LBS provider what type of points is retrieved. For a cloaking region with $n \times n$ cells and m types of points, the total communication complexity for the mobile user to retrieve a type of k nearest POIs is $O(n+m)$ while the computation complexities of the mobile user and the LBS provider are $O(n+m)$ and $O(n^2m)$, respectively. Compared with existing solutions for kNN queries with location privacy, our solutions are more efficient. Experiments have shown that our solutions are practical for kNN queries.**

I. Introduction

The embedding of positioning capabilities (e.g., GPS) in mobile devices facilitates the emergence of location-based services (LBS), which is considered as the next "killer application" in the wireless data market. LBS allows clients to query a service provider (such as Google or Bing Maps) in a ubiquitous manner, in order to retrieve detailed information about points of interest (POIs) in their vicinity (e.g., restaurants, hospitals, etc.).

The LBS provider processes spatial queries on the basis of the location of the mobile user. Location information collected from mobile users, knowingly and unknowingly, can reveal far more than just a user's latitude and longitude. Knowing where a mobile user is can mean knowing what he/she is doing: attending a religious service or a support meeting, visiting a doctor's office, shopping for an engagement ring, carrying out non-work related activities in office, or spending an evening at the corner bar. It might reveal that he is interviewing for a new job or "out" him as a participant at a gun rally or a peace protest. It can mean knowing with whom he/she spends time, and how often. When location data are aggregated it can reveal his/her regular habits and routines - and when he deviates from them.

A 2010 survey conducted for Microsoft in the United Kingdom, Germany, Japan, the United States, and Canada found that 94 percent of consumers who had used location-based services considered them valuable, but the same survey found that 52 percent were concerned about potential loss of privacy[1].

In this paper, we study k nearest neighbor (kNN) queries where the mobile user queries the LBS provider about k nearest points of interest (POIs). In general, the mobile user needs to submit his location to the LBS provider which then finds out and returns to the user the k nearest POIs by comparing the distances between the mobile user's location and POIs nearby. This reveals the mobile user's location to the LBS provider.

There have been numerous techniques that can provide a certain degree of location privacy. These techniques mainly include

- Information access control [12], [28];
- Mix zone [2];
- k-anonymity [11], [1], [5]
- "Dummy" locations [10], [27], [21];
- Geographic data transformation [9], [25], [8], [26];
- Private Information Retrieval (PIR) [6], [7], [18], [19], [17].

LBS queries based on access control, mix zone and k-anonymity require the service provider or the middleware that maintains all user locations. They are vulnerable to misbehavior of the third party. They offer little protection when the service provider/middleware is owned by an untrusted party. There have been private data inadvertently disclosed over the Internet in the past.

k-anonymity is initially used for identity privacy protection. It is generally inadequate for location privacy protections, where the notion of distance between locations is important (unlike distances between identities). The effect of LBS queries based on k-anonymity depends heavily on the distribution and density of the mobile users, which, however, are beyond the control of the location privacy technique.

LBS queries based on dummy locations require the mobile user randomly to choose a set of fake locations, to send the fake locations to the LBS and to receive the false reports from the LBS over the mobile network. This incurs both computation and communication overhead in mobile devices. For the purpose of efficiency, the mobile user may choose less

[1] http://www.microsoft.com/security/resources/research.aspx#LBS

fake locations, but the LBS provider can restrict the user in a small sub space of the total domain, leading to weak privacy.

LBS queries based geographic data transformation are prone to access pattern attacks [24] because the same query always returns the same encoded results. For example, the LBS may observe the frequencies of the returned ciphertexts. Having knowledge about the context of the database, it can match the most popular plaintext POI with the most frequently returned ciphertext and, thus, unravel information about the query.

LBS queries based on PIR provide strong cryptographic guarantees, but are often computationally and communicationally expensive. To improve efficiency, trusted hardware was employed to perform PIR for LBS queries [17]. This technique is built on hardware-aided PIR [23], which assume that a trusted third party (TTP) initializes the system by setting the secret key and the permutation of the database. Like LBS queries based on access control, mix zone and k-anonymity, this technique is vulnerable to misbehavior of the third party.

It is a challenge to give practical solutions for kNN queries with location privacy on the basis of PIR.

In this paper, we construct solutions for kNN queries on the basis of PIR with the Paillier public-key cryptosystem [15]. We have three main contributions as follows:

- Current PIR-based LBS queries [6], [7], [18], [19] usually require two stages. In the first stage, the mobile user retrieves the index of his location from the LBS provider. In the second stage, the mobile user retrieves the POIs according to the index from the LBS provider. To simplify the process, we give a solution for kNN queries which needs one stage only, i.e., the mobile user sends his location (encrypted) to the LBS provider and receives the k nearest POIs (encrypted) from the LBS provider.
- Current PIR-based LBS queries only allow the mobile user to find out k nearest POIs regardless of the type of POIs. For the first time, we take into account the type of POIs in kNN queries and give a solution for the mobile user to find out k nearest PIOs of the same type without revealing to LBS provider what type of POIs he is interested in. For example, our solution allows the mobile user to find out k nearest car parks from the LBS provider.
- Current PIR-based LBS queries all need to fix a cloaking region based on which the LBS provider generates the responses to the mobile user's queries. If the cloaking region is large, the LBS queries are inefficient. If the cloaking region is small, the LBS queries have weak privacy. We give a solution for the mobile user to specify a large public cloaking region but let the LBS provider generate the responses actually based on a small private cloaking region repeatedly.

To analyze the security of our solutions, we define a security model for private kNN queries. The security analysis has shown that our solutions ensures both location privacy in the sense that the user does not reveal any information about his location to the LBS provider and data privacy in the sense that the LBS provider releases to the user only k nearest POIs per query.

For a cloaking region with $n \times n$ cells and m types of points, assume that the mobile user wishes to retrieve a type of k nearest POIs at his location, the total communication complexity is $O(n + m)$ while the computation complexities of the mobile user and the LBS provider are $O(n + m)$ and $O(n^2m)$, respectively. Compared with existing solutions for kNN queries with location privacy, our solution is more efficient.

We have implemented our solution on an example of location-based database and experiments have shown that our solutions are practical.

The rest of the paper is arranged as follows. Related works are surveyed in Section II. We define our model and described our solutions in Section III. The security and performance analysis is carried out in Section IV. Experiment results are shown in Section V. Conclusions are drawn in the last section.

II. Related Works

Current main techniques to preserve location privacy for LBS are as follows.

- Information access control [12], [28]: User locations are sent to the LBS provider as usual. This technique relies on the LBS provider to restrict access to stored location data through rule-based polices. It supports three types of location-based queries: 1) user location queries (querying the location of a specific user or users, identified by their unique identifiers); 2) enumeration queries (querying lists of users at specific locations, expressed either in terms of geographic or symbolic attributes); 3) asynchronous queries (querying "event" information, such as when users enter or leave specific areas). This technique requires the LBS provider to maintain all user locations. It is vulnerable to misbehavior of the LBS provider.
- Mix zone [2]: A trusted middleware relays between the mobile users and the LBS provider. Before forwarding the location-based queries of the users to the LBS, the middleware anonymizes their locations by pseudonyms. The basic idea is: when a user enters a mix zone, the middleware assigns him a pseudonym, by which the user queries LBS. The communication between the user and the LBS is through the middleware and the pseudonym changes whenever the user enters the mix zone. Recently, the mix-zone has been applied to road networks [16]. This technique requires the middleware to anonymize user locations. It is vulnerable to misbehavior of the middleware.
- k-anonymity [22]: This technique ensures that a record could not be distinguished from k-1 other records. Instead of sending a single user's exact location to the LBS, k-anonymity based schemes collect k user locations and send a corresponding (minimum) bounding region to the LBS as the query parameter. The collection of different mobile user locations is done either by a trusted third-party [11], [1] between the users and the LBS, or via a

peer-to-peer collaboration [5] among users. Because k-anonymity is achieved, an adversary can only identify a location's user with probability no higher than $1/k$. This technique relies on the third party or a peer user to collect different mobile user locations. It is vulnerable to misbehavior of the third party or the peer user.

- "Dummy" locations [10], [21]: The basic idea is when the mobile user queries the LBS, he sends many random other locations along with his location to the LBS provider to confuse his location such that the server cannot distinguish the actual location from the fake locations. Different from k-anonymity based schemes, this approach include fake or fixed locations, rather than those of other mobile users, as parameters of queries sent to the LBS provider. Fake dummy locations are generated at random, and fixed locations are chosen from special ones such as road intersections. Either way, the exact user locations are hidden from the service provider. Although this technique does not rely on any third party, the LBS provider can restrict the user in a small sub space of the total domain, leading to weak privacy.
- Private Information Retrieval (PIR) [14]: This technique allows a user to retrieve a record from a database server without revealing which record he is retrieving. PIR-based protocols [6], [7], [18], [19] are proposed for POI queries and composed of two stages. In the first stage, the user privately determines the index of his location through the service provider without disclosing his coordinates to it. In the second stage, the user runs a PIR protocol with the service provider to retrieve the POIs corresponding to the index. The difference between Ghinita et al. [6], [7] and Paulet et al. [18], [19] PIR-based protocols is in the first stage, where Ghinita et al. approach is based on homomorphic encryption [15] while the technique of Paulet et al. is based on oblivious transfer [13]. In addition, trusted hardware was employed to perform PIR for LBS queries [17]. Their technique is built on hardware-aided PIR [23], which relies on a trusted third party (TTP) to set the secret key and the permutation of the database. Like LBS queries based on access control, mix zone and k-anonymity, this technique is vulnerable to misbehavior of the third party.
- Geographic data transformation [25], [8], [26]: This technique involves three parties: 1) A data owner who has a database D of points, and would like to outsource D to a server (i.e., cloud service provider) that cannot be fully trusted. 2) A user who wants to access and pose queries to the database D. 3) A server that is honest but potentially curious in the tuples in D and/or the queries from the users. A sever could be curious either because he is just curious or he has been compromised to become curious on the behalf of a third party without his explicit knowledge. In this setting the data owner is different from the LBS. The owner transforms the database (using some encoding methodology) prior to transmitting it to the LBS. An authorized user that possesses the secret transformation keys issues an encoded query to the LBS. Both the database and the queries are unreadable by the LBS and, thus, location privacy is protected. The goal is to provide the LBS with searching capabilities over the encoded data. Wong et al. [25] propose a secure point transformation, which preserves the relative distances of all the database POIs to any query point. Another solution [26] uses the order-preserving encryption [3], [4], given only the encryption of location point $E(q)$ and the encryption of database $E(D)$, the server can return a relevant (encrypted) partition $E(G)$ from $E(D)$, such that that $E(G)$ is guaranteed to contain the answer for the NN query. These techniques allow approximate NN search directly on the transformed points. They are prone to access pattern attacks [24] because the same query always returns the same encoded results.

III. Private k Nearest Neighbor Queries

A. Our Model

Our model considers a location-based service scenario in mobile environments, as shown in Fig. 1, where there exist the mobile user, the location-based service (LBS) provider, the base station and satellites, each playing a different role.

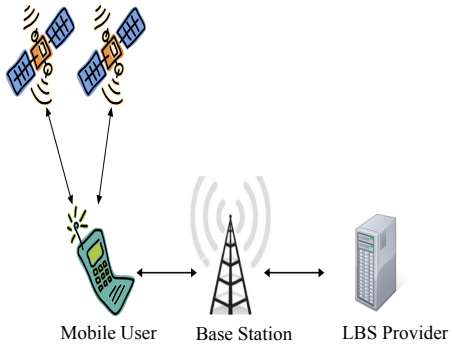

Fig. 1. Location-Based Service

- The mobile user sends location-based queries to the LBS provider and receives location-based service from the provider.
- The LBS provider provides location-based services to the mobile user.
- The base station bridges the mobile communications between the mobile user and the LBS provider.
- Satellites provide the location information to the mobile user.

We assume that the mobile user can acquire his location from satellites anonymously, and the base station and the LBS provider do not collude to comprise the user location privacy or there exists an anonymous channel such as Tor[2] for the

[2]https://www.torproject.org/

mobile user to send queries to and receive services from the LBS provider. Our model focuses on user location privacy protection against the LBS provider and a kNN query protocol (where k is fixed) is composed of three algorithms as follows.

(1) Query Generation (QG): Takes as input a cloaking region CR with $n \times n$ cells and m distinct types of POIs, the location (i, j) of the mobile user and the type t of POIs, (the mobile user) outputs a query Q (containing CR) and a secret s, denoted as $(Q, s) = \mathsf{QG}(CR, n, m, (i, j), t)$.

(2) Response Generation (RG): Takes as input the query Q and the location-based database D of POIs, (the LBS provider) outputs a response R, denoted as $R = \mathsf{RG}(Q, D)$.

(3) Response Retrieval (RR): Takes as input the response R and the secret s of the mobile user, (the mobile user) outputs k nearest POIs of the type t, denoted as $kNN = \mathsf{RR}(R, s)$.

A private kNN query protocol can be illustrated in Fig. 2 and is correct if $kNN = \mathsf{RR}(R, s)$ outputs k nearest POIs of the type t corresponding the cell at (i, j), where $(Q, s) = \mathsf{QG}(CR, n, m, (i, j), t)$ and $R = \mathsf{RG}(Q, D)$.

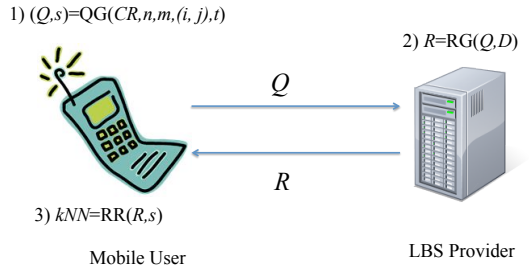

Fig. 2. Private kNN Query

The security of a private kNN query protocol involves data privacy and location privacy. Intuitively, the LBS provider S wishes to release only the k nearest POIs of one type to the mobile user $\mathcal{U}$ each time when the user sends a kNN query. Meanwhile, the mobile user $\mathcal{U}$ does not wish to reveal to the LBS provider his location (i, j) and the type t of POIs he is interested in.

Formally, data privacy can be can be defined with a game as follows.

Given a user location (i, j) where $1 \leq i, j \leq n$ and one type t of POIs, consider the following game between an adversary (the user) $\mathcal{A}$, and a challenger $\mathcal{C}$. The game consists of the following steps:

(1) The adversary chooses any two distinct cloaking regions CR_1 and CR_2 with $n \times n$ cells such that k nearest POIs of the type t in the cell (i, j) are same. The adversary generates a query Q to retrieve the k nearest POIs of the type t in the cell (i, j) and sends Q, CR_1, CR_2 to the challenger $\mathcal{C}$.

(2) The challenger $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and runs the response generation algorithm RG to obtain $R_b = \mathsf{RG}(Q(CR_b), D)$, and then sends R_b back to $\mathcal{A}$.

(3) The adversary $\mathcal{A}$ can experiment with the code of R_b in an arbitrary non-black-box way. If the adversary can retrieve the k nearest POIs of the type t in the cell (i, j) from R_b, he outputs $b' \in \{0, 1\}$.

The adversary wins the game if $b' = b$ and loses otherwise. We define the adversary $\mathcal{A}$'s advantage in this game to be

$$\mathsf{Adv}_{\mathcal{A}}(k) = |\mathsf{Pr}(b' = b) - 1/2|,$$

where k is the security parameter.

Definition 1 (Data Privacy Definition). In a kNN query protocol, the LBS provider has data privacy if for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, we have that $\mathsf{Adv}_{\mathcal{A}}(k)$ is a negligible function, where the probability is taken over coin-tosses of the challenger and the adversary.

Remark. Data privacy ensures that the response distributions on the user's view are computationally indistinguishable for any two cloaking regions CR_1 and CR_2 such that the k nearest POI of the type t in the cell (i, j) in the two cloaking regions are the same. This means that a computationally bounded user does not receive information about more than one cell in the cloaking region CR.

Next, we formally define location privacy with a game as follows.

Give a cloaking region CR with $n \times n$ cells and m types of POIs, consider the following game between an adversary (the LBS provider) $\mathcal{A}$, and a challenger $\mathcal{C}$. The game consists of the following steps:

(1) The adversary $\mathcal{A}$ chooses two distinct tuples (i_0, j_0, t_0) and (i_1, j_1, t_1), where (i_b, j_b) represents the cell and t_b stands for the type of POIs, from the cloaking region CR and sends them to the challenger $\mathcal{C}$.

(2) The challenger $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and executes the Query Generation (QG) to obtain $(Q_b, s) = \mathsf{QG}(CR, n, m, (i_b, j_b), t_b)$ and then sends Q_b back to the adversary $\mathcal{A}$.

(3) The adversary $\mathcal{A}$ can experiment with the code of Q_b in an arbitrary non-black-box way, and finally outputs a bit $b' \in \{0, 1\}$.

The adversary wins the game if $b' = b$ and loses otherwise. We define the adversary $\mathcal{A}$'s advantage in this game to be

$$\mathsf{Adv}_{\mathcal{A}}(k) = |\mathsf{Pr}(b' = b) - 1/2|$$

where k is the security parameter.

Definition 2 (Location Privacy Definition). In a kNN query protocol, the user has location privacy if for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, we have that $\mathsf{Adv}_{\mathcal{A}}(k)$ is a negligible function, where the probability is taken over coin-tosses of the challenger and the adversary.

Remark. Location privacy ensures that the server cannot determine the location of the mobile user in the cloaking region CR and the type of POIs with the kNN query from the mobile user.

Based on our model, we give some constructions of private kNN query protocol which allows the mobile user to find k nearest POIs from a cloaking region. Our solution are built on the Paillier homomorphic encryption scheme [15] and the Rabin encryption scheme [20], both of them are described in Appendix A and Appendix B, respectively.

B. Private kNN Queries without Data Privacy

First of all, we give a basic construction of kNN query protocol without considering data privacy of the LBS provider. We assume that there is only one type of POIs and so we ignore the type of POIs and t in our model in this case.

Initially, the LBS provider divides the location-based database D (a geographic map) into cells with the same size, for example, 1 km width and 1 km length. Based on the center of each cell, the LBS provider collects k nearest POIs, $P_1, P_2, \cdots, P_k$ as shown in Fig. 3 and each point is represented by a tuple (x, y), where x and y are the latitude and longitude of the point, respectively. For each cell (i, j), the LBS provider keeps k nearest POIs, represented as a stream of bits, denoted as an integer $d_{i,j}$. We assume $M = max(d_{i,j})$, i.e., the longest record.

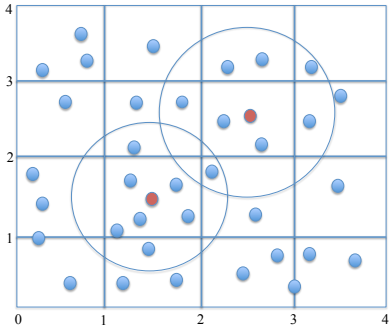

Fig. 3. k Nearest POIs for Cells

Remark. Because the LBS provider collects k nearest POIs according to the center of each cell (i.e., the red points shown in Fig. 3), the LBS provider responses the same k nearest POIs to the two mobile users within the same cell no matter where the two mobile users are in the cell. For the mobile user locating near the border of two cells, he may query two cells or even four cells around his location and then find out k nearest POIs among the query responses. The purpose of our method is to avoid privately comparing distances, which is hard to do without revealing the location of the user.

We assume that the mobile user $\mathcal{U}$ wishes to find k nearest POIs around his location. To do so, the user $\mathcal{U}$ chooses a cloaking region CR with $n \times n$ cells, where $\mathcal{U}$ is located in the cell (i, j), and runs the kNN query protocol with the LBS provider $\mathcal{S}$, composed of three algorithms, Query Generation (QG), Response Generation (RG), and Response Retrieval (RR), as described in Algorithms 1-3.

Algorithm 1 Query Generation (User)

Input: $CR, n, (i, j)$
Output: Q, s

1: Randomly choose two large primes p, q such that $N = pq > M$.
2: Let $sk = \{p, q\}$ and $pk = \{g, N\}$, where g is chosen from $\mathbb{Z}_{N^2}$ and its order is a nonzero multiple of N.
3: For each $\ell \in \{1, 2, \cdots, n\}$, pick a random integer $r_\ell \in \mathbb{Z}^*_{N^2}$, compute

$$c_\ell = \begin{cases} Encrypt(1, pk) = g^1 r_\ell^N (mod\ N^2) & \text{if } \ell = i \\ Encrypt(0, pk) = g^0 r_\ell^N (mod\ N^2) & \text{otherwise} \end{cases}$$

where the encryption algorithm is described in the Paillier cryptosystem (please refer to Appendix A).
4: Let $Q = \{CR, n, c_1, c_2, \cdots, c_n, pk\}$, $s = sk$.
5: **return** Q, s

Remark The CR may be specified by the coordinates (x, y) of an origin point and the order n of a square grid. The cell which contains the origin point is labelled as (1,1). The CR covers the square grid from the cell (1,1) to the cell (n, n).

Algorithm 2 Response Generation RG (Server)

Input: $D, Q = \{CR, n, c_1, c_2, \cdots, c_n, (g, N)\}$
Output: $R = \{C_1, C_2, \cdots, C_n\}$

1: Based on CR and n, compute $R = \{C_1, C_2, \cdots, C_n\}$ where for $\gamma = 1, 2, \cdots, n$,

$$C_\gamma = \prod_{\ell=1}^{n} c_\ell^{d_{\ell,\gamma}} (mod\ N^2)$$

2: **return** R

Algorithm 3 Response Retrieval RR (User)

Input: $R = \{C_1, C_2, \cdots, C_n\}, sk = s$
Output: d

1: Compute

$$d = Decrypt(C_j, sk),$$

where the decryption algorithm is described in the Paillier cryptosystem (please refer to Appendix A).
2: **return** d

Remark In Algorithm 3, when the mobile user receives the response, he can ignore C_ℓ $(\ell \neq j)$ and receive C_j only because only C_j contains the information about the k nearest POIs in the cell (i, j).

Theorem 1 (Correctness) Our kNN query protocol without considering data privacy of the LBS provider (Algorithms 1-3) is correct. In other words, for any cloaking region CR with $n \times n$ and the index i, j of a cell ($1 \leq i, j \leq n$), we have

$$d_{i,j} = \mathsf{RR}(R, s),$$

where $d_{i,j}$ stands for k nearest POIs for the cell (i, j), $(Q, s) = \mathsf{QG}(CR, n, (i, j))$, $R = \mathsf{RG}(D, Q)$.

Proof. Based on Algorithms 1-3, we have

$$C_j = \prod_{\ell=1}^{n} c_\ell^{d_{\ell,j}} = g^{d_{i,j}} (\prod_{\ell=1}^{n} r_\ell^{d_{\ell,j}})^N (mod\ N^2),$$

which is a Paillier encryption of $d_{i,j}$. Therefore, we $d_{i,j} = Decrypt(C_j, sk) = \mathsf{RR}(R, sk)$ and the theorem is proved. $\triangle$

C. Private kNN Queries with Data Privacy

In our kNN query protocol without considering data privacy of the LBS provider, $C_\gamma = g^{d_{i,\gamma}} (\prod_{\ell=1}^{n} r_\ell^{d_{\ell,j}})^N (mod\ N^2)$ and thus the mobile user is able to obtain the k nearest POIs in cells (i, γ) for $\gamma = 1, 2, \cdots, n$. Therefore, it does not have data privacy which requires that the mobile user retrieves the k nearest POIs for one cell only per query.

Now we give a construction of the kNN query protocol, composed of Algorithms 4-6, which provides data privacy for the LBS provider.

Theorem 2 (Correctness) Our kNN query protocol with data privacy (Algorithms 4-6) is correct. In other words, for any cloaking region CR with $n \times n$ and the index i, j of a cell ($1 \leq i, j \leq n$), we have

$$d_{i,j} = \mathsf{RR}(R, s)$$

holds, where $d_{i,j}$ stands for k nearest POIs, $(Q, s) = \mathsf{QG}(CR, n, (i, j))$, $R = \mathsf{RG}(D, Q)$.

Proof. Based on Algorithms 4-6, we have

$$\begin{aligned} C_j &= (c/g^j)^{w_j} \prod_{\ell=1}^{n} c_\ell^{d_{\ell,j}^2} (mod\ N^2) \\ &= g^{d_{i,j}^2} (r^{w_j} \prod_{\ell=1}^{n} r_\ell^{d_{\ell,j}^2})^N (mod\ N^2), \end{aligned}$$

which is a Paillier encryption of $d_{i,j}^2 (mod\ N)$. Therefore, we have $C'_j = PaillierDecrypt(C_j, sk) = d_{i,j}^2 (mod\ N)$, which is the Rabin encryption of $d_{i,j}$. At last, we have $d_{i,j} = RabinDecrypt(C_j, sk) = \mathsf{RR}(R, s)$ and the theorem is proved. $\triangle$

Remark For any γ, $C_\gamma = g^{d_{i,\gamma}^2} g^{(j-\gamma)w_\gamma} (r^{w_\gamma} \prod_{\ell=1}^{n} r_\ell^{d_{\ell,j}^2})^N$. When $\gamma \neq j$, C_γ is not a Paillier encryption of $d_{i,\gamma}^2$ because of $g^{(j-\gamma)w_\gamma}$. This means that the mobile user cannot obtain k nearest POIs for the cell (i, γ) when $\gamma \neq j$. In addition, we use the Rabin encryption $d_{i,j}^2$ instead of $d_{i,j}$ in the response generation to prevent the mobile user from retrieving the nearest POIs for the cell (ℓ, j) when $\ell \neq i$. If we encode $d_{i,j}$ rather than $d_{i,j}^2$, a malicious user may retrieve a linear equation of $d_{1,j}, d_{2,j}, \cdots, d_{n,j}$ by including more than one encryptions of 1 in the list of $c_1, c_2, \cdots, c_n$. The linear relation may disclose more than one $d_{i,j}$ to the user. By Rabin encryption, the user can only retrieve a nonlinear equation of $d_{1,j}, d_{2,j}, \cdots, d_{n,j}$ if there are more than one encryptions of 1 in the list of $c_1, c_2, \cdots, c_n$. From the nonlinear equation, it is hard to retrieve any $d_{i,j}$.

Algorithm 4 Query Generation (User)

Input: $CR, n, (i, j)$
Output: Q, s

1: Randomly choose two large primes p, q such that $N = pq > M$.
2: Let $sk = \{p, q\}$ and $pk = \{g, N\}$, where g is chosen from $\mathbb{Z}_{N^2}$ and its order is a nonzero multiple of N.
3: For each $\ell \in \{1, 2, \cdots, n\}$, pick a random integer $r_\ell \in \mathbb{Z}_{N^2}^*$, compute

$$c_\ell = \begin{cases} Encrypt(1, pk) = g^1 r_\ell^N (mod\ N^2) & \text{if } \ell = i \\ Encrypt(0, pk) = g^0 r_\ell^N (mod\ N^2) & \text{otherwise} \end{cases}$$

4: Pick a random integer $r \in \mathbb{Z}_{N^2}^*$, compute

$$c = Encrypt(j, pk) = g^j r^N (mod\ N^2)$$

5: Let $Q = \{CR, n, c_1, c_2, \cdots, c_n, c, pk\}$, $s = sk$.
6: **return** Q, s

Algorithm 5 Response Generation RG (Server)

Input: $D, Q = \{CR, n, c_1, c_2, \cdots, c_n, c, (g, N)\}$
Output: $R = \{C_1, C_2, \cdots, C_n\}$

1: Based on CR and n, compute $R = \{C_1, C_2, \cdots, C_n\}$ where for $\gamma = 1, 2, \cdots, n$,

$$C_\gamma = (c/g^\gamma)^{w_\gamma} \prod_{\ell=1}^{n} c_\ell^{d_{\ell,\gamma}^2} (mod\ N^2),$$

where w_t is randomly chosen from $\mathbb{Z}_N^*$.
2: **return** R

Algorithm 6 Response Retrieval RR (User)

Input: $R = \{C_1, C_2, \cdots, C_n\}, sk = s$
Output: d

1: Compute

$$C'_j = PaillierDecrypt(C_j, sk),$$

where the decryption algorithm is described in the Paillier cryptosystem (please refer to Appendix A).
2: Compute

$$d = RabinDecrypt(C'_j, sk),$$

where the decryption algorithm is described in the Rabin cryptosystem (please refer to Appendix B).
3: **return** d

D. Private kNN Queries Based on POI Type

Now we take the POI type in kNN query into account. Slightly different from the initialization phase in our kNN query protocol without data privacy, based on the center of each cell, the LBS provider collects k nearest POIs, $P_1, P_2, \cdots, P_k$ and each point is represented by a tuple (x, y, t), where x and y are the latitude and longitude of the point, respectively and t is the type of the points. Examples of POI types includes:

- Churches, schools
- Post offices, shops, postboxes, telephone boxes
- Pubs
- Car parks
- Speed cameras
- Tourist attractions

We assume that POI types are coded into $1, 2, \cdots, m$ which is published to the public. For each cell (i, j) and each POI type t, the LBS keeps k nearest POIs of type t, represented by a stream of bits, denoted as an integer $d_{i,j,t}$. We assume $M = max(d_{i,j,t})$.

Assume that the mobile user $\mathcal{U}$ located in the cell (i, j) wishes to find k nearest POIs of the type t, our kNN query protocol based on POI type is composed of Algorithm 7-9.

Algorithm 7 Query Generation (User)

Input: $CR, n, m, (i, j), t$
Output: Q, s

1: Randomly choose two large primes p_1, q_1 such that $N_1 = p_1 q_1 > M$.
2: Randomly choose two large primes p_2, q_2 such that $N_2 = p_2 q_2$, where $N_1^2 < N_2 < N_1^4$.
3: Let $sk_1 = \{p_1, q_1\}, sk_2 = \{p_2, q_2\}, pk_1 = \{g_1, N_1\}$, $pk_2 = \{g_2, N_2\}$, where g_1 is chosen from $\mathbb{Z}_{{N_1}^2}$ and its order is a nonzero multiple of N_1 and g_2 is chosen from $\mathbb{Z}_{{N_2}^2}$ and its order is a nonzero multiple of N_2.
4: For each $\ell \in \{1, 2, \cdots, m\}$, pick a random integer $r_\ell \in \mathbb{Z}^*_{N_1^2}$, compute

$$c_\ell = \begin{cases} E(1, pk_1) = {g_1}^1 r_\ell^{N_1} (mod\ {N_1}^2) & \text{if } \ell = t \\ E(0, pk_1) = {g_1}^0 r_\ell^{N_1} (mod\ {N_1}^2) & \text{otherwise} \end{cases}$$

5: For each $\ell \in \{1, 2, \cdots, n\}$, pick a random integer $r'_\ell \in \mathbb{Z}^*_{N_2^2}$, compute

$$c'_\ell = \begin{cases} E(1, pk_2) = {g_2}^1 {r'_\ell}^{N_2} (mod\ {N_2}^2) & \text{if } \ell = i \\ E(0, pk_2) = {g_2}^0 {r'_\ell}^{N_2} (mod\ {N_2}^2) & \text{otherwise} \end{cases}$$

6: Pick a random integer $r \in \mathbb{Z}^*_{{N_2}^2}$, compute

$$c = E(j, pk_2) = {g_2}^j r^{N_2} (mod\ {N_2}^2)$$

7: Let $Q = \{CR, n, m, c_1, c_2, \cdots, c_m, c'_1, c'_2, \cdots, c'_n, c, pk_1, pk_2\}$, $s = \{sk_1, sk_2\}$.
8: **return** Q, s

Algorithm 8 Response Generation RG (Server)

Input: $D, Q = \{CR, m, n, c_1, c_2, \cdots, c_m, c'_1, c'_2, \cdots, c'_n, c, pk_1, pk_2\}$
Output: $R = \{C_1, C_2, \cdots, C_n\}$

1: Based on CR and m, for each cell (α, β) in CR, compute

$$C_{\alpha,\beta} = \prod_{\ell=1}^{m} c_\ell^{d^2_{\alpha,\beta,\ell}} (mod\ {N_1}^2)$$

2: Based on CR and n, compute $R = \{C_1, C_2, \cdots, C_n\}$, where for $\beta \in \{1, 2, \cdots, n\}$,

$$C_\beta = (c/g^\beta)^{w_\beta} \prod_{\alpha=1}^{n} {c'_\alpha}^{C^2_{\alpha,\beta}} (mod\ {N_2}^2),$$

where w_β is randomly chosen from $\mathbb{Z}^*_{N_2}$
3: **return** R

Algorithm 9 Response Retrieval RR (User)

Input: $R = \{C_1, C_2, \cdots, C_n\}, sk$
Output: d

1: Compute

$$C'_j = PaillierDecrypt(C_j, sk_2).$$

where the decryption algorithm is described in the Paillier cryptosystem (please refer to Appendix A)
2: Compute

$$C''_j = RabinDecrypt(C'_j, sk_2).$$

where the decryption algorithm is described in the Rabin cryptosystem (please refer to Appendix B)
3: Compute

$$C'''_j = PaillierDecrypt(C''_j, sk_1).$$

4: Compute

$$d = RabinDecrypt(C'''_j, sk_1).$$

5: **return** d

Theorem 3 (Correctness) Our kNN query protocol based on POI type (Algorithms 7-9) is correct. In other words, for any cloaking region CR with $n \times n$ and m types of POIs, and the index i, j of a cell ($1 \le i, j \le n$) and a type t of POIs, we have

$$d_{i,j,t} = \mathsf{RR}(R, sk)$$

holds, where $d_{i,j,t}$ stands for k nearest POIs of the type t in the cell (i, j), and $(Q, sk) = \mathsf{QG}(CR, n, m, (i, j), t)$, $R = \mathsf{RG}(D, Q)$.

Proof. Follow the proof of Theorem 2, we can prove that

$$C''_j = C_{i,,j} = \prod_{\ell=1}^{m} c_\ell^{d^2_{i,j,\ell}} (mod\ {N_1}^2).$$

In fact, $C_{i,j} = g^{d_{i,j,t}^2}(\prod_{\ell=1}^{m} r_\ell^{d_{i,j,\ell}^2})^{N_1}(mod\ N_1^2)$ which is a Paillier encryption of $d_{i,j,t}^2(mod\ N_1)$. Therefore, we have

$$C_j''' = d_{i,j,t}^2(mod\ N_1),$$

which is the Rabin encryption of $d_{i,j,t}$. At last, we have $d_{i,j,t} = RabinDecrypt(C_j''', sk_1) = RR(R, s)$ and the theorem is proved. $\triangle$

E. Private Cloaking Region

In our kNN query protocols, the mobile user needs to specify a cloaking region CR in his query Q. If the CR is too large, the kNN query will be inefficient. However, if the CR is too small, the kNN query has weak location privacy.

To facilitate our kNN query protocols, we give a solution for the mobile user to specify a (small) private cloaking region (encrypted) in a (big) public cloaking region. After that, the mobile user and the LBS provider can run our kNN query protocols over the private cloaking region repeatedly.

Assume that the public cloaking region CR contains $\Delta \times \Delta$ small cloaking regions $CR_{\alpha,\beta}$ ($\alpha = 1, 2, \cdots, \Delta, \beta = 1, 2, \cdots, \Delta$). Without loss of generality, we assume that each small $CR_{\alpha,\beta}$ contains λ data elements, $d_{\alpha,\beta,\gamma}$ for $\gamma = 1, 2, \cdots, \lambda$, although the small $CR_{\alpha,\beta}$ can be further divided into $n \times n$ cells later.

Assume that the mobile user wishes to specify a private cloaking region $CR_{i,j}$ (encrypted), our private cloaking region protocol is composed of Algorithm 10, by which the mobile user generates a request for private cloaking region, and Algorithm 11, by which the LBS provider generates the private cloaking region (encrypted) for the mobile user.

Algorithm 10 Private Cloaking Region Request (User)

Input: CR, Δ, i, j
Output: Q, s

1: Randomly choose two large primes p_1, q_1 such that $N_1 = p_1 q_1 > M$.
2: Randomly choose two large primes p_2, q_2 such that $N_2 = p_2 q_2$, where $N_1^2 < N_2 < N_1^4$.
3: Let $sk_1 = \{p_1, q_1\}, sk_2 = \{p_2, q_2\}, pk_1 = \{g_1, N_1\}$, $pk_2 = \{g_2, N_2\}$, where g_1 is chosen from $\mathbb{Z}_{{N_1}^2}$ and its order is a nonzero multiple of N_1 and g_2 is chosen from $\mathbb{Z}_{{N_2}^2}$ and its order is a nonzero multiple of N_2.
4: For each $\ell \in \{1, 2, \cdots, \Delta\}$, pick a random integer $r_\ell \in \mathbb{Z}_{N_1^2}^*$, compute

$$c_\ell = \begin{cases} E(1, pk_1) = {g_1}^1 r_\ell^{N_1}(mod\ {N_1}^2) & \text{if } \ell = i \\ E(0, pk_1) = {g_1}^0 r_\ell^{N_1}(mod\ {N_1}^2) & \text{otherwise} \end{cases}$$

5: For each $\ell \in \{1, 2, \cdots, \Delta\}$, pick a random integer $r'_\ell \in \mathbb{Z}_{N_2^2}^*$, compute

$$c'_\ell = \begin{cases} E(1, pk_2) = {g_2}^1 {r'_\ell}^{N_2}(mod\ {N_2}^2) & \text{if } \ell = j \\ E(0, pk_2) = {g_2}^0 {r'_\ell}^{N_2}(mod\ {N_2}^2) & \text{otherwise} \end{cases}$$

6: Let $Q = \{CR, \Delta, c_1, c_2, \cdots, c_\Delta, c'_1, c'_2, \cdots, c'_\Delta, pk_1, pk_2\}$, $s = \{sk_1, sk_2\}$.
7: **return** Q, s

Before we describe the private cloaking region generation algorithm, we introduce a notation as follows:

$$c_i^{CR_{\alpha,\beta}} = (c_i^{d_{\alpha,\beta,1}}, c_i^{d_{\alpha,\beta,2}}, \cdots, c_i^{d_{\alpha,\beta,\lambda}})$$

and

$$(c_i^{d_{\alpha,\beta,1}}, c_i^{d_{\alpha,\beta,2}}, \cdots, c_i^{d_{\alpha,\beta,\lambda}})(c_j^{d_{\alpha',\beta',1}}, c_j^{d_{\alpha',\beta',2}}, \cdots, c_j^{d_{\alpha',\beta',\lambda}})$$

$$= (c_i^{d_{\alpha,\beta,1}} c_j^{d_{\alpha',\beta',1}}, c_i^{d_{\alpha,\beta,2}} c_j^{d_{\alpha',\beta',2}}, \cdots, c_i^{d_{\alpha,\beta,\lambda}} c_j^{d_{\alpha',\beta',\lambda}})$$

Algorithm 11 Private Cloaking Region Generation (Server)

Input: $D, Q = \{CR, \Delta, c_1, c_2, \cdots, c_\Delta, c'_1, c'_2, \cdots, c'_\Delta, pk_1, pk_2\}$
Output: R

1: Based on CR and Δ, CR is divided into small cloaking regions $CR_{\alpha,\beta}$ where $1 \le \alpha, \beta \le \Delta$.
2: For $\beta = 1, 2, \cdots, \Delta$, compute

$$CR_\beta = {c_1}^{CR_{1,\beta}} {c_2}^{CR_{2,\beta}} \cdots {c_\Delta}^{CR_{\Delta,\beta}}.$$

3: Compute

$$R = {c'_1}^{CR_1} {c'_2}^{CR_2} \cdots {c'_\Delta}^{CR_\Delta}.$$

4: **return** R

In Algorithm 11, the output R (i.e., the encrypted private cloaking region) contains λ data elements.

Theorem 4. In Algorithms 10 and 11, R is the encryption of private cloaking region $CR_{i,j}$.

Proof. In Algorithms 10 and 11, assume that $CR_\beta = (C_{\beta,1}, C_{\beta,2}, \cdots, C_{\beta,\lambda})$ and $R = (C_1, C_2, \cdots, C_\lambda)$, then for $\gamma = 1, 2, \cdots, \lambda$, we have

$$C_{\beta,\gamma} = \prod_{\ell=1}^{\Delta} c_\ell^{d_{\ell,\beta,\gamma}} = {g_1}^{d_{i,\beta,\gamma}}(r_{\beta,\gamma})^{N_1}(mod\ N_1),$$

which is a Paillier encryption of $d_{i,\beta,\gamma}$ with g_1, N_1. In addition, for $\gamma = 1, 2, \cdots, \lambda$, we have

$$C_\gamma = \prod_{\ell=1}^{\Delta} {c'_\ell}^{C_{\ell,\gamma}} = {g_2}^{C_{j,\gamma}}(r_\gamma)^{N_2}(mod\ N_2),$$

which is a Paillier encryption of $C_{j,\gamma}$ with g_2, N_2. This means $PaillierDecrypt(C_\gamma, sk_2) = C_{j,\gamma}$ and $PaillierDecrypt(C_{j,\gamma}, sk_1) = d_{i,j,\gamma}$ for $\gamma = 1, 2, \cdots, \lambda$ and the theorem is proved. $\triangle$

Remark. If the LBS provider has sufficient storage, it can keep the private cloaking region (PCR) for the time being. The PCR is encrypted and only the mobile user can decrypt. The LBS provider still knows the POI types of data elements in the PCR, but it has no idea where PCR is located. Therefore, the mobile user does not need to hide his location within the PCR in his query and only needs to embed the POI type t in his query in the same way as Algorithm 1. In addition, the user can repeatedly query the different cells in the PCR.

IV. Security Analysis

We have proposed three kNN query protocols and a private cloaking region protocol, we now analyze their security on the basis of our security model given in Section III.

A. Security of Private kNN Query without Data Privacy

Our basic kNN query protocol without data privacy for the LBS provider (Algorithms 1-3) intends to provide location privacy for the mobile user only. Now we prove that this protocol has location privacy based on Definition 2 in Section III. Because we do not consider POI type in this protocol, we ignore the POI type parameter t in the security model.

The security of the basic protocol is built on the security of the Paillier public key cryptosystem.

Theorem 5. If the Paillier cryptosystem is semantically secure, then our kNN query protocol without data privacy has location privacy.

Proof. Suppose our kNN query protocol does not have location privacy, then there is a PPT adversary $\mathcal{A}$ (the LBS provider) who has non-negligible advantage ϵ to break the location privacy of our protocol. Now we use $\mathcal{A}$ to break the semantic security of the Paillier cryptosystem.

Given the public key of the Paillier cryptosystem, we challenge two plaintexts 0 and 1 and the challenger randomly chooses $b \in \{0,1\}$ and returns $E(b)$ to us. We construct our query $Q = \{CR, n, c_1, c_2, \cdots, c_n, (g, N)\}$ where $c_i = E(b)$ and $c_\ell = E(0)$ and send Q to the adversary $\mathcal{A}$, who gives us his guess i'. If $i' = i$, we output our guess $b' = 1$ and 0 otherwise.

When $b = 1$, Q is a real query and the adversary $\mathcal{A}$ wins the game with the probability $1/2+\epsilon$. When $b = 0$, Q is independent of i and the adversary $\mathcal{A}$ can only win the game with 1/2. Therefore, our advantage to break the Paillier cryptosystem (i.e., our guess $b' = b$) is $1/2 \cdot 1/2 + 1/2(1/2+\epsilon) = 1/2 + \epsilon/2$, which is non-negligible. We break the semantic security of the Paillier cryptosystem. It is in contradiction with the assumption of the theorem. Based on Definition 2, our protocol has location privacy and the theorem is proved. $\triangle$

Remark. In our kNN query protocol without data privacy, our query Q is independent of j. Thus there is no way for the LBS provider to guess j with Q.

B. Security of Private kNN Query with Data Privacy

Our basic kNN query protocol with data privacy (Algorithms 4-6) intends to provide both data privacy and location privacy. At first we prove that this protocol has data privacy for the LBS provider based on Definition 1 in Section III. We ignore the POI type parameter t in the security model.

Theorem 6. Our kNN query protocol with data privacy has data privacy for the LBS provider.

Proof. Suppose the mobile user arbitrarily chooses $n+1$ plaintexts $b_1, b_2, \cdots, b_n, b$ and constructs the query $Q = \{CR, n, c_1, c_2, \cdots, c_n, c, (g, N)\}$ by letting $c_\gamma = E(b_\gamma, pk)$ and $c = E(b, pk)$.

Based on Algorithm 5, for $\gamma = 1, 2, \cdots, n$, the LBS provider computes

$$C_\gamma = g^{\sum_{\ell=1}^{n} b_\ell d_{\ell,\gamma}^2} g^{(b-\gamma)w_\gamma} (r^{w_\gamma} \prod_{\ell=1}^{n} r_\ell^{d_{\ell,j}^2})^N (mod\ N^2).$$

Only C_j contains $d_{i,j}$ in $\sum_{\ell=1}^{n} b_\ell d_{\ell,j}^2$. So in order to get $d_{i,j}$, the mobile user has to choose $b = j$. Otherwise, C_j is an Paillier encryption of a random integer because of $g^{(b-j)w_j}$.

When $b = j$, C_j is a Paillier encryption of $\sum_{\ell=1}^{n} b_\ell d_{\ell,j}^2 (mod\ N)$. When computing $d_{\ell,j}^2$, the LBS provider can add some random bits in the end of $d_{\ell,j}$ so that $d_{\ell,j}^2 > N$ for all ℓ. So if there is more than two non-zeros in $\{b_1, b_2, \cdots, b_n\}$, the mobile user cannot split the Rabin encryptions in $\sum_{\ell=1}^{n} b_\ell d_{\ell,j}^2 (mod\ N)$ and a collision occurs. To get $d_{i,j}$, the mobile user has to choose $b_i \neq 0$ and $b_\gamma = 0$ for $\gamma \neq i$.

In the case that $b = j$, C_γ ($\gamma \neq j$) is a Paillier encryption of random integer because of $g^{(j-\gamma)w_\gamma}$ in C_γ. In C_j, the mobile user can only retrieve $d_{i,j}$ by choosing $b_i \neq 0$ and $b_\gamma = 0$ for $\gamma \neq i$. Based on Definition 1, our protocol has data privacy for the LBS provider and the theorem is proved. $\triangle$

Next, we prove that this protocol has location privacy based on Definition 2.

Theorem 7. If the Paillier cryptosystem is semantically secure, our kNN query protocol with data privacy has location privacy.

Proof. Comparing Algorithm 4 (our kNN query protocol with data privacy) and Algorithm 1 (our kNN query protocol without data privacy), we can see that the only difference is c by which Algorithm 4 embeds j coordinate into the query Q. Algorithm 4 embeds i coordinate into Q in the same way as Algorithm 1. We have proved that Algorithm 1 preserves location privacy about i coordinate in Theorem 5. So we only need to prove Algorithm 4 keeping location privacy about j coordinate.

Suppose Algorithm 4 does not have location privacy about j coordinate, then there is a PPT adversary $\mathcal{A}$ (the LBS provider) who has non-negligible advantage ϵ to break the location privacy about j coordinate. Now we use $\mathcal{A}$ to break the semantic security of the Paillier cryptosystem.

Given the public key of the Paillier cryptosystem, we challenge two plaintexts 0 and 1 and the challenger randomly chooses $b \in \{0,1\}$ and returns $E(b)$ to us. We construct our query $Q = \{CR, n, c_1, c_2, \cdots, c_n, (g, N)\}$ where $c = E(b)^j = E(jb)$ and send Q to the adversary $\mathcal{A}$, who gives us his guess j'. If $j' = j$, we output our guess $b' = 1$ and 0 otherwise.

When $b = 1$, Q is a real query and the adversary $\mathcal{A}$ wins the game with the probability $1/2 + \epsilon$. When $b = 0$, Q is independent of j and the adversary $\mathcal{A}$ can only win the game with 1/2. Therefore, our advantage to break the Paillier cryptosystem (i.e., our guess $b' = b$) is $1/2 \cdot 1/2 + 1/2(1/2 + \epsilon) = 1/2 + \epsilon/2$, which is non-negligible. It is in contradiction with the assumption of the theorem. Based on Definition 2, our protocol has location privacy about j and the theorem is proved. $\triangle$

C. Security of Private kNN Query Based on POI Type

In our private kNN query protocol based on POI type (Algorithms 7-9), Algorithm 7 embeds the location (i, j) into the query Q in the same way as Algorithm 4 and embeds the POI type t into the query Q in the same as Algorithm 1. Following the proofs of Theorems 5 and 7, we can prove that our protocol has location privacy about (i, j, t) on the basis of Definition 2.

In addition, in Algorithm 8, only $C_{i,j}$ contains $d_{i,j,t}$. If the mobile user wants to get $d_{i,j,t}$, he has to get $C_{i,j}$ from C_j at first. Following the proof of Theorem 6, we can prove that the mobile user has to embed i, j into Q as Algorithm 7 to obtain $C_{i,j}$. In the same way, we can prove that the mobile user has to embed t into Q as Algorithm 7 to get $d_{i,j,t}$. In the response R to such a query Q, C_β $(\beta = j)$ is a Paillier encryption of a random integer. From C_j, the mobile user can retrieve $d_{i,j,t}$ only.

Based on the above analysis, we conclude

Theorem 8. If the Paillier cryptosystem is semantically secure, our kNN query protocol based on POI type has both location privacy and data privacy.

D. Security of Private Cloaking Region

For our private cloaking region protocol (Algorithms 10-11), Algorithm 10 embeds i, j into the private cloaking region request in the same way as Algorithm 1. Following the proof of Theorem 5, we can show that Algorithm 10 can preserve the privacy of i, j and thus keep the small cloaking region $CR_{i,j}$ private to the LBS provider.

V. Performance Analysis

Now we analyze the performance of our three kNN query protocols and our private cloaking region protocol. In the performance analysis, we consider the computation of modular exponentiations (exp.) and ignore the computation of modular multiplications and squares because the latter is much cheaper than the former. We also ignore the process of key generation because it can be pre-computed.

A. Performance of Our Protocols

In our kNN query protocol without data privacy (Algorithms 1-3), the mobile user needs to compute n Paillier encryptions (about n exp.) in Algorithm 1, and 1 Paillier decryption (about 2 exp.) in Algorithm 3. So the total computation complexity of the mobile user is about $O(n)$ exp. In Algorithm 2, the LBS provider needs to compute n^2 exp. and the total computation complexity of the LBS provider is $O(n^2)$ exp. In addition, the communication complexity is $2n \log_2 N$ bits.

In our kNN query protocol with data privacy (Algorithms 4-6), the mobile user needs to compute $n+1$ Paillier encryptions (about n exp.) in Algorithm 4, and 1 Paillier decryption and 1 Rabin decryption (about 3 exp.) in Algorithm 6. So the total comp. complexity of the user is about $O(n)$ exp. In Algorithm 5, the LBS provider needs to compute $(2 + n)n$ exp. and the total comp. complexity of the LBS provider is $O(n^2)$ exp. In addition, the comm. complexity is $2n \log_2 N$ bits.

Component	*Algorithms 1-3*	*Algorithms 4-6*	*Algorithms 7-9*
User Comp.	$O(n)$	$O(n)$	$O(n + m)$
Server Comp.	$O(n^2)$	$O(n^2)$	$O(mn^2)$
Comm.	$2n \log_2 N$	$2n \log_2 N$	$(2n + m) \log_2 N$

TABLE I
Performance of Our kNN Query protocols

In our kNN query protocol based on POI type (Algorithms 7-9), the mobile user needs to compute $n + m + 1$ Paillier encryptions (about $n + m$ exp.) in Algorithm 7, and 2 Paillier decryption and 2 Rabin decryption (about 6 exp.) in Algorithm 9. So the total computation complexity of the mobile user is about $O(2n)$ exp. In Algorithm 8, the LBS provider needs to compute $mn^2 + (n + 2)n$ exp. and the total computation complexity of the LBS provider is $O(mn^2)$ exp. In addition, the communication complexity is $(2n + m) \log_2 N$ bits.

Table I shows the performance of the above three protocols.

In addition, in our private cloaking region protocol (Algorithms 10-11), the mobile user needs to compute 2Δ Paillier encryptions (about 2Δ exp.) in Algorithm 10 while the LBS provider needs to compute $\lambda\Delta^2$ exp., and the communication complexity is $2\Delta \log_2 N$. After generation of the private cloaking region, the mobile user can repeatedly query it with $O(1)$ (without POI type) or $O(m)$ (with POI type) computation and communication complexities.

B. Performance Comparison

We now compare our kNN query protocol with data privacy with PIR-based LBS query protocols [6], [7], [18], [19] in TABLE II. All these protocols do not consider POI type in their queries. We assume the cloaking region has $n \times n$ cells.

From TABLE II, we can see that Ghinita et al. and Paulet et al. protocols both have two stages while our protocol has one stage only. The performance of our protocol is better than Ghinita et al. protocol in terms of user and server computation complexities and communication complexity. In addition, Paulet et al. protocol and our protocol have almost the same server computation and communication complexities. The mobile user in our protocol needs to compute much less than Paulet et al. protocol. In stage 2, Paulet et al. protocol needs to generate a group G, a generator g and a prime q for each query and compute a discrete logarithm $c_i = \log_h h_e$. This process takes more time than computing n exp.

VI. Experimental Evaluation

We implemented a prototype of our kNN query protocol based on POI type, which is composed of Algorithms 7-9, to test the performance of our solution. The prototype was executed on a machine with an Intel Core i7-2600 processor at a clock speed of 3.40GHz, and with 16GB of RAM. The experiment used Linux as the operating system and is written using the C programming language. We used the GMP library for computations using large integers.

We set $n = 25$ for the size of the $n \times n$ grid and set $m = 5$ for the number of types of points of interest. We constructed

Component	*Ghinita et al.*	*Paulet et al.*	*Our Protocol*
User Comp.	$O(n^2)/O(n)$	$O(1)$ / generate G, g, q and solve discrete log	$O(n)$
Server Comp.	$O(n^2)/O(n^2)$	$O(n)/O(n^2)$	$O(n^2)$
Comm.	$n^2 \log_2 N/2n \log_2 N$	$2n \log_2 N/O(1)$	$2n \log_2 N$

TABLE II
PERFORMANCE COMPARISON (STAGE 1/STAGE 2)

Component	*Paulet et. al*	*Our Protocol*
Query Gen.	0.00484s / 9.6498s	0.157726s
Res. Gen.	0.11495s / 12.6978s	8.661929s
Res. Retrieval	0.0031s / 0.25451s	0.016211s

TABLE III
AVERAGE EXECUTION TIME FOR OUR PROTOCOL AND PAULET ET AL. PROTOCOL (STAGE 1/ STAGE 2)

two instances of the Paillier cryptosystem, with their respective private / public key pairs $(sk_1 = (p_1, q_1), pk_1 = (g_1, N_1))$ and $(sk_2 = (p_2, q_2), pk_2 = (g_2, N_2))$. The instances were created such that the modulus of $N_1 = p_1 q_1$ is 1025 bits in length, and the modulus of N_2 is 2051 bits long. The whole protocol was executed for 100 trials, the results of our protocol compared with Paulet et al. protocol are listed in TABLE III. This table shows that our protocol based on POI type is even more efficient than Paulet et al.'s protocol without considering POI type.

We ignored public key initialisation because these variables can be precomputed, and hence not taken into consideration for runtime execution. Based on the results, we can see that our solution is practical.

VII. CONCLUSION

In this paper, we have presented three private kNN query protocol and one private cloaking region protocol. We have proved that our protocols are all correct. Security analysis has shown that all of our protocols has location privacy. Our protocol with data privacy and our protocol based on POI type have data privacy for the LBS provider. Performance has shown that our protocol with data privacy is more efficient than previous PIR-based LBS query protocols. Experiment evaluation has shown that our protocol based on POI type is practical. Note that some experiment results are not included in the paper because of page limit. Our future work is to implement our protocols on mobile devices and evaluate performance.

REFERENCES

[1] B. Bamba, L. Liu, P. Pesti, and T. Wang. Supporting anonymous location queries in mobile environments with PrivacyGrid. in Proc. WWW 2008.

[2] A. R. Beresford and F. Stajano. Location privacy in pervasive computing. IEEE Pervasive Computing 2(1), 2003.

[3] A. Boldyreva, N. Chenette, Y. Lee, and A. ONeill. Order-preserving symmetric encryption. In Proc. EUROCRYPT 2009.

[4] A. Boldyreva, N. Chenette, and A. ONeill. Order-preserving encryption revisited: Improved security analysis and alternative solutions. In Proc. CRYPTO 2011.

[5] C. Y. Chow, M. F. Mokbel, and X. Liu. A peer-to-peer spatial cloaking algorithm for anonymous location-based services. In Proc. ACM GIS 2006.

[6] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private queries in location-based services: Anonymizers are not necessary. in Proc. ACM SIGMOD 2008.

[7] G. Ghinita, P. Kalnis, and S. Skiadopoulos. PRIVE: Anonymous location-based queries in distributed mobile systems. in Proc. WWW 2007.

[8] Haibo Hu, Jianliang Xu, Chushi Ren, and Byron Choi, Processing Private Queries over Untrusted Data Cloud through Privacy Homomorphism, in Proc. ICDE 2011.

[9] A. Khoshgozaran and C. Shahabi. Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In Proc. SSTD 2007.

[10] H. Kido, Y. Yanagisawa, and T. Satoh. An anonymous communication technique using dummies for location-based services. In Proc. ICPS 2005, pages 88 - 97.

[11] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new casper: query processing for location services without compromising privacy. In Proc. VLDB 2006.

[12] G. Myles, A. Friday, and N. Davies. Preserving privacy in environments with location-based applications. IEEE Pervasive Computing 2(1):5664, 2003.

[13] M. Naor and B. Pinkas. Oblivious transfer with adaptive queries. In Proc. CRYPTO 1999, pages 791 - 791.

[14] R. Ostrovsky and W. Skeith. A survey of single-database private information retrieval: techniques and applications. In Proc. PKC 2007, pages 393 - 411.

[15] P. Paillier. Public key cryptosystems based on composite degree residue classes. In Proc. EUROCRYPT 1999, pages 223 - 238.

[16] B. Palanisamy and L. Liu. Mobimix: Protecting location privacy with mix-zones over road networks. In Proc. ICDE 2011, pages 494 505.

[17] S. Papadopoulos, S. Bakiras, D. Papadias. Nearest neighbor search with strong location privacy. In Proc. VLDB 2010.

[18] R. Paulet, M. Golam Kaosar, X. Yi, and E. Bertino. Privacy-preserving and content-protecting location based queries. In Proc. ICDE 2012, pages 44 - 53.

[19] R. Paulet, M. Golam Kaosar, X. Yi, and E. Bertino. Privacy-preserving and content-protecting location based queries. IEEE Transactions on Knowledge and Data Engineering, accepted in 2013.

[20] Rabin, Michael. Digitalized Signatures and Public-Key Functions as Intractable as Factorization. MIT Laboratory for Computer Science, January 1979.

[21] P. Shankar, V. Ganapathy and L. Iftode. Privately querying location-based services with SybilQuery. In Proc. Ubicomp 2009, pages 31 - 40.

[22] L. Sweeney. k-anonymity: a model for protecting privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 10: 557 - 570, 2002.

[23] S. Wang, X. Ding, R. H. Deng, and F. Bao. Private information retrieval using trusted hardware. In Proc. ESORICS 2006.

[24] P. Williams and R. Sion. Usable PIR. In NDSS, 2008.

[25] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, Secure kNN computation on encrypted databases. In Proc. SIGMOD 2009.

[26] B. Yao, F. Li, and X. Xiao, Secure nearest neighbor revisited. In Proc. ICDE 2013.

[27] M. L. Yiu, C. Jensen, X. Huang, and H. Lu. SpaceTwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile systems. In Proc. ICDE 2008.

[28] M. Youssef, V. Atluri, and N. R. Adam. Preserving mobile customer privacy: An access control system for moving objects and custom proles. In Proc. MDM 2005.

APPENDIX A: PAILLIER PUBLIC-KEY CRYPTOSYSTEM

Let $N = pq$, where p and q are large primes, and g be an element of $Z^*_{N^2}$, whose order is a nonzero multiple of N, then a map

$$\mathcal{E}_g : \begin{cases} Z_N \times Z_N^* & \longmapsto & Z_{N^2}^* \\ (x, y) & \longmapsto & g^x y^N \ (mod\ N^2) \end{cases} \quad (1)$$

is bijective.

The N-th residuosity class of an integer c (of $Z^*_{N^2}$) with respect to g, denoted as $[\![c]\!]_g$, is the unique integer x in Z_N, such that $c = g^x y^N (mod\ N^2)$ for an integer y in Z_N^*. $[\![c]\!]_g$ can be computed with

$$[\![c]\!]_g \quad = \quad \frac{(c^\lambda (mod\ N^2) - 1)/N}{(g^\lambda (mod\ N^2) - 1)/N} (mod\ N) \quad (2)$$

where $\lambda = lcm(p-1, q-1)$, i.e., the least common multiple of $p-1$ and $q-1$. It can be seen that the N-th residuosity class of c with respect to g cannot be computed without p and q. Based on this property, Paillier [15] proposed a public key cryptosystem as follows.

- *Key Generation*: A user randomly choose two large distinct primes p, q and an element g of $Z^*_{N^2}$ whose order is a nonzero multiple of $N = pq$, publish the public keys (N, g), and keeps the private keys (p, q) secret.
- *Encryption*: Given the public key (N, g) of the user, one can encrypt a message m where m is a positive integer less than N by randomly choosing r from Z_N^* and computing

  $$c = \mathcal{E}_g(m) = g^m r^N \ (mod\ N^2) \quad (3)$$

  c is the ciphertext of m. Since r is randomly chosen, the ciphertext c of a message m is random. Therefore, Paillier cryptosystem is a probabilistic encryption.
- *Decryption*: The user can decrypt the ciphertext c with the private key (p, q) by computing $m = [\![c]\!]_g$ according to (2) where $\lambda = lcm(p-1, q-1)$.

Homomorphic Properties: Paillier cryptosystem has two homomorphic encryption properties as follows:

$$\mathcal{E}_g(m_1)\mathcal{E}_g(m_2) \quad = \quad \mathcal{E}_g(m_1 + m_2) \quad (4)$$

$$\mathcal{E}_g(m_1)^a \quad = \quad \mathcal{E}_g(am_1) \quad (5)$$

for any $m_1, m_2, m, a \in Z_N$.

APPENDIX B: RABIN PUBLIC-KEY CRYPTOSYSTEM

Rabin cryptosystem [20] is based on quadratic residues and its properties.

Let a be any integer and n a natural number, suppose that the greatest common divisor of a and n is 1, i.e., $gcd(a, n) = 1$, then a is called a quadratic residue modulo n if the congruence

$$x^2 \quad = \quad a (mod\ n) \quad (6)$$

is soluble. The solutions are called modular square roots of quadratic residue a modulo n.

If $p = q = 3(mod\ 4)$ and a is a quadratic residue modulo $n = pq$, we can determine all four modular square roots r_1, r_2, r_3, r_4 by letting

$$x = a^{\frac{p+1}{4}} \ (mod\ p) \quad (7)$$

$$y = a^{\frac{q+1}{4}} \ (mod\ q) \quad (8)$$

and computing

$$r_{1,2} \quad = \quad x \cdot q \cdot q^* \pm y \cdot p \cdot p^* \ (mod\ n) \quad (9)$$

$$r_{3,4} \quad = \quad -x \cdot q \cdot q^* \pm y \cdot p \cdot p^* \ (mod\ n) \quad (10)$$

where $p^* = p^{-1} \ (mod\ q)$ and $q^* = q^{-1} \ (mod\ p)$. Since $gcd(p, q) = 1$, both p^* and q^* can be computed based on Euclid algorithm.

Based on the above property, Rabin cryptosystem can be described as follows:

- *Key Generation*: A user chooses two distinct large primes p, q such that $p = q = 3\ (mod\ 4)$ and computes the product $n = p \cdot q$. Then he publishes n but keep p, q secret.
- *Encryption*: Give the public key n, one can encrypt a message m (where $m \cdot 2^\ell < n$ and ℓ is a integer more than 30) by calculating $c = (m \cdot 2^\ell)^2 \ (mod\ n)$. c is a ciphertext of m
- *Decryption*: The user can decrypt the ciphertext c with the private key p, q by computing all four distinct modular square roots of $x^2 = c \ (mod\ n)$ on the basis of formulae (7)-(10). The modular square root whose ℓ least-significant bits are zeroes is $m \cdot 2^\ell$, from which the original message m can be obtained.