This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Liu, Shushu; Liu, An; Yan, Zheng; Feng, Wei

Efficient LBS queries with mutual privacy preservation in IoV

Published in:
Vehicular Communications

DOI:
[10.1016/j.vehcom.2019.03.001](https://doi.org/10.1016/j.vehcom.2019.03.001)

Published: 01/04/2019

Document Version
Peer reviewed version

Published under the following license:
CC BY-NC-ND

Please cite the original version:
Liu, S., Liu, A., Yan, Z., & Feng, W. (2019). Efficient LBS queries with mutual privacy preservation in IoV. *Vehicular Communications*, 16, 62-71. <https://doi.org/10.1016/j.vehcom.2019.03.001>

Efficient LBS Queries with Mutual Privacy Preservation in IoV^{*}

Shushu Liu^a, An Liu^{b,*}, Zheng Yan^{c,a,*}, Wei Feng^c

^a*Department of Comnet, Aalto University, Maarintie 8, 02150 Espoo, Finland*

^b*School of Computer Science and Technology, Soochow University, 215006 Suzhou, China*

^c*State Key Lab of Integrated Services Networks, School of Cyber Engineering, Xidian University, 710071 Xi'an, China*

Abstract

Public awareness on privacy stimulates many researches about privacy-preserving location based services (LBS) in terms of providing mutual privacy to both LBS and its users. However, the high latency of privacy preservation in LBS becomes a main obstacle for applying LBS to Internet of Vehicles (IoV). To solve this problem, we propose two privacy-preserving LBS query schemes (kNN and T-kNN) by taking the advance of fog computing and by applying oblivious transfer (OT) and ciphertext-policy attribute based encryption (CP-ABE). Given a query from a vehicle, both schemes return k nearest POIs as response, with the difference that T-kNN supports fine-grained type based POI queries. Based on our proposed oblivious key transfer and privacy-preserving secret key generation, both schemes preserve mutual privacy of both LBS provider and vehicles. Complexity analysis and empirical study show that our approach outperforms the other two state-of-the-art works.

Keywords: LBS, fog computing, IoV, kNN query with type

1. Introduction

As one of the most popular location-awareness applications, location based services (LBS) is widely applied in the context of Internet of Vehicles (IoV) [1, 2]. It facilitates people with great convenience in traffic tracking, points of interest (POI) searching, advertisement receiving, and so on. With location related queries from vehicles, an LBS provider responses with POIs tailored to that location. For example, with specific queries, a vehicle can get information like k nearest gas stations, restaurants, or retail stores to its current geographical position from the LBS provider.

Despite the benefits of LBS, privacy is a serious concern for launching it in practice, as query information collected from vehicles can reveal far more than their latitude and longitude [3, 4]. Knowing where a vehicle is plus some background knowledge can infer many sensitive information about vehicle drivers and passengers, such as home address, hobbies, travel trajectory and destination, which could greatly threat their properties and even life security. On the other hand, from the perspective of LBS provider, POI data should be protected from any unauthorized parties for free access because it is considered as a valuable asset of the LBS provider that requires enormous investment to collect and maintain [5]. A mature LBS should provide mutual privacy with respect to both LBS and vehicles.

The above concern stimulates researches in designing privacy-preserving LBS. Solutions such as Mix zone [6, 7], k-anonymity [8, 9, 10] and dummy location [11, 12, 13] were firstly provided to prevent LBS from locating the user. However, they

^{*}This work is sponsored by the Academy of Finland (grant No. 308087), the NSFC (grants 61672410 and U1536202), the Project Supported by Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2016ZDJC-06), the 111 project (grants B08038 and B16037).

^{*}Corresponding author

Email addresses: anliu@suda.edu.cn (An Liu), zyan@xidian.edu.cn (Zheng Yan)

Figure 1: System overview

fail to support mutual privacy without considering the protection of POI data that are valuable assets of the LBS provider and should not be accessed freely [14]. Cryptographic techniques like homomorphic encryption [15], private information retrieval (PIR) [16, 17] were then introduced to solve this problem. In spite of the mutual privacy they provided, they also incur heavy overheads, which is prohibitive in practice.

On the other hand, overheads caused by protecting privacy is not the only reason of high latency. Traditional LBS systems normally maintain a centralized processing scheme with LBS remotely deployed in the cloud [18, 3]. Generally, each query needs to go through the whole network to reach a LBS provider. Even with the high coverage of the Internet connectivity of IoV, the further adoption of LBS in the field of IoV is still limited due to the high-latency and network bandwidth cost induced by centralized processing scheme.

Owing to the mobility of vehicles, the low-latency becomes very crucial for LBS queries, otherwise obtained results may not satisfy the current spatial range of the vehicle. Therefore, the above existing solutions are not applicable to IoV where query responses should be immediate and reliable.

To address the above problems and practically deploy LBS into IoV, we take the advance of fog computing and propose two privacy-preserving LBS schemes. Fig 1 shows the hierarchical struc-

ture of a fog computing based LBS system. In this system, the LBS provider outsources part of its service to the fog nodes (e.g., base stations and road-side units) and vehicles can query these nodes for POIs searching. Since each node is only responsible for nearby POIs in a specific range, the query starts from roadside units (the leaf node) and goes to the upper level until the root node (the LBS provider) when the query cannot be satisfied at the current node. This design makes up the disadvantage of centralized processing in the cloud. By processing data closer to its query source, it can significantly expedite computation and save bandwidth required for transmitting data, thus enabling low-latency query response. In addition, compared with traditional centralized systems, this decentralized design is more resistant to bottleneck problem when massive vehicles query simultaneously [19].

In terms of privacy preservation, we propose two LBS query schemes: k -nearest neighbours (k NN) query and type based k -nearest neighbours (T- k NN) query since these two schemes can be easily adapted to other queries like nearest neighbour or range query by setting $k = 1$ or setting distance threshold for retrieved POIs. Given a vehicle's GPS coordinates, k NN returns the k nearest POIs to the vehicle considering only location matching. T- k NN takes as input not only coordinates but also POI types and output k nearest POIs of required types. In general, both of the proposed schemes are composed of two phases: *system initialization* and *online query*. The system initialization is processed offline where the LBS provider encrypts its POI data and distributes the encrypted data to each fog node according to their coverage regions. The online query is performed based on GPS coordinates and optional POI type preference. In this process, the vehicle obtains some appropriate private keys from the LBS provider and then performs decryption over the encrypted POIs retrieved from a fog node. By applying oblivious transfer (OT) [20] and ciphertext-policy attribute based encryption (CP-ABE) [21], we realize oblivious key transfer and privacy-preserving secret key (SK) generation to achieve privacy-preserving key retrieval. It is

noteworthy that not all but only the query related POIs can be decrypted during this process. In particular, with sophisticated design of initialization, we decrease the required number of keys from $O(n)$ to $O(\log_2 n)$, thus dramatically reduces latency. Thus, our schemes have a great improvement in online query response compared with the state-of-the-arts.

Specifically, the contributions of this paper can be summarized as below.

- We propose a new LBS framework based on fog computing to support mobile query in IoV. Compared with traditional centralized systems, the decentralized design provides a low-latency and reliable response with resistant to incessant query problems.
- Based on OT and CP-ABE, we propose two privacy-preserving LBS schemes named k NN and T- k NN to achieve mutual privacy preservation in terms of both LBS and vehicles. In particular, T- k NN supports fine-grained type-based POI query.
- Owing to the mobility of vehicles, computational cost and communication delay should be as little as possible. Both complexity analysis and empirical study shows that our approach outperforms the other two state-of-the-art works.

The organization of this paper is as follows. Section 2 briefly reviews some related work. Section 3 defines our research problem and gives an overview on our approach. Section 4 and Section 5 describe the details of our schemes and their security analysis, followed by experimental results presented in Section 6. Finally, we conclude this paper in Section 7.

2. Related Work

The increasing public attention to privacy has spurred many research efforts in privacy-preserving LBS. Solutions such as Mix zone [6, 7], k -anonymity [8, 9, 10] and dummy location [11, 12, 13] were firstly provided to prevent LBS from

locating the user. By injecting some fake locations that are undistinguishable from the real one into each query, the users make the LBS provider unable to locate their exact locations. However, the above works fail to support mutual privacy without considering the protection of POI data that are valuable and should be kept secret from unauthorized users. In addition, the generation of fake locations highly depends on the distribution and density of users, which could lead to weak privacy preservation when only a limited number of users are available in infrequently visited regions, not to mention the extra computation and communication cost induced by the above methods.

An integration of cryptographic techniques guarantees more strong privacy than former methods. Based on cryptographic techniques like homomorphic encryption [15], private information retrieval (PIR) [16, 17], mutual privacy protection can be achieved. Based on Computational PIR [22], Ghinita et al [16] proposed a method for nearest neighbors searching which aims to retrieve the POIs in the target cell without compromising the location information in the query. A similar work was introduced in [17], but using OT and PIR in retrieving target cell. Yi et al. [15] firstly proposed a k NN query with location privacy by applying Paillier Homomorphic encryption [23] and Rabin encryption [24]. Although the above schemes support k NN query with POI types, different from multiple-type query supported by our scheme proposed in this paper, they only support one interest type for each POI, which obviously cannot satisfy practical user demand. On the other hand, despite strong privacy preservation, the computational costs induced by these schemes is big, thus they are unable to provide a low latency service in IoV.

Fog computing has been introduced to make up the disadvantage of the centralized cloud computing [25]. However, existing applications of fog computing mainly focus on wireless sensor networks [26] or real-time navigation of vehicular ad hoc networks [27]. Little work has paid attention to privacy issues in IoV. One related work was proposed by Yang et al [19], a fine grained and privacy-preserving query scheme named FG PQ

Figure 2: Overview of mutual privacy preserving location-based queries

for fog computing-enhanced location-based service. Different from range query provided in [19], we more focus on k NN query with different types, which is more commonly needed in practice.

3. Problem Definition

3.1. System Model

Fig. 1 shows the system model of fog computing enhanced LBS in the context of IoV. There are three types of entities: LBS provider, fog nodes and vehicles. The LBS provider outsources the POI data to the fog nodes and vehicles can query fog nodes for POIs searching. Currently, we provide two types of location-based queries: k NN query and T- k NN query. Given a vehicle's GPS coordinates, k NN return the k nearest POIs to the vehicle. The T- k NN also takes POI types as input not only coordinates and output k nearest POIs with the expected types.

Fig 2 gives a high level overview of our design of privacy preserving LBS query. It is composed of two major steps: *system initialization* and *online query*. *System initialization* is an offline processing where LBS encrypts its POI data and distributes the encrypted data to each fog node according to their coverage region. As the owner of POI dataset, the LBS provider is responsible for key generation and data encryption. The encrypted POIs are then downloaded by the

fog node through a stable and fast network connection. The key used for encryption is held by the LBS provider. *Online query* is an online process where the vehicle first obtains some appropriate private keys from the LBS provider based on its GPS coordinate and optional POI type preference, and then performs some decryption over the encrypted POIs obtained from the fog nodes to receive the corresponding query results. It is noteworthy that not all but only query related POIs can be decrypted during this process.

It is also remarkable that in our model, we assumed an ideal situation where the whole map is covered and evenly divided by fog nodes, when a vehicle enters some region, it sends out query to the closet node, and in the tricky location where the distance is equal to two or more nodes, it should connect to the fog node in the moving direction or random one in the worst situation. And for the situation when the queried result is not supported in current region, it goes to higher level node covering bigger region until the whole map, so in the ideal situation, all queries will be responded if there is a positive result. However, it still has some limitation in practice since it may not be that easy to achieve a fully coverage of the whole map, uncovered region may appear sometimes, but since vehicles are moving fast, they can pass this region quickly and connect to the LBS service immediately once they enter a new covered region again. Thus, it is still feasible as long as the main map region are covered by node.

Algorithm 1: Privacy Preserving k NN Query - system initialization

Input: LBS with POI database

Output: LBS with encryption key, Fog nodes with outsourced POIs

- 1: LBS builds a grid index P with $M = m \times m$ cells over its POI database and the content of each cell is $c = m_{x,y}^1 |m_{x,y}^2| \cdots |m_{x,y}^d|$
 - 2: LBS selects key generator K , which is $l = \log_2 M$ pairs random values $(K_1^0, K_1^1), (K_2^0, K_2^1), \dots, (K_l^0, K_l^1)$
 - 3: For each cell I , LBS selects a unique key from $K_I = K_1^{i_1}, K_2^{i_2}, \dots, K_l^{i_l}$ with binary representation of $I = (i_1, i_2, \dots, i_l)$
 - 4: For each cell, LBS encrypts the content c by $E(c) \leftarrow H(g^{\prod_{k=1}^l K_k^{i_k}}, d\alpha) \otimes c$
 - 5: LBS distributes encrypted cells to fog nodes according to their coverage region
-

3.2. Threat Model

We take an assumption that the system is in a semi-honest adversary model, that is, all the parties will follow the scheme specification, but at the same time, attempt to use the observed internal state to learn more information than output. Although malicious adversary model is preferred for achieving stronger security, it is too inefficient to implement and to be used in practice.

Herein, we focus on the security between the vehicle and the LBS provider because the fog node is just an intermediate node for encrypted data storage and communications. Concretely, the fog nodes are responsible for storing of nearby POIs and answering the pull request from vehicles. However, the POIs stored in fog node are all encrypted in a secure way and the key for decryption is hold by LBS. Except data transmission for storage, no other further communications between the fog node and the LBS provider, thus the query and POI information are secure from the fog nodes. Therefore, we only consider two possible adversaries: the LBS provider that tries to obtain the sensitive information about vehicles from their location-based queries, for example, their physical locations and their hobbies through the types of POIs they queried; the vehicle that aims to obtain the POIs unauthorized to access, which are not the answer to its own location-based queries. Hence, the objective of our work is to design a scheme that can protect mutual privacy (i.e., the queries from the vehicle and the POI data owned by the LBS provider) when processing location-based queries and offering LBS.

This paper adopts the *ideal/real simulation paradigm*, which is a standard security formulation in the area of secure multi-party computation [28]. In this formulation, a real protocol execution in a “real world” is mapped to an ideal protocol realization in an “ideal world”. In the ideal world, there exists a secure physical channel, where the adversary can only observe messages sent. A protocol is said to be *secure* if for all adversaries, there exists a simulator, so that the outputs of the “adversary in the real game” and “simulator in the ideal game” are indistinguishable.

4. Privacy-Preserving k NN Query

We first present our scheme of privacy preserving k NN query. In this query, both POI data and vehicle query are represented by the GPS coordinates (x, y) without type restriction. Its algorithms and security analysis are elaborated in the following.

Algorithm 2: Privacy Preserving k NN Query - online query processing

Input: LBS with key generator K , Fog node with encrypted map, Vehicle v with location query q
Output: v gets k nearest POIs to q

- 1: v downloads encrypted map from fog node
- 2: v inserts all cells of P to a priority queue T
- 3: **while** T is not empty and $\max(T) < \min(L)$ **do**
- 4: $I = \text{Dequeue}(T)$
- 5: LBS chooses random elements $r_1, r_2, \dots, r_l$ and compute $g^{1/\prod_{k=1}^l r_k}$
- 6: LBS feeds Alg. 3 with input $(K_1^0 r_1, K_1^1 r_1), (K_2^0 r_2, K_2^1 r_2), \dots, (K_l^0 r_l, K_l^1 r_l)$
- 7: v obtains l keys $K_1^{i_1} r_1, K_2^{i_2} r_2, \dots, K_l^{i_l} r_l$ by feeding I into Alg. 3
- 8: v obtains the decryption key for c_I by $g^{\prod_{k=1}^l K_k^{i_k}} = (g^{1/\prod_{k=1}^l r_k})^{\prod_{k=1}^l K_k^{i_k} r_k}$
- 9: v obtains POIs by decrypting $E(c_I)$ with $H(g^{\prod_{k=1}^l K_k^{i_k}}, d\alpha) \otimes E(c_I)$
- 10: v updates L using newly retrieved POIs
- 11: **end while**
- 12: v takes POIs in L as the query result

4.1. System Initialization

Algorithm 1 is system initialization of privacy-preserving k NN query. We adopt the grid index proposed in [15] for k NN query. More specifically, LBS divides the geographic map of its POI database into $M = m \times m$ cells with the same size (M is public). Suppose each POI can be represented by an α -bits integer, by concatenating all POIs located in that cell, each cell is represented by an $d\alpha$ -bits integer, where d is the maximum number of POIs among all cells. The generated grid index is denoted as P .

To protect POI privacy, LBS needs to encrypt grid index P before outsourcing them to fog nodes. Assume that P is an index with 1024 cells, to encrypt all cells, a plain method is creating 1024 unique keys, and allocating them to each cell. However, as the growing of cell number, it incurs not only the space for key storage, but also heavy communication cost for transferring these

keys during online query. To cope with this problem, we propose a key generator which requires $\log_2 M$ pairs of key for grid with M cells. In line 2 (Alg.1), LBS generates l random pairs keys where $l = \lceil \log_2 M \rceil$. By using this strategy for key generation, only 20 ($2 \times \log_2 1024$) keys are needed in our example.

Algorithm 3: Oblivious Key Transfer

Input: LBS with key generator K , v with query index $I = i_1, i_2, \dots, i_l$
Output: v with unique key $K_I = K_1^{i_1}, K_2^{i_2}, \dots, K_l^{i_l}$
1: **for** $k = 1$ to l **do**
2: with i_k as input, v uses a 1-out-of-2 OT on (K_k^0, K_k^1) to obtain $K_k^{i_k}$
3: **end for**

To encrypt the content of each cell, the binary representation $(i_1, i_2, \dots, i_l)$ of cell index I is used as the selection bits to select l keys from the l pairs of keys. The exclusive-OR of the l selected keys, $\bigotimes_{k=1}^l K_k^{i_k}$, is given to a secure hash function H to produce an $d\alpha$ -bits pseudorandom string. The encryption of c is completed by computing the exclusive-OR of c and the pseudorandom string (in line 3 of Alg. 1). For example, when $I = 7$, the unique key $K_I = (K_1^0, K_2^0, \dots, K_7^0, K_8^1, K_9^1, K_{10}^1)$ which is selected from key generator K with $I = (0000000111)$. Then, with the content of the cell and the unique key as an input, the encryption function outputs the ciphertext corresponding to that cell. As we can see, with key generator K , the key size decreases from M to $2 \times \log_2 M$ for index with M cells, which can not only decrease the storage requirement for key, but also release the pressure during online query. After that, these encrypted cells are distributed to each fog nodes according their coverage region while key is kept by LBS secretly.

Next, we will explain how can vehicle retrieve k nearest POIs while keeping mutual privacy in this framework.

4.2. Online Query Processing

When vehicle enters a region, it queries the nearest fog node and LBS to retrieve k nearest POIs to its location q . If there is no enough qualified POIs supported by current fog node, it goes to upper level and queries the fog node with bigger

Figure 3: Distance between a point and a cell

coverage region. For a specific node, the communication between three parties is shown in Alg. 2. It is implemented based on best-first search strategy [29]. In this procedure, a priority queue T and a sorted list L are needed for storing the cells of grid P that will be explored during the search (only the cell index in T) and k nearest POIs to q that have been retrieved so far. Function $\max(T)$ outputs the max distance between q and cells in T , while $\min(L)$ returns the min distance between q and POIs in L .

Initially, all cells of P are inserted into T with their distances to q as priority, and L is set to be empty. After that, the top cell (i.e., the nearest cell to q) is removed from T and its index is fed into oblivious key transfer algorithm to obtain some appropriate keys that can be used to decrypt this cell. Holding the index I of this cell, vehicle can query LBS for l unique keys corresponding to the cell by running an obvious key transfer protocol (shown in Alg. 3), which is built on the oblivious transfer protocol presented in [20]. By XORing these l keys, the vehicle can generate the decryption key for c_I , and with the downloaded map from fog node, vehicle can perform decryption over c_I locally and get POIs located in the cell. Based on the decrypted POIs, the list L is updated with up-to-date k nearest POIs to q . This process continues until T is empty, or $\max(T)$ is larger than $\min(L)$, which means the unexplored POIs in T are all farther than the current POIs in L . It is remarkable that vehicle also goes to upper level when interested cell is not covered by current node.

For the distance measurement between query

q and cell c , we use the following equation:

$$d(q, c) = \sqrt{d_x^2 + d_y^2} \quad (1)$$

where

$$\begin{aligned} d_x &= \frac{1}{2} (|x - x_1| + |x - x_2| - |x_2 - x_1|) \\ d_y &= \frac{1}{2} (|y - y_1| + |y - y_2| - |y_2 - y_1|) \end{aligned}$$

and $(x, y), (x_1, y_1), (x_2, y_2)$ are the coordinates of q , the upper left vertex v_1 of c , the lower right vertex v_2 of c , respectively, as depicted in Fig. 3. It represents three cases. If q is located in c (e.g., the cell B2 in Fig. 3), we have $d(q, c) = 0$. For green cells say the cell A4, $d(q, c) = d_1$ is measured by the distance from the nearest edge of c to q . For yellow cells say the cell D2, $d(q, c) = d_2$ is measured by the distance from the nearest edge of c to q . Clearly, the distance between q and any point in A4 is larger than or equal to d_1 , and the distance between q and any point in D2 is larger than or equal to d_2 . That is why we can safely prune some unvisited cells in our solution.

4.3. Security Analysis

Before we get into the proof of our proposed protocol, we first present the security definition of OT. More detailed proof about OT can be found in paper [30].

Theorem 1. Chooser's security: *for any $c_1, c_2 \in 0, 1$ and for any adversary B' executing the sender's part, the views that B' sees in case she tries to obtain M_{c_1} and in case the chooser tries to obtain M_{c_1} are statistically indistinguishable given M_0 and M_1 .*

Theorem 2. Sender's security: *for every distribution on the inputs (M_0, M_1) and any adversarial A substituting the chooser, there exists a simulator A' that gets to play the chooser's role in the ideal model and the same a priori information about M_0 and M_1 as A , such that the outputs of A and A' are statistically indistinguishable given M_0 and M_1 .*

Next, we define the security of our proposed protocol and present the proof as following. As we stated in Section 3.2, we focus on the security analysis between vehicle and LBS because fog node .

Theorem 3. Vehicle's security: *Based on chooser's security, the privacy preserving kNN query hides the query q from LBS.*

Proof: We prove that there is a polynomial-time view simulator $Sim_{LBS}(Q)$ for LBS, which generates a distribution statistically close to what is viewed from LBS, $VIEW(Q)$, here $Q = \{q_1, q_2, \dots, q_n\}$ is n continuous queries from vehicle. For each $Sim_{LBS}(q_j)$, it works as follows. First, it randomly chooses a query $\bar{q}_j$ which also denoted as $\bar{i}_1, \bar{i}_2, \dots, \bar{i}_l$. It then executes OT protocol with $\bar{i}_k$ as input, during which the view of chooser is $VIEW(q'_j)$. According to theorem 1, $VIEW(q_j)$ and $VIEW(q'_j)$ are statistically indistinguishable from each other. Then, it is easy to conclude that $VIEW(Q)$ and $VIEW(Q')$ are statistically indistinguishable with each other.

Theorem 4. LBS's security: *Based on sender's security, the privacy preserving kNN query prevents the leakage of unauthorized POIs from vehicle.*

Proof: We define a polynomial-time view simulator $Sim_v(Key)$ for vehicle v , which generates a distribution statistically close to what is viewed from v , $VIEW(Key) = VIEW(Key_1, Key_2, \dots, Key_n)$ where $VIEW(Key_j) = (K_1^{i_1} r_1, K_2^{i_2} r_2, \dots, K_l^{i_l} r_l)$, here Key_j is the original key pairs generated by LBS. $Sim_v(Key)$ works as follows. First, it randomly chooses l pairs of key which is Key' . Then, for each $Sim_v(Key_j)$, it randomly chooses l random number $r' = (\bar{r}_1, \bar{r}_2, \dots, \bar{r}_l)$ and executes OT protocol on Key' and r' , during which the view of vehicle is $VIEW(Key'_j) = (K_1^{i_1} \bar{r}_1, K_2^{i_2} \bar{r}_2, \dots, K_l^{i_l} \bar{r}_l)$. According to theorem 2, $VIEW(Key'_j)$ and $VIEW(Key_j)$ are statistically indistinguishable with each other. And for continuous communication, r' is indistinguishable from each other, which also means the $VIEW(Key)$ and $VIEW(Key')$ are statistically indistinguishable.

Algorithm 4: Privacy Preserving k NN Query with Type - offline index building

Input: LBS with POI database
Output: LBS with encryption key, Fog nodes with outsourced POIs

- 1: LBS generate the public parameters PK and a master key MK according to CP-ABE
- 2: LBS encrypts each POI with $E(m_{x,y,T}) \leftarrow EN_{cp}(PK, m_{x,y,T}, A)$, where access structure $A = (t_1 \vee t_2 \vee \dots \vee t_l)$
- 3: LBS builds a grid index P with $M = m \times m$ cells over the whole map
- 4: LBS stores $c = E_1(m_{x,y,T_1})|E_2(m_{x,y,T_2})|\dots|E_d(m_{x,y,T_t})$ for each cell
- 5: LBS generates encryption key K , which is composed of $l = \log_2 M$ random pairs of keys $(K_1^0, K_1^1), (K_2^0, K_2^1), \dots, (K_l^0, K_l^1)$
- 6: For each cell I , LBS selects a unique key from $K_I = K_1^{i_1}, K_2^{i_2}, \dots, K_l^{i_l}$ with binary representation of $I = (i_1, i_2, \dots, i_l)$
- 7: For each cell, LBS encrypts the content c by $E(c) \leftarrow H(g^{\prod_{k=1}^l K_k^{i_k}}, d\alpha) \otimes c$
- 8: LBS distributes encrypted cells to fog nodes according to their coverage region

5. Privacy-Preserving k NN Query with Type

In this section, the type of POI is considered. The provided solution can support the most complicated scenario where POI with one or more types for both POI database and k NN query are considered. Now each POI data is represented by a tuple (x, y, T) , where $T = (t_1, t_2, \dots, t_l)$ is a set of different types related to each POI. Examples of type t_i include shopping mall, school, restaurant, bank and so on. For any vehicle with query $= (x, y, QT)$, this query returns k nearest POIs with type attribute specified in QT which is the subset of returned POI type set T .

5.1. System Initialization

Here, we provide a solution with a little adaptation on k NN query. First, before dividing all POIs into M cells, we encrypt each POI with type attributes specified in T (line 2 in Alg. 4) and require that only the subset of T can decrypt this POI. To achieve this, we adopt ciphertext-policy attribute-based encryption (CP-ABE)[21] for encryption. In CP-ABE, a user will be able to decrypt a ciphertext, if and only if his attribute satisfies the policy of the respective ciphertext. For instance, a ciphertext is encrypted with respect

Algorithm 5: Privacy Preserving k NN Query with Type - online query processing

Input: LBS with encryption key, Fog node with encrypted map, v with $q = (x, y, QT)$
Output: v gets k nearest POIs to q with type QT

- 1: v obtain SK from LBS with QT as input (see Alg. 6)
- 2: v downloads encrypted map from fog node
- 3: v inserts all cells of P to a priority queue T
- 4: **while** T is not empty and $\max(T) < \min(L)$ **do**
- 5: $I = \text{Dequeue}(T)$
- 6: LBS chooses random elements $r_1, r_2, \dots, r_l$ and computes $g^{1/\prod_{k=1}^l r_k}$
- 7: LBS feeds Alg. 3 with input $(K_1^0 r_1, K_1^1 r_1), (K_2^0 r_2, K_2^1 r_2), \dots, (K_l^0 r_l, K_l^1 r_l)$
- 8: v obtains l keys $K_1^{i_1} r_1, K_2^{i_2} r_2, \dots, K_l^{i_l} r_l$ by feeding I into Alg. 3 also
- 9: v obtains the decryption key for c_I by $g^{\prod_{k=1}^l K_k^{i_k}} = (g^{1/\prod_{k=1}^l r_k})^{\prod_{k=1}^l K_k^{i_k} r_k}$
- 10: v obtains encrypted POIs $(E(m_{x,y,T}))$ by decrypting $E(c_I)$, which is, computing $H(g^{\prod_{k=1}^l K_k^{i_k}}, k\alpha) \otimes E(c_I)$
- 11: v decrypts each POI $(E(m_{x,y,T}))$ with SK
- 12: v updates L using newly retrieved POIs
- 13: **end while**
- 14: v takes POIs in L as the query result

Algorithm 6: Privacy Preserving SK Generation

Input: LBS with master key MK , v with query type $QT = t_1, t_2, \dots, t_l$
Output: v with SK

- 1: v generates l random r_j for each attribute
- 2: LBS generates a random r and l random r'_j for each attribute
- 3: LBS computes $D = g^{(\alpha+r)/\beta}$ and send to v
- 4: **for** $j = 1$ to l **do**
- 5: v computes $H(t_j)^{r_j}$ and sends to LBS
- 6: LBS computes $D_j = g^r \cdot (H(t_j)^{r_j})^{r'_j}$, $D'_j = (g^{r_j})^{r'_j}$
- 7: **end for**
- 8: LBS sends all D_j and D'_j to v
- 9: v constructs $SK = (D, D_j, D'_j) \text{ for } 1 \leq j \leq l$

to the policy $A \vee B \vee C$, then user with a key to attribute A is able to decrypt. Even though key-policy attribute-based encryption (KP-ABE)[31] has the same property and produces smaller ciphertext, it cannot guarantee a secure way for secret key generation which is also important in our problem.

In conclusion, the system initialization of T- k NN works as following. With the public parameters PK generated according to CP-ABE, LBS provider encrypts each POI with type attributes specified in T as the access structure. Thus, each POI is encrypted into a unique ciphertext and only secret key SK with respect to this access structure can decrypt the POI data cor-

rectly. Then same to the k NN query in Alg. 1, LBS provider creates and encrypts the grid index over the whole region, but each POI is represented as an ciphertext now.

5.2. Online Query Processing

It follows the best-first strategy described in Alg. 2. Each time, priority queue pops up a cell index from the candidate, with this index and key transfer algorithm (see Alg. 3), vehicle can successfully retrieve the corresponding d POIs within the cell. But it is noteworthy that the message we stored in each cell of P is the encrypted. So, what user get with cell index is d encrypted POIs. To show POIs in plaintext, we only need to get the secret key (SK) with respect to query type QT . The generated SK can decrypt POIs only when QT is a subset of POI type T .

However, the generation of SK is highly dependent on query type QT and master key MK which is hold by vehicle and the LBS provider separately. In addition, SK should only be exposed to vehicle at the end. By investigating into the detail of CP-ABE, we provide a privacy-preserving SK generation algorithm (Alg. 6). Instead of sending plaintext of type query QT , we disguise QT with hash function and random value r_j (line 5 in Alg. 6). SK can be further generated according to this disguised QT and returned back to vehicle. With retrieved SK , it is easy for user to decrypt ($E(m_{x,y,T})$) (in line 10 of Alg. 5). We apply the decryption function on each POI in c_i , only the POI containing query type QT can be decrypted. The whole process repeats until k nearest POIs are collected.

5.2.1. Security Analysis

Recall that T- k NN query is composed of cell key retrieval (oblivious key transfer) and type key retrieval (privacy preserving SK generation). Since we have proved the security of vehicle and LBS during oblivious key transfer (theorem 3 and 4), we only need to show the security under SK generation. Before we get into the detail, we first give the definition of Decisional Diffie-Hellman (DDH) assumption which is the theoretical foundation for LBS security later.

Theorem 5. DDH assumption: For a cyclic group G with generator g , given $(g^a, g^b, g^{ab}) \in G$ and $(g^a, g^b, g^c) \in G$, g^{ab} and g^c are computationally indistinguishable.

Theorem 6. Vehicle's security: With randomness of r , T- k NN query hides type QT from LBS provider.

Proof: We assume that there is a polynomial-time view simulator $Sim_{LBS}(QT)$ for server LBS, which generates a distribution statistically close to what is viewed from LBS, which is $VIEW(QT) = (H(t_1)^{r_1}, H(t_2)^{r_2}, \dots, H(t_l)^{r_l})$. For $Sim_{LBS}(QT)$, it randomly chooses a query QT which also denoted as $t'_1, t'_2, \dots, t'_l$ and random $R = (r'_1, r'_2, \dots, r'_l)$. It then computes $H(t'_j)^{r'_j}$ accordingly, during which the view is $VIEW(QT') = (H(t'_1)^{r'_1}, H(t'_2)^{r'_2}, \dots, H(t'_l)^{r'_l})$. As long as the security of hash function H and randomness of R , $VIEW(QT)$ and $VIEW(QT')$ are indistinguishable from each other.

Theorem 7. LBS's security: Based on hard problem of the DDH, the T- k NN query prevents the leakage of unauthorized POIs from vehicle.

Proof: Similarly, we define a polynomial-time view simulator $Sim_v(MK)$ for vehicle v , which generates a distribution statistically close to what is viewed from v , $VIEW(MK) = SK = (D, D_j, D'_j)$. $Sim_v(MK)$ works as follows. It randomly generates $MK' = (\beta, g^\alpha)$, and computes D' , D'_j and D''_j for according to query type (line 3 and 6 in Alg. 6). The view during this process is $VIEW(MK') = (D', D'_j, D''_j)$. Based on DDH assumption, it is easy to prove that $VIEW(MK)$ is computationally indistinguishable from $VIEW(MK')$ which also means the privacy of LBS.

6. Performance Evaluation

6.1. Complexity Analysis

This section discusses query latency in our proposed scheme and gives a comparison with two state-of-the-art works presented by Paulet et al. [17] and Yi et al. [15]. Query latency is the time

Table 1: Comparison of Complexity

	k NN Query			T- k NN Query		
	Comp.		Comm.	Comp.		Comm.
	client	server		client	server	
Ours	$O(\log_2 m)$	$O(\log_2 m)$	$6\log_2 m + km^2/n + 2$	$O(\log_2 m + l)$	$O(\log_2 m + 2l)$	$6\log_2 m + km^2/n + 3 + 3l$
Yi	$O(m)$	$O(m^2)$	$2m$	$O(m + t)$	$O(tm^2)$	$2m + t$
Paulet	$O(m^2)$	$O(m)$	$7 + 8m$	-

Figure 4: Latency comparison of k NN Figure 5: Latency comparison of T- k NN

interval from a vehicle sending a query to receiving the feedback, which contains the online computing delay and communication delay. In online computation 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.

To simplify the comparison resulting from different settings, we unify the problem into a setting where the whole map is divided into $M = m \times m$ cells, with k nearest POIs to the cell center. Also, for our scheme, we suppose that there are n fog nodes with cells evenly deployed on them. The performance analysis is based on average of several retrieval result. Besides, we set the bit of all the items transferred online to 1024, this is reasonable, since no matter the POI or group member used for encryption, the length required for security consideration is normally equal or smaller than 1024. The total type is denoted as t and the number of type in the query is denoted as l . The complexity comparison result is presented in Table 1.

6.1.1. Complexity Analysis of k NN

We start from the computational costs analysis of k NN query from both three papers, which are denoted as Ours, Yi and Paulet respectively.

Recall that in our scheme, everytime when vehicle enters a region, it downloads the encrypted map from nearest node and retrieves the key back from LBS to decrypt them to get k nearest POIs. Thus, the complexity analysis is composed of two parts, map downloading and key retrieval. For map downloading which involves only map size, the communication cost is $k \times m^2/n$ with m^2/n cells and k POIs in each cell. For the latency from key retrieval, it results from both computation and communication cost. As one OT requires 2 exp for chooser(client) and 1 exp for sender(server), thus, plus the exp from online query processing, it is easy to conclude that for transferring $2\log_2 m$ (enough for M cells in the grid) key, it requires $4\log_2 m + 1$ exp for client and $2\log_2 m + 1$ exp for server. The communication overheads during this process is $6\log_2 m + 2$ group members with 1024 bits for each.

Same to the grid index adopted here, Yi im-

Figure 6: Computation cost of k NN

Figure 7: Communication cost of k NN

plemented k NN query by dividing the map into cells and retrieving cell with k nearest POIs to the cell center (Approximate k NN) with Paillier and Rabin encryption. For each query, it requires $m + 4$ exp for client and $2m + m^2$ exp for server. In addition, the communication overheads is $2m$ items with 1024 bits for each.

Paulet involved two stages: retrieving encryption key and encrypted cells, which are achieved by OT and PIR respectively. According to the paper, the computational cost is $7 + 2m^2$ for client and $6m + 1$ for server, with $7 + 8m$ items over communication (1024 bits each).

6.1.2. Complexity Analysis of T - k NN

Since T - k NN query is not supported in Paulet, we compare our protocol with Yi.

In T - k NN, we encrypt each POI with CP-ABE and the key is generated according to POI types, which promise only query satisfies the type requirement can decrypt the POIs. Therefore, except the cost in k NN, T - k NN also introduces the cost for obtaining SK from server, which includes $l + 1$ exp for client, $2l$ exp for server and $3l + 1$ items for communication, where l is the number of type in the query.

Yi also extended their work based on k NN by

storing k nearest neighbour of each type into each cell, resulting in extra cost linearly to the number of type t . Specifically, it includes $t + 3$ exp for client, tM exp for server and t items for communication cost where t is number of type in whole dataset.

6.2. Experimental Evaluation

The experiment is taken under a real-world dataset which is collected in Beijing, China. The information of this dataset is presented in Table 2. We crawl the data with baidu API. The whole dataset is a $33 * 35km^2$ region with 160000 POIs. This area is the most important city center of Beijing. All experiments are performed on a PC with 2.19GHz CPU, 4GB RAM, JDK 8 and Win 10. In this experiment, we present the query latency of each method under varying grid granularity m . To have a better analysis of trend, we also provide detailed computation (red line chart) and communication cost (blue histogram) of them.

Table 2: Summary of Dataset

MAXLON	116.603738	MINLON	116.200006
MAXLAT	40.073334	MINLAT	39.780002
NumType	20	NumPOI	160000

Figure 8: Computation cost of T- k NN

Figure 9: Communication cost of T- k NN

6.2.1. Analysis of k NN

According to our complexity analysis, the grid granularity m ($M = m^2$) is the main factor affecting the performance. Therefore, we evaluate three methods under parameter m varying from 40 to 400 with a step of 40, while other parameters are set to a default value. The max number of POIs stored in each cell, d , is set to 25, the bit length of POI, α , is 64, the key size of pseudo-random function H is set to 80 and the number of nearest neighbour k is 5. Since the number of POI stored in each cell is equal or larger than k , every method can retrieve k NN with one transfer.

Fig. 4 shows the query latency of each method under varying m . Fig. 6 and Fig. 7 are detailed computation and communication cost of them respectively. Clearly, our approach outperforms the other two significantly. Although the increase of m , our approach still keeps a quick feedback under seconds with the communication cost under 0.15 MB. In comparison, even Yi and Paulet also have a quick feedback at the beginning, they increase rapidly while the increasing of m . When $m = 400$ which keeps each cell in a 100×100 area, the time for Yi increases to 800s and Paulet increases to 350000s, which is obviously prohibitive in IoV. The efficiency of our approach is the result of key

generator strategy, which transferred most of the computation to the offline phase, thus, during the online query processing, our scheme works only on $\log_2 M$ pairs of key generators, instead of querying on the whole database as the solution provided in Yi and Paulet.

6.2.2. Analysis of T- k NN

The latency comparison of T- k NN is presented in Fig. 5 with detail information presented in Fig. 8 and Fig. 9 respectively. The latency comparison is conducted under varying m with $l = 5, t = 20$. Obviously, compared to our scheme, Yi still has the limitation of scalability. Since the server of Yi needs to iterate over the whole database M , the time cost goes parabola with m ($M = m \times m$). An interesting finding here is that our approach with varying m has the similar performance to k NN. As we known, the cost of our scheme is introduced by the oblivious key transfer and SK generation of CP-ABE. However, the cost over CP-ABE which runs only one time during the whole processing is much smaller than the oblivious key transfer. Therefore, k NN and T- k NN showed a similar performance under fixed l . Besides, we also evaluate the performance of our protocol under varying l and Yi under varying t with fixed $m = 200$. Both

performance goes linearly with varying parameter just as we analyzed in Table 1.

6.2.3. Discussion

In short, our schemes are indeed efficient in terms of low-latency, but it is also remarkable that they still have some limitations. In general, we summarise it into three folds. First, the proposed schemes still have some limitation in practical deployment since it may not be that easy to realize full coverage of the whole map, uncovered region may occur sometimes. Possible solutions may be incorporation of edge computing with crowdsourcing and other technologies [32]; Second, the overhead of storage and communication is still a problem. Apart from the solutions provided above, we can also use some data compression technologies or effective data structures to reduce the cost; Third, more fine-grained LBS queries should be supported, such as searching the nearest restaurant that opens in the afternoon and satisfies the five-star service score or have good sales profile, which make the LBS query more useful.

7. Conclusion

In this paper, based on fog computing enabled LBS framework, we proposed two low-latency LBS query schemes (k NN and T- k NN) to support mobile query in IoV. Security analysis shows that both schemes achieve mutual privacy with respect to the vehicles and the LBS provider. Through an empirical study based on a real world dataset, we show that our schemes are more efficient than the state-of-the-art works in terms of low latency. The further work will concentrate on the improvement of current work as we discussed above.

References

- [1] Y. Fangchun, W. Shangguang, L. Jinglin, L. Zhihan, S. Qibo, An overview of internet of vehicles, *China Communications* 11 (10) (2014) 1–15.
- [2] Z. Yan, P. Zhang, A. V. Vasilakos, A survey on trust management for internet of things, *Journal of network and computer applications* 42 (2014) 120–134.
- [3] J. Kang, R. Yu, X. Huang, M. Jonsson, H. Bogucka, S. Gjessing, Y. Zhang, Location privacy attacks and defenses in cloud-enabled internet of vehicles, *IEEE Wireless Communications* 23 (5) (2016) 52–59.
- [4] J. Shao, R. Lu, X. Lin, Fine: A fine-grained privacy-preserving location-based service framework for mobile devices, in: *INFOCOM, 2014 Proceedings IEEE*, IEEE, 2014, pp. 244–252.
- [5] X. Yi, R. Paulet, E. Bertino, V. Varadharajan, et al., Practical approximate k nearest neighbor queries with location and query privacy., *IEEE Trans. Knowl. Data Eng.* 28 (6) (2016) 1546–1559.
- [6] C. Bettini, X. S. Wang, S. Jajodia, Protecting privacy against location-based personal identification., in: *Vldb International Conference on Secure Data Management*, 2005, pp. 185–199.
- [7] B. Gedik, L. Liu, Location privacy in mobile systems: A personalized anonymization model, in: *IEEE International Conference on Distributed Computing Systems*, 2005. *ICDCS 2005. Proceedings*, 2005, pp. 620–629.
- [8] M. F. Mokbel, C. Y. Chow, W. G. Aref, The new casper: query processing for location services without compromising privacy, in: *International Conference on Very Large Data Bases*, 2006, pp. 763–774.
- [9] B. Bamba, L. Liu, P. Pesti, T. Wang, Supporting anonymous location queries in mobile environments with privacygrid, in: *International Conference on World Wide Web*, 2008, pp. 237–246.
- [10] C. Y. Chow, M. F. Mokbel, X. Liu, A peer-to-peer spatial cloaking algorithm for anonymous location-based service, in: *ACM International Symposium on Geographic Information Systems, Acm-Gis 2006*, November 10–11, 2006, Arlington, Virginia, Usa, *Proceedings*, 2006, pp. 171–178.
- [11] H. Kido, Y. Yanagisawa, T. Satoh, An anonymous communication technique using dummies for location-based services, in: *Icps '05. Proceedings. International Conference on Pervasive Services*, 2005, pp. 88–97.
- [12] L. Y. Man, C. S. Jensen, X. Huang, H. Lu, Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services, in: *IEEE International Conference on Data Engineering*, 2008, pp. 366–375.
- [13] J. Krumm, A survey of computational location privacy, *Personal and Ubiquitous Computing* 13 (6) (2009) 391–399.
- [14] S. Liu, A. Liu, L. Zhao, G. Liu, Z. Li, P. Zhao, K. Zheng, L. Qin, Efficient query processing with mutual privacy protection for location-based services, in: *International Conference on Database Systems for Advanced Applications*, Springer, 2016, pp. 299–313.
- [15] X. Yi, R. Paulet, E. Bertino, V. Varadharajan, Practical k nearest neighbor queries with location privacy, in: *2014 IEEE 30th International Conference on Data*

- Engineering, IEEE, 2014, pp. 640–651.
- [16] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, K. L. Tan, Private queries in location based services: anonymizers are not necessary, in: ACM SIGMOD International Conference on Management of Data, 2008, pp. 121–132.
 - [17] R. Paulet, M. G. Kaosar, X. Yi, E. Bertino, Privacy-preserving and content-protecting location based queries, *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 26 (5) (2014) 1200–1210.
 - [18] M.-K. Jiau, S.-C. Huang, J.-N. Hwang, A. V. Vasilakos, Multimedia services in cloud-based vehicular networks, *IEEE Intelligent Transportation Systems Magazine* 7 (3) (2015) 62–79.
 - [19] X. Yang, F. Yin, X. Tang, A fine-grained and privacy-preserving query scheme for fog computing-enhanced location-based service, *Sensors* 17 (7) (2017) 1611.
 - [20] M. Naor, B. Pinkas, Oblivious transfer with adaptive queries, *Lecture Notes in Computer Science* 1666 (1999) 573–590.
 - [21] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute-based encryption, in: *Security and Privacy, 2007. SP'07. IEEE Symposium on*, IEEE, 2007, pp. 321–334.
 - [22] E. Kushilevitz, R. Ostrovsky, Replication is not needed: single database, computationally-private information retrieval, in: *Foundations of Computer Science, 1997. Proceedings., Symposium on*, 2002, pp. 364–373.
 - [23] P. Paillier, Public-key cryptosystems based on composite degree residuosity classes, in: *International Conference on Theory and Application of Cryptographic Techniques*, 1999, pp. 223–238.
 - [24] M. O. Rabin, Digitalized signatures and public-key functions as intractable as factorization, *Massachusetts Institute of Technology*, 1979.
 - [25] O. Osanaiye, S. Chen, Z. Yan, R. Lu, K.-K. R. Choo, M. Dlodlo, From cloud to fog computing: A review and a conceptual live vm migration framework, *IEEE Access* 5 (2017) 8284–8300.
 - [26] C. Huang, M. Ma, Y. Liu, A. Liu, Preserving source location privacy for energy harvesting wsns, *Sensors* 17 (4) (2017) 724.
 - [27] L. Wang, G. Liu, L. Sun, A secure and privacy-preserving navigation scheme using spatial crowdsourcing in fog-based vanets, *Sensors* 17 (4) (2017) 668.
 - [28] C. Hazay, Y. Lindell, *Efficient secure two-party protocols: Techniques and constructions*, Springer Science Business Media, 2010.
 - [29] G. R. Hjaltason, H. Samet, Distance browsing in spatial databases, *ACM Transactions on Database Systems (TODS)* 24 (2) (1999) 265–318.
 - [30] M. Naor, B. Pinkas, Efficient oblivious transfer protocols, in: *Proceedings of the Twelfth Annual Symposium on Discrete Algorithms*, January 7-9, 2001, Washington, DC, USA., 2001, pp. 448–457.
 - [31] V. Goyal, O. Pandey, A. Sahai, 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*, Acm, 2006, pp. 89–98.
 - [32] W. Feng, Z. Yan, H. Zhang, K. Zeng, Y. Xiao, Y. T. Hou, A survey on security, privacy, and trust in mobile crowdsourcing, *IEEE Internet of Things Journal* 5 (4) (2018) 2971–2992.