EPLQ: Efficient Privacy-Preserving Location-based Query over Outsourced Encrypted Data

Lichun Li, Rongxing Lu, *Senior Member, IEEE* and Cheng Huang

***Abstract*—With the pervasiveness of smart phones, location-based services (LBS) have received considerable attention and become more popular and vital recently. However, the use of LBS also poses a potential threat to user's location privacy. In this paper, aiming at spatial range query, a popular LBS providing information about POIs (Points Of Interest) within a given distance, we present an efficient and privacy-preserving location based query solution, called EPLQ. Specifically, to achieve privacy-preserving spatial range query, we propose the first predicate-only encryption scheme for inner product range, which can be used to detect whether a position is within a given circular area in a privacy-preserving way. To reduce query latency, we further design a privacy-preserving tree index structure in EPLQ. Detailed security analysis confirms the security properties of EPLQ. In addition, extensive experiments are conducted, and the results demonstrate that EPLQ is very efficient in privacy-preserving spatial range query over outsourced encrypted data. In particular, for a mobile LBS user using an Android phone, around 0.9 second is needed to generate a query; and it also only requires a commodity workstation, which plays the role of the cloud in our experiments, a few seconds to search POIs.**

***Index Terms*—Location-based Services, Privacy-Enhancing Technology, Spatial Range Query, Outsourced Encrypted Data.**

I. Introduction

A few decades ago, location-based services (LBS) were used in military only. Today, thanks to advances in information and communication technologies, more kinds of LBS have appeared, and they are very useful for not only organizations but also individuals. Let's take the spatial range query, one kind of LBS that we will focus on in this paper, as an example. Spatial range query is a widely used LBS, which allows a user to find POIs (Point Of Interests) within a given distance to his/her location, i.e., the query point. As illustrated in Fig. 1, with this kind of LBS, a user could obtain the records of all restaurants within walking distance (say 500 meters). Then the user can go through these records to find a desirable restaurant considering price and reviews.

While location-based services are popular and vital, most of these services today including spatial range query require users to submit their locations, which raises serious concerns about the leaking and misusing of user location data. For example, criminals may utilize the data to track potential victims and predict their locations. For another example, some sensitive location data of organization users may involve trade secret or national security. Protecting the privacy of user location in LBS has attracted considerable interest. However,

L. Li, R. Lu and C. Huang are with School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. 639798. E-mail: lcli@ntu.edu.sg; rxlu@ntu.edu.sg; huangcheng@ntu.edu.sg.

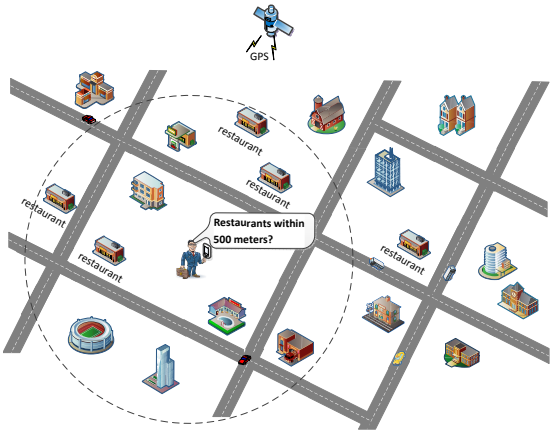


Fig. 1: An example of spatial range query

significant challenges still remain in the design of privacy-preserving LBS, and new challenges arise particularly due to data outsourcing. In recent years, there is a growing trend of outsourcing data including LBS data because of its financial and operational benefits. Lying at the intersection of mobile computing and cloud computing, designing privacy-preserving outsourced spatial range query faces the challenges below:

- *Challenge on querying encrypted LBS data*. The LBS provider is not willing to disclose its valuable LBS data to the cloud. As illustrated in Fig. 2, the LBS provider encrypts and outsources private LBS data to the cloud, and LBS users query the encrypted data in the cloud. As a result, querying encrypted LBS data without privacy breach is a big challenge, and we need to protect not only the user locations from the LBS provider and cloud, but also LBS data from the cloud.
- *Challenge on the resource consumption in mobile devices*. Many LBS users are mobile users, and their terminals are smart phones with very limited resources. However, the cryptographic or privacy-enhancing techniques used to realize privacy-preserving query usually result in high computational cost and/or storage cost at user side.
- *Challenge on the efficiency of POI searching*. Spatial range query is an online service, and LBS users are sensitive to query latency. To provide good user experiences, the POI search performing at the cloud side must be done in a short time (e.g. a few seconds at most). Again, the techniques used to realize privacy-preserving query usually increase the search latency.
- *Challenge on security*. LBS data are about POIs in real world. It is reasonable to assume that the attacker

may have some knowledge about original LBS data. With such knowledge, known-sample attacks are possible (elaborated later in Section II).

Recently, there are already some solutions for privacy-preserving spatial range query [1]–[6]. However, as elaborated in Section VIII later, existing solutions cannot address all above challenges. Aiming at these, in this paper, we propose an efficient solution for privacy-preserving spatial range query named EPLQ, which allows queries over encrypted LBS data without disclosing user locations to the cloud or LBS provider. To protect the privacy of user location in EPLQ, we design a novel predicate-only encryption scheme for inner product range (IPRE scheme for short), which, to the best of our knowledge, is the first predicate/predicate-only scheme of this kind. To improve the performance, we also design a privacy-preserving index structure named $\hat{ss}$-tree. Specifically, the main contributions of this paper are three folds:

- First, we propose a novel predicate-only encryption scheme for inner product range named IPRE, which allows testing whether the inner product of two vectors is within a given range without disclosing the vectors. In *predicate encryption*, the key corresponding to a predicate, f, can decrypt a ciphertext if and only if the attribute of the ciphertext, x, satisfies the predicate, i.e. $f(x) = 1$. *Predicate-only encryption* is a special type of predicate encryption not designed for encrypting/decrypting messages. Instead, it reveals that whether $f(x) = 1$ or not. Predicate-only encryption schemes supporting different types of predicates [7], [8] have been proposed for privacy-preserving query on outsourced data. To the best our knowledge, there does not exist predicate/predicate-only scheme supporting inner product range. Though our scheme is used for privacy-preserving spatial range query in this paper, it may be applied in other applications as well.
- Second, we propose EPLQ, an efficient solution for privacy-preserving spatial range query. In particular, we show that whether a POI matches a spatial range query or not can be tested by examining whether the inner product of two vectors is in a given range. The two vectors contain the location information of the POI and the query respectively. Based on this discovery and our IPRE scheme, spatial range query without leaking location information can be achieved. To avoid scanning all POIs to find matched POIs, we further exploit a novel index structure named $\hat{ss}$-tree, which conceals sensitive location information with our IPRE scheme. Experiments on our implementation demonstrate that our solution is very efficient. Moreover, security analysis shows that EPLQ is secure under known-sample attacks and ciphertext-only attacks.
- Third, our techniques can be used for more kinds of privacy-preserving queries over outsourced data. In the spatial range query discussed in this work, we consider Euclidean distance, which is widely used in spatial databases. Our IPRE scheme and $\hat{ss}$-tree may be used for searching records within a given weighted Euclidean distance or great-circle distance as well. Weighted Euclidean distance is used to measure the dissimilarity in many kinds of data, while great-circle distance is the distance of two points on the surface of a sphere. Using great-circle distance instead of Euclidean distance for long distances on the surface of earth is more accurate. By supporting these two kinds of distances, privacy-preserving similarity query and long spatial range query can also be realized.

The remainder of this paper is organized as follows. In Section II, we formalize the system model and attack models considered in our work, and identify the design goal. In Section III, we recall bilinear pairing and related complexity assumptions as preliminaries, which will be used in subsequent sections. In Section IV, we propose IPRE, a predicate-only encryption scheme for inner product range. Then, based on IPRE, we design the EPLQ solution for privacy-preserving spatial range query in Section V, followed by its security analysis and performance evaluation in Section VI and Section VII, respectively. We give related work in Section VIII, and finally conclude this work in Section IX.

II. Models and Design Goal

In this section, we formalize the system model and attack models considered in this paper, and identify the design goal.

A. System Model

Privacy-preserving POI query has been studied in two settings of LBS: public LBS and outsourced LBS. In this paper, we focus on the latter setting. In the former setting, there is an LBS provider holding a spatial database of POI records in plaintext, and LBS users query POIs at the provider's site. In outsourced LBS, as shown in Fig. 2, the system consists of three kinds of entities: LBS provider, LBS users and cloud.

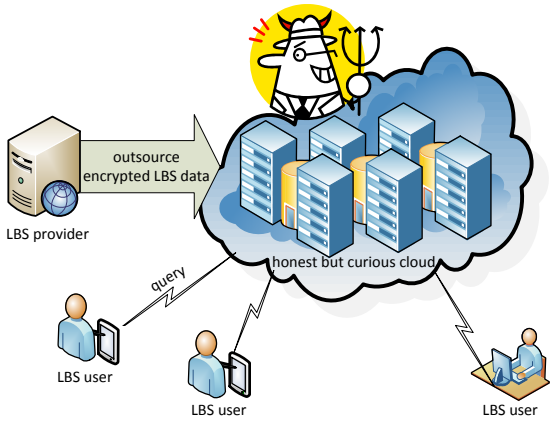


Fig. 2: System model of outsourced LBS under consideration

- The LBS provider has abundant of LBS data, which are POI records. The LBS provider allows authorized users (i.e. LBS users) to utilize its data through location-based queries. Because of the financial and operational benefits of data outsourcing, the LBS provider offers the query services via the cloud. However, the LBS provider is not willing to disclose the valuable LBS data to the cloud.

Therefore, the LBS provider encrypts the LBS data, and outsources the encrypted data to the cloud.
- The cloud has rich storage and computing resources. It stores the encrypted LBS data from the LBS provider, and provides query services for LBS users. So the cloud has to search the encrypted POI records in local storage to find the ones matching the queries from LBS users.
- LBS users have the information of their own locations, and query the encrypted records of nearby POIs in the cloud. Cryptographic or privacy-enhancing techniques are usually utilized to hide the location information in the queries sent to the cloud. To decrypt the encrypted records received from the cloud, LBS users need to obtain the decryption key from the LBS provider in advance.

B. Attack Models

Similar as most previous works on outsourced data query, the cloud is assumed *honest but curious* and considered as the potential attacker in this work. That is, the cloud would honestly store and search data as requested, however the cloud would also have financial incentives to learn those stored LBS data and user location data in query. Because both LBS data and user location data are valuable, they should be protected and hidden from the cloud. In general, in the outsourced LBS setting, the cloud can observe both queries from LBS users and encrypted LBS data from the LBS provider, which could be an advantage to learn user locations. Therefore, assuming different abilities of the attacker, there are mainly four attack models in oursourced LBS setting.

- *Ciphertext-only attack*. In this model, the attacker is able to observe the ciphertexts of POIs' locations and queries, but does not know the plaintexts. Obviously, every cloud has this ability. This is a weak attack model.
- *Known-sample attack*. In this model, the attacker knows the plaintexts of some POIs' locations and/or queries. The attacker also knows that their corresponding ciphertexts must exist in all the ciphertexts observed by the attacker. However, the attacker does not know which ciphertext is corresponding to a known plaintext. Utilizing such information, the attacker may be able to reveal the plaintext corresponded to any given ciphertext. Such information is not hard to obtain if the attacker has the background knowledge that the LBS database must contain the POIs of certain type in a certain area.
- *Known-plaintext attack*. In this model, the attacker knows the plaintexts of some POIs' locations and/or queries as well as their corresponding ciphertexts. Utilizing this information, the attacker may be able to reveal the plaintexts corresponded to other ciphertexts.
- *Access-pattern attack*. In this model, the attacker has some background knowledge about the pattern of POI accessing. For example, the attacker knows that a known POI would be the most popular POI. If an encrypted POI appears most frequently in query results, it must be the encrypted version of the known POI. Then the attacker knows that corresponding query points must be close to the known POI.

In addition to the above attacks, other attacks such as insider attacks may be possible. In this paper, we consider ciphertext-only and known-sample attacks, which do not require attackers with very strong abilities. We will leave the attacks requiring very strong abilities for future study.

C. Design Goal

Under the outsourced LBS system model, our design goal is to develop an efficient, accurate and secure solution for privacy-preserving spatial range query. Specifically, the following three objectives should be achieved:

- *Efficiency*. As discussed in Section I, spatial range query has stringent performance requirements. A good solution should not consume many resources of mobile LBS users, and the POI search latency should be acceptable for online query.
- *Accuracy*. It's desirable that a query result contains exact the records matching the query. False negatives would hurt user experience, while false positives would increase communication cost. Additional computational cost is also required at the user side to filter out false positives.
- *Security*. The proposed solution should be resilient to ciphertext-only attacks and known-sample attacks. An accurate and efficient solution for spatial range query [1] already exists, which is resilient to ciphertext-only attacks but not to known-sample attacks and more powerful attacks. The proposed solution should be more secure than the solution in [1].

Though subject to more powerful attacks such as known-plaintext attacks, the solution proposed in this paper still can be used in many situations where the attackers do not have the required abilities or knowledge. Our solution also has advantages over the solutions resilient to such attacks. As we will see in the related works in Section VIII, such solutions are either very computationally costly or not applicable to outsourced LBS.

III. Preliminaries: Bilinear Pairing and Complexity Assumptions

In this section, we outline the cryptographic technique of bilinear pairing and related complexity assumptions, which will serve as the basis of our IPRE scheme.

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups of the same big prime order p, and g be a generator of $\mathbb{G}_1$. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a pairing, i.e. a map satisfies the following properties:

- Bilinearity. $e(P^a, Q^b) = e(P, Q)^{ab}$ for any $a, b \in \mathbb{Z}_p^*$ and any $P, Q \in \mathbb{G}_1$
- Non-degeneracy. $e(g, g) \neq 1$
- Computability. e can be computed efficiently.

The definitions of pairing parameter generator and pairing related complexity assumptions are given below. For more comprehensive descriptions, please refer to [9].

Definition 1: $\mathcal{G}en$. The pairing parameter generator $\mathcal{G}en$ is a probabilistic algorithm that takes a security parameter λ as input and outputs a 5-tuple $(\mathbb{G}_1, \mathbb{G}_2, g, p, e)$.

Definition 2: *Computational Diffie-Hellman (CDH) problem*. The CDH problem is: Given $(P, P^a, P^b) \in \mathbb{G}_1$ for unknown $a, b \in Z_p^*$, compute $P^{ab} \in \mathbb{G}_1$.

Definition 3: *Decisional Bilinear Diffie-Hellman (DBDH) problem*. The DBDH problem is: Given $(P, P^a, P^b, P^c) \in \mathbb{G}_1$ and $W \in \mathbb{G}_2$ for unknown $a, b, c \in Z_p^*$, determine whether $W = e(P, P)^{abc}$ or a random element from $\mathbb{G}_2$.

Definition 4: *Discrete Logarithm (DL) problem*. The DL problem is: Given $Q \in \mathbb{G}_1$, compute $a \in Z_p^*$ such that $g^a = Q$.

IV. IPRE: A Novel Predicate-only Encryption Scheme for Inner Product Range

In this section, we present a novel predicate-only encryption scheme named IPRE (Inner Product Range Encryption), which will serve as the basis of our EPLQ solution for privacy-preserving spatial range query.

A. Overview

The proposed IPRE scheme allows computing inner products and comparing their values with a predefined range in a privacy-preserving way. As far as we know, our scheme is the first predicate/predicate-only encryption scheme for inner product range. In IPRE, both attributes and predicates are vectors. So we use *attribute vectors* and *predicate vectors* to refer to the attributes and predicates in IPRE. Let $\Lambda \subseteq \mathbb{Z}_p^t$ be the attribute set and $F \subseteq \mathbb{Z}_p^t$ be the class of predicates in IPRE. p is a big prime here. IPRE allows testing if the inner product of a vector from Λ and a vector from F is in a predefined range without disclosing the vectors.

IPRE scheme is a symmetric predicate-only encryption scheme, and it consists of four algorithms: **Setup** algorithm for generating a public parameter PP, an attribute encryption key AK and a predicate encryption key PK; **Enc** algorithm for encrypting attribute vectors to ciphertexts; **GenToken** algorithm for encrypting predicate vectors to tokens; **Check** algorithm for checking if a ciphertext's attribute satisfies a token's predicate.

For the reader's convenience, we summarize the important notations to be used in Table I.

B. Encoding Attribute Vectors and Predicate Vectors

Before describing IPRE's algorithms, we define the encodings of attribute vectors and predicate vectors, which serve as a building block of IPRE. Let $Encode_U()$ and $Encode_V()$ be the functions of encoding predicate vectors and attribute vectors, respectively. $Encode_U()$ takes a predicate vector $\overrightarrow{U}_i = (u_{i,1}, u_{i,2}, ..., u_{i,t})$ and a random integer h_i as input, and outputs a vector $\overrightarrow{EU}_i$. Similarly, $Encode_V()$ takes an attribute vector $\overrightarrow{V}_j = (v_{j,1}, v_{j,2}, ..., v_{j,t})$ and a random integer s_j as input, and outputs a vector $\overrightarrow{EV}_j$.

We want the encoding functions to meet the requirement that

$$\langle \overrightarrow{EU}_i, \overrightarrow{EV}_j \rangle = (\alpha \times \langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle + \beta)^d + h_i + s_j$$

for any pair of $\overrightarrow{EU}_i$ and $\overrightarrow{EV}_j$. α and β are two predefined secret integers. Next, we show how to find such encoding functions.

TABLE I: Notations frequently used in IPRE and EPLQ

Notation	Description
$\langle , \rangle$	inner product operator
α, β	two secret numbers in $\mathbb{F}_p$
$[\tau_1, \tau_2]$	an inner product range
AK	the key to encrypt attribute vectors
C_j	the ciphertext of the j-th attribute vector
d	a positive integer
$e(.,.)$	a non-degradable bilinear mapping
g	a generator of $\mathbb{G}_1$
$\mathbb{G}_1$	a cyclic group of order p
$\mathbb{G}_2$	a cyclic group of order p
K_i	the i-th token
n	the length of encoded vectors
M	a secret $n \times n$ invertible matrix over the Field $\mathbb{F}_p$.
N	the number of POIs in the LBS database
p	a big prime
PK	the key to encrypt predicate vectors (i.e. the key to generate tokens)
PP	the public parameter of IPRE
R	the number of matched POIs
r_i	the radius of the i-th query area
S	the set of all inner products' values
t	the length of attribute/predicate vector
T	matrix transpose operator
$\overrightarrow{U_i}$	the i-th predicate vector $\overrightarrow{U_i} = (u_{i,1}, u_{i,2}, ..., u_{i,t})$
$\overrightarrow{V_j}$	the j-the attribute vector $\overrightarrow{V_j} = (v_{j,1}, v_{j,2}, ..., v_{j,t})$
(x_i, y_i)	the coordinates of the i-th query area's centroid

Following the well known multinomial theorem, we have the equation below, where $\binom{d}{l_1, l_2, ..., l_{t+1}} = \frac{d!}{l_1! l_2! ... l_{t+1}!}$.

$$(\alpha \times \langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle + \beta)^d = (\beta + \alpha \times \sum_{k=1}^{t} u_{i,k} \times v_{j,k})^d$$

$$= \sum_{\substack{l_1 + l_2 + ... + l_{t+1} = d \\ l_1, l_2, ..., l_{t+1} \in [0,d]}} \binom{d}{l_1, l_2, ..., l_{t+1}} \beta^{l_{t+1}} \prod_{k=1}^{t} (\alpha \times v_{j,k})^{l_k} \prod_{k=1}^{t} u_{i,k}^{l_k}$$

Then, the last sum of the above equation $\sum_{\substack{l_1 + l_2 + ... + l_{t+1} = d \\ l_1, l_2, ..., l_{t+1} \in [0,d]}} \binom{d}{l_1, l_2, ..., l_{t+1}} \beta^{l_{t+1}} \prod_{k=1}^{t} (\alpha \times v_{j,k})^{l_k} \prod_{k=1}^{t} u_{i,k}^{l_k}$ has $\binom{d+t}{t}$ terms. Let $a_{i,m}$ and $b_{j,m}$ be the $\prod_{k=1}^{t} u_{i,k}^{l_k}$ and $\binom{d}{l_1, l_2, ..., l_{t+1}} \beta^{l_{t+1}} \prod_{k=1}^{t} (\alpha \times v_{j,k})^{l_k}$ respectively in the m-th term. Then the last sum can be written as $\sum_{m=1}^{\binom{d+t}{t}} a_{i,m} \times b_{j,m}$. Without loss of generality, let $l_{t+1} = d$ and $l_1, l_2, ..., l_t = 0$ when $m = 1$. So $a_{i,1} = 1$ and $b_{j,1} = \beta^d$. Then, we have the equation below.

$$(\alpha \times \langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle + \beta)^d + h_i + s_j$$

$$= h_i + s_j + \sum_{m=1}^{\binom{d+t}{t}} a_{i,m} \times b_{j,m}$$

$$= h_i + s_j + a_{i,1} \times b_{j,1} + \sum_{m=2}^{\binom{d+t}{t}} a_{i,m} \times b_{j,m}$$

$$= h_i \times 1 + 1 \times (\beta^d + s_j) + \sum_{m=2}^{\binom{d+t}{t}} a_{i,m} \times b_{j,m}$$

$$= \langle (h_i, 1, a_{i,2}, a_{i,3}, ..., a_{i,n-1}), (1, \beta^d + s_j, b_{j,2}, b_{j,3}, ..., b_{j,n-1}) \rangle$$

Thus, we can define $Encode_U()$ and $Encode_V()$ as below where $n = \binom{d+t}{t} + 1$.

$$Encode_U(\overrightarrow{U_i}, h_i) = (h_i, 1, a_{i,2}, a_{i,3}, ..., a_{i,n-1}) \quad (1)$$

$$Encode_V(\overrightarrow{V_j}, s_j) = (1, \beta^d + s_j, b_{j,2}, b_{j,3}, ..., b_{j,n-1}) \quad (2)$$

C. Setup Algorithm

The setup algorithm is a probabilistic algorithm, which takes a security parameter λ, the attribute/predicate vector length t and an inner product range $[\tau_1, \tau_2]$ as input. The algorithm outputs an attribute encryption key $AK = (\alpha, \beta, d, M)$, a predicate encryption key $PK = (d, M)$ and a public parameter $PP = ((\mathbb{G}_1, \mathbb{G}_2, g, p, e), (\Omega_k)_{k=\tau_1}^{\tau_2})$.

$\alpha, \beta \in \mathbb{F}_p$ are two random numbers for encoding functions. d is a positive integer, and its value depends on the security parameter. The scheme is more secure if d is bigger. d could be 2, or d is an integer satisfying $GCD(d, p-1) = 1$. If $d = 2$, α, β must be chosen to make sure that the intersection of the set $\{z : z = -z_1 - 2\beta/\alpha \bmod p, \tau_1 \le z_1 \le \tau_2\}$ and the set $S - [\tau_1, \tau_2]$ is empty. Here, S be the set of all possible values of inner products. M is an $n \times n$ random invertible matrix over the Field $\mathbb{F}_p$. n is the length of encoded attribute/predicate vectors.

$(\mathbb{G}_1, \mathbb{G}_2, g, p, e)$ is the pairing parameter generated by running $\mathcal{G}en(\lambda)$, and $\Omega_k = Hash(e(g,g)^{(\alpha \times perm(k) + \beta)^d})$. Here, $Hash()$ is a hash function and $perm()$ is a random bijection mapping from $[\tau_1, \tau_2]$ to $[\tau_1, \tau_2]$, i.e. a random permutation function. Let $|Hash()|$ be the range size of $Hash()$. $Hash()$ is chosen to make $p \gg |Hash()| \gg \tau_2 - \tau_1$.

D. Enc Algorithm

The algorithm of encrypting attribute vectors is a probabilistic algorithm, which takes an attribute vector $\overrightarrow{V_j} = (v_{j,1}, v_{j,2}, ...v_{j,t})$ and a random number $s_j \in F_p$ as input, and outputs $C_j = (c_{j,1}, c_{j,2}..., c_{j,n+1}) = ((g^{v''_{j,k}})_{k=1}^n, e(g,g)^{s_j})$. Here, $(v''_{j,1}, v''_{j,2}, ...v''_{j,n})^T = M^{-1}(Encode_V(\overrightarrow{V_j}, s_j))^T \bmod p$. T is the matrix transpose operator.

E. GenToken Algorithm

The token generation algorithm $GenToken$ is a probabilistic algorithm, which takes a predicate vector $\overrightarrow{U_i} = (u_{i,1}, u_{i,2}, ...u_{i,t})$ and a random number $h_i \in F_p$ as input, and outputs $K_i = (q_{i,1}, q_{i,2}..., q_{i,n+1}) = ((g^{u''_{i,k}})_{k=1}^n, e(g,g)^{h_i})$. Here, $(u''_{i,1}, u''_{i,2}, ...u''_{i,n}) = Encode_U(\overrightarrow{U_i}, h_i)M \bmod p$.

F. Check Algorithm

The check algorithm takes a ciphertext $C_j = (c_{j,1}, c_{j,2}..., c_{j,n+1})$ of an attribute vector $\overrightarrow{V_j}$ and a token $K_i = (q_{i,1}, q_{i,2}..., q_{i,n+1})$ associated with a predicate vector $\overrightarrow{U_i}$ as input. The algorithm computes $\Psi = \frac{\prod_{k=1}^n e(c_{j,k}, q_{i,k})}{c_{j,n+1} \times s_{j,n+1}}$. If $Hash(\Psi)$ is in the set $\{\Omega_k : \tau_1 \le k \le \tau_2\}$, the algorithm outputs 1. Otherwise, it outputs 0.

Remark. If two inner products are equal, their Ψ are the same. This is not desirable for some applications. This problem can be circumvented by adding randomness in the generation of predicate/attribute vectors. For a pair of fixed predicate vector and attribute vector, the value Ψ is still fixed. However, given a pair of predicate and attribute, their vectors and their vectors' inner product all have multiple possible values. We will demonstrate it in our EPLQ solution in Section V.

Correctness proof. First, we prove $Hash(\Psi) \in \{\Omega_k : \tau_1 \le k \le \tau_2\}$ if $\tau_1 \le \langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle \le \tau_2$. Recall that $\Omega_k = Hash(e(g,g)^{(\alpha \times perm(k) + \beta)^d})$. From the equation below, it is easy to find out that $Hash(\Psi) \in \{\Omega_k : \tau_1 \le k \le \tau_2\}$ if $\tau_2 \ge \langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle \ge \tau_1$.

$$\begin{aligned}
\Psi &= \frac{\prod_{k=1}^n e(c_{j,k}, q_{i,k})}{c_{j,n+1} \times s_{j,n+1}} = \frac{\prod_{k=1}^n e(g^{u''_{i,k}}, g^{v''_{j,k}})}{e(g,g)^{h_i} \times e(g,g)^{s_j}} \\
&= \frac{e(g,g)^{\sum_{k=1}^n u''_{i,k} \times v''_{j,k}}}{e(g,g)^{h_i + s_j}} \\
&= \frac{e(g,g)^{Encode_U(\overrightarrow{U_i}, h_i) M M^{-1} (Encode_V(\overrightarrow{V_j}, s_j))^T}}{e(g,g)^{h_i + s_j}} \\
&= \frac{e(g,g)^{\langle Encode_U(\overrightarrow{U_i}, h_i), Encode_V(\overrightarrow{V_j}, s_j) \rangle}}{e(g,g)^{h_i + s_j}} \\
&= \frac{e(g,g)^{(\alpha \times \langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle + \beta)^d + h_i + s_j}}{e(g,g)^{h_i + s_j}} \\
&= e(g,g)^{(\alpha \times \langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle + \beta)^d}
\end{aligned}$$

Second, we prove $Hash(\Psi) \notin \{\Omega_k : \tau_1 \le k \le \tau_2\}$ with overwhelming probability if $\langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle \notin [\tau_1, \tau_2]$.

Lemma 4.1: $\forall a \in \mathbb{F}_p$, a has at most one d-th root if $GCD(d, p-1) = 1$ and p is a prime.

Proof: Clearly, if $a = 0$, it has only one d-th root 0. If $a \neq 0$, we prove the lemma by contradiction. If a had two or more distinct d-th roots, let $\chi_1, \chi_2 \in \mathbb{F}_p$ be two of them. Since $\chi_1^d = \chi_2^d = a$, we have $(\chi_1/\chi_2)^d = 1$. Noticing that $\mathbb{Z}_p^*$ is a cyclic group of order $p-1$, we have $order(\chi_1/\chi_2) | d$ and $order(\chi_1/\chi_2) | (p-1)$. So $order(\chi_1/\chi_2)$ is a common divisor of d and $p-1$. Since χ_1 and χ_2 are distinct, we have $order(\chi_1/\chi_2) \neq 1$. This contradicts with $GCD(d, p-1) = 1$. Therefore, the assumption of a having two or more distinct square roots must be false. ■

Lemma 4.2: $\forall a \in \mathbb{F}_p$, a has at most two square roots if p is a prime.

Proof: Clearly, if $a = 0$, it has only one square root 0. If $a \neq 0$, we prove the lemma by contradiction. Assume that a has three or more distinct square roots. Then we can find two square roots $\chi_1, \chi_2 \in \mathbb{F}_p$ satisfying $\chi_1/\chi_2 \neq -1 \bmod p$. Because $\chi_1^2 = \chi_2^2 = a$, we have $(\chi_1/\chi_2)^2 = 1$ and the order of χ_1/χ_2 is 2. This contracts with the fact that $\mathbb{Z}_p^*$ has only one subgroup of order 2 and the subgroup's elements are 1 and $-1 \bmod p$. ■

Let Υ_k be $e(g,g)^{(\alpha \times perm(k) + \beta)^d}$ for any $k \in [\tau_1, \tau_2]$. Then $Hash(\Upsilon_k) = \Omega_k$. Let's consider two cases of d as follows.

- Case 1: $GCD(d, p-1) = 1$. Consider an equation $f_k(z)$ over the field $\mathbb{F}_p$: $(\alpha \times z + \beta)^d \bmod p = \log \Upsilon_k \bmod p$. From Lemma 4.1, we know that $\log \Upsilon_k$ has only one d-th root over the Field $\mathbb{F}_p$. Then the equation has only one root $z = (\chi - \beta)/\alpha \bmod p$ where $\chi \in \mathbb{F}_p$ is the d-th root of $\log \Upsilon_k$. From the definition of Υ_k, we know that $\tau_1 \le$

$z \leq \tau_2$ for any $k \in [\tau_1, \tau_2]$. Therefore, if $\langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle \notin [\tau_1, \tau_2]$, the resulting $\Psi = e(g,g)^{(\alpha \times \langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle + \beta)^d}$ is not equal to any Υ_k. Because the range size of $Hash()$ is much bigger than the size of $[\tau_1, \tau_2]$, the probability of $Hash(\Psi) \in \{\Omega_k : \tau_1 \leq k \leq \tau_2\}$ is very low.

- Case 2: $d = 2$. From Lemma 4.2, we know that the equation $f_k(z) : (\alpha \times z + \beta)^2 \bmod p = \log \Upsilon_k \bmod p$ has at most two roots. Suppose there are two roots z_1 and z_2. We have $(\alpha \times z_2 + \beta) \bmod p = -(\alpha \times z_1 + \beta) \bmod p$ and $z_2 = (-z_1 - 2\beta/\alpha) \bmod p$. From the definition of Υ_k, we know that at least one of the roots is in $[\tau_1, \tau_2]$. Without loss of generality, suppose $z_1 \in [\tau_1, \tau_2]$. Then z_2 is in the set $\{z : z = -z_1 - 2\beta/\alpha \bmod p, \tau_1 \leq z_1 \leq \tau_2\}$. Recall that the values of β and α are chosen to avoid the overlap between this set and the set $S - [\tau_1, \tau_2]$. Then, if $\langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle \notin [\tau_1, \tau_2]$, the inner product is not in the set of roots. Therefore, the resulting Ψ is not equal to any Υ_k. Again, because $|Hash()| \gg \tau_2 - \tau_1$, the probability of $Hash(\Psi) \in \{\Omega_k : \tau_1 \leq k \leq \tau_2\}$ is very low.

Remark. With a suitable $Hash()$ function, using $\{\Omega_k : \tau_1 \leq k \leq \tau_2\}$ instead of $\{\Upsilon_k : \tau_1 \leq k \leq \tau_2\}$ to check inner product range can reduce the size of public parameter, and the resulting false positives are rare. We will show in Section VII that the false positives are negligible in our EPLQ solution.

V. EPLQ: Proposed Solution for Privacy-preserving Spatial Range Query

In this section, we propose a novel tree data structure named $\hat{ss}$-tree and the EPLQ solution based on the above IPRE scheme and $\hat{ss}$-tree.

A. Preliminary: ss-tree

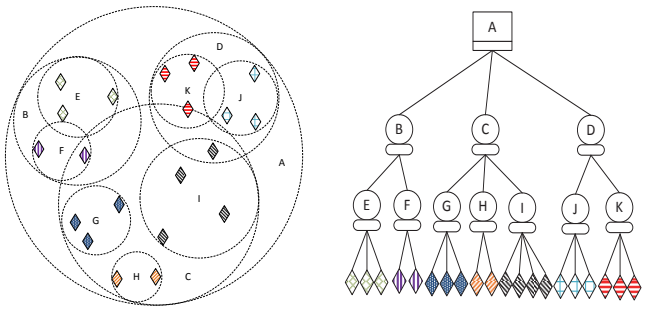
The $\hat{ss}$-tree introduced in this work is a variant of ss-tree [10]. For indexing spatial data, there actually exist quite a few data structures such as r-tree [11] and ss-tree, and some of them (e.g., ss-tree and sh-tree [12]) can be used for spatial range query. When such kind of data structures are used for privacy-preserving query, location data, e.g., the location data of points, circular areas, rectangular areas, single-dimension ranges, etc., must be concealed. Since the above-proposed IPRE scheme can conceal location data of points and circular areas, and at the same time ss-tree and some of its variants only need these location data, it is natural to apply IPRE to these data structures for privacy-preserving query. Hence, we choose ss-tree for its simplicity, and propose $\hat{ss}$-tree based on ss-tree and IPRE.

As shown in Fig. 3, ss-tree and $\hat{ss}$-tree share the same structure, and the data structures of their nodes are compared in Fig. 4. Before describing $\hat{ss}$-tree, we give an introduction on ss-tree. An ss-tree has following properties: 1) each non-root parent node has m_{min} to m_{max} children; 2) the root node has 0 to m_{max} children; 3) each leaf node represents a record; 4) each node has a centroid field and a radius field.

In the context of spatial database of Cartesian coordinate system, the centroid is a pair of coordinates (x, y). A leaf node's centroid is the corresponding POI's coordinates, and its radius is 0. A non-leaf node's centroid and radius depend on its children. Its centroid is the mean of all its children's centroids. Its radius is not smaller than the distance between its centroid and any descendant node's centroid. In other words, a non-leaf node is associated with a circular area defined by the node's centroid and radius, and all its descendant nodes are in this area.

The same as other tree index structures, a non-leaf node has a field (child_pointer_array) for pointers to its children, and a leaf node has a field (leaf_data) storing some data of the corresponding record (e.g. the whole record, the pointer to the record, or the record's ID). A node of ss-tree also has some other fields to support tree building, approximation search and sampling operations. We omit these fields in this paper as they are not relevant to our solution. Please refer to [10] for the details and the building of ss-tree.

With the ss-tree, searching POI records matching a spatial range query is very efficient. Noticing that all descendant nodes of a non-leaf node are in the non-leaf node's associated circular area. Search POI records can be done by scanning the ss-tree from root to leaves. If a node's circular area intersects with the query area, all children nodes of the node will be scanned. Otherwise, its descendant nodes will be skipped. Then $O(\log N + R)$ trees nodes will be scanned to find matched records where N is the number of POI records in the database and R is the number of matched records.



(a) POI distribution (b) ss-tree and $\hat{ss}$-tree structure

Fig. 3: Index POIs with ss-tree and $\hat{ss}$-tree

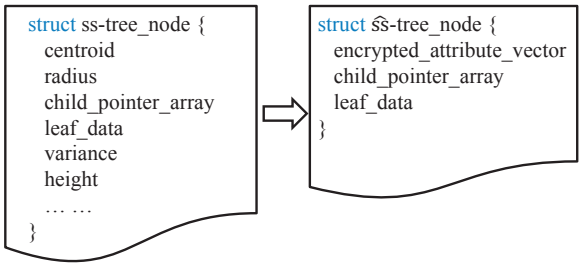


Fig. 4: The data structures of ss-tree node and $\hat{ss}$-tree node

B. Proposed $\hat{ss}$-tree

$\hat{ss}$-tree is the core of our EPLQ solution. It is a variant of ss-tree. They share the same tree structure, which is shown in Fig. 3. The difference between ss-tree and $\hat{ss}$-tree is the tree nodes' data, which are shown in Fig. 4. $\hat{ss}$-tree hides

each tree node's location information using our predicate-only encryption scheme, and removes unnecessary information. Because of the encryption, detecting circular area intersection and matched records are also different when searching matched records with the tree. More specifically, we use two kinds of inner products for detecting circular area intersection and matched records, and our IPRE scheme assures the detection via inner product range in a privacy-preserving way.

Building $\hat{ss}$-tree. $\hat{ss}$-tree can be built from ss-tree. After building an ss-tree for the spatial database, an $\hat{ss}$-tree can be built by the following steps.

- *Step 1:* configure parameters τ_1, τ_2 and ϕ.
  As mentioned in the IPRE scheme, τ_1 and τ_2 are the lower limit and upper limit of the given inner product range respectively. The lower limit τ_1 is fixed to 0 in $\hat{ss}$-tree. τ_2 is set to a value not smaller than r_{max1}^2 where r_{max1} is the maximum query range allowed in the system. ϕ is a positive integer used to scale down the inner products for detecting area intersection, and $\tau_2 \geq (\lceil r_{max1}/\phi \rceil + \lceil r_{max2}/\phi \rceil + 2)^2$ where r_{max2} is the biggest radius of ss-tree's circular areas. (Such inner products are scaled down to make them fit in a smaller range, which reduces the size of IPRE's public parameter. As explained later at the end of this subsection, the scaling down won't decrease accuracy.)
- *Step 2:* generate an attribute vector for each tree node from the node's centroid $(\grave{x}_j, \grave{y}_j)$ and radius $\grave{r}_j$.
  Case 1: the node is a leaf node. Generate the attribute vector $\overrightarrow{V_j}$ as below
  $((-1)^{\grave{\xi}_j}, (-1)^{\grave{\xi}_j}(\grave{r}_j^2 - \grave{x}_j^2 - \grave{y}_j^2), (-1)^{\grave{\xi}_j} \times 2\grave{x}_j, (-1)^{\grave{\xi}_j} \times 2\grave{y}_j, (-1)^{\grave{\xi}_j} \times 2\grave{r}_j, \grave{\xi}_j\tau_2, (-1)^{\grave{\xi}_j})$.
  Here, $\grave{\xi}_j$ is a random number, and its value is 1 or 0.
  Case 2: the node is a non-leaf node. Do the following two steps.
  (1) Modify the node's original circular area to a minimal normalized area covering the original one. In this paper, we say a circular area is a normalized area if its centroid's coordinates and its radius are all integral multiples of ϕ. The centroid's coordinates will be modified to the closest coordinates that are integral multiples of ϕ, and the radius will be modified to the smallest integral multiple of ϕ that makes the normalized area cover the original one.
  (2) After modifying $(\grave{x}_j, \grave{y}_j)$ and $\grave{r}_j$, generate the attribute vector $\overrightarrow{V_j}$ as below

  $$((-1)^{\grave{\xi}_j}\grave{\mu}_j, (-1)^{\grave{\xi}_j}(\grave{\mu}_j(\grave{r}_j^2 - \grave{x}_j^2 - \grave{y}_j^2)/\phi^2 + \grave{\theta}_j),$$
  $$(-1)^{\grave{\xi}_j}\grave{\mu}_j\grave{x}_j/\phi, (-1)^{\grave{\xi}_j}\grave{\mu}_j\grave{y}_j/\phi, (-1)^{\grave{\xi}_j}\grave{\mu}_j\grave{r}_j/\phi, \grave{\xi}_j\tau_2, (-1)^{\grave{\xi}_j})$$

  Here, $\grave{\mu}_j = \lfloor \tau_2/(\lceil r_{max1}/\phi + 1 \rceil + \grave{r}_j/\phi)^2 \rfloor$, and $\grave{\theta}_j$ is a random non-negative integer not more than $\grave{\mu}_j$ and $\tau_2 - \grave{\mu}_j(\lceil r_{max1}/\phi + 1 \rceil + \grave{r}_j/\phi)^2$. Again, $\grave{\xi}_j$ is a random number, and its value is 1 or 0.
- *Step 3:* encrypt each attribute vector with IPRE scheme.
- *Step 4:* remove all unnecessary fields of each tree node. At last, a node's data include its encrypted attribute vector. If the node is a non-leaf node, the data also include pointers to its children. If the node is a leaf node, its leaf_data field also stores the pointer to the corresponding record.

Searching $\hat{ss}$-tree. Searching $\hat{ss}$-tree is the same as searching ss-tree except that detecting circular area intersection and matched records are based on our IPRE scheme.

Suppose a spatial range query wants to find all POIs within a circular area centered at coordinates (x_i, y_i) with radius r_i. To search $\hat{ss}$-tree, the tokens of two predicate vectors associated with the query should be provided. The two vectors $\overrightarrow{U_i}$ and $\overrightarrow{U_i'}$ are shown below, and tokens are generated with IPRE's $GenToken$ algorithm.

$\overrightarrow{U_i} = ((-1)^{\xi_i}(\mu_i(r_i^2 - x_i^2 - y_i^2) + \theta_i), (-1)^{\xi_i}\mu_i, (-1)^{\xi_i}\mu_i x_i, (-1)^{\xi_i}\mu_i y_i, (-1)^{\xi_i}\mu_i r_i, 1, \xi_i\tau_2)$

$\overrightarrow{U_i'} = ((-1)^{\xi_i'}(r_i'^2 - x_i'^2 - y_i'^2)/\phi^2, (-1)^{\xi_i'}, (-1)^{\xi_i'}2x_i'/\phi, (-1)^{\xi_i'}2y_i'/\phi, (-1)^{\xi_i'}2r_i'/\phi, 1, \xi_i'\tau_2)$

Here, ξ_i and ξ_i' are random integers in $\{0, 1\}$. $\mu_i = \lfloor \tau_2/r_i^2 \rfloor$, and θ_i is a random non-negative integer not more than μ_i and $\tau_2 - \mu_i {r_i}^2$. (x_i', y_i') and r_i' are the centroid coordinates and radius of the minimal normalized area covering the query area. The way to find this normalized area is the same that in the generation of attribute vectors.

The tokens of $\overrightarrow{U_i}$ and $\overrightarrow{U_i'}$ are used for detecting matched records and circular area intersection respectively. We call the former vector POI-matching predicate vector and the latter area-intersecting predicate vector.

Algorithm 1 Search_$\hat{ss}$-tree(node nd, query_tokens Ks, node_list ndl)

```
1: \\ nd: the node to be searched
2: \\ Ks: the array of two tokens associated with the query's predicate
   vectors. Ks[0] is the token for POI matching detection, while Ks[1] is
   the one for detecting intersection of circular areas.
3: \\ ndl: the list to store matched leaf nodes
4:
5: C ← nd.encrypted_attribute_vector
6: if nd is a leaf node then
7:     if Check(Ks[0], C) == 1 then
8:         \\ nd's record matches the q's area
9:         Add nd to node_list ndl.
10:    end if
11: else
12:    if Check(Ks[0], C) == 1 then
13:        \\ nd's area intersects with the q's area
14:        for each child node cld_i of nd do
15:            Search_ŝs-tree(cld_i, Ks, ndl)
16:        end for
17:    end if
18: end if
```

Given the above tokens associated with the query, POI records matching the query can be found by searching $\hat{ss}$-tree. The pseudo-code of the search algorithm is shown in Algorithm 1. The search starts from the root node. If a non-leaf node's area intersects with the query area, all children of the node will be scanned. Otherwise, all descendant nodes of this non-leaf node are skipped. Detecting circular area intersection and matched records are based on our IPRE scheme for inner product range. We give the correctness proofs of the detection as follows.

Correctness of detecting matched records via inner product range. For any POI-matching predicate vector $\overrightarrow{U_i}$ and any leaf node's attribute vector $\overrightarrow{V_j}$, we have the equation

below.

$$\begin{aligned}
&\langle\overrightarrow{U_i},\overrightarrow{V_j}\rangle\\
=&\langle((-1)^{\xi_i}(\mu_i(r_i^2-x_i^2-y_i^2)+\theta_i),(-1)^{\xi_i}\mu_i,(-1)^{\xi_i}\mu_i x_i,\\
&(-1)^{\xi_i}\mu_i y_i,(-1)^{\xi_i}\mu_i r_i,1,\xi_i\tau_2),\\
&((-1)^{\dot{\xi}_j},(-1)^{\dot{\xi}_j}(\dot{r}_j^2-\dot{x}_j^2-\dot{y}_j^2),(-1)^{\dot{\xi}_j}\times 2\dot{x}_j,(-1)^{\dot{\xi}_j}\times 2\dot{y}_j,\\
&(-1)^{\dot{\xi}_j}\times 2\dot{r}_j,\dot{\xi}_j\tau_2,(-1)^{\dot{\xi}_j})\rangle\\
=&(-1)^{\xi_i+\dot{\xi}_j}(\mu_i(r_i^2-x_i^2-y_i^2)+\theta_i)+(-1)^{\xi_i+\dot{\xi}_j}\mu_i(\dot{r}_j^2-\dot{x}_j^2-\dot{y}_j^2)+\\
&(-1)^{\xi_i+\dot{\xi}_j}2\mu_i x_i\dot{x}_j+(-1)^{\xi_i+\dot{\xi}_j}2\mu_i y_i\dot{y}_j+(-1)^{\xi_i+\dot{\xi}_j}2\mu_i r_i\dot{r}_j+\\
&\dot{\xi}_j\tau_2+(-1)^{\dot{\xi}_j}\xi_i\tau_2\\
=&(-1)^{\xi_i+\dot{\xi}_j}(\mu_i((r_i+\dot{r}_j)^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2)+\theta_i)+\dot{\xi}_j\tau_2+\\
&(-1)^{\dot{\xi}_j}\xi_i\tau_2\\
=&\begin{cases}\mu_i((r_i+\dot{r}_j)^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2)+\theta_i, & \text{if } \xi_i=\dot{\xi}_j\\ \tau_2-\mu_i((r_i+\dot{r}_j)^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2)-\theta_i, & \text{otherwise}\end{cases}
\end{aligned}$$

As the tree node is a leaf node, $\dot{r}_j=0$. Then, we have

$$\langle\overrightarrow{U_i},\overrightarrow{V_j}\rangle=\begin{cases}\mu_i(r_i^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2)+\theta_i, & \text{if } \xi_i=\dot{\xi}_j\\ \tau_2-\mu_i(r_i^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2)-\theta_i, & \text{otherwise}\end{cases}$$

The distance between the leaf node's POI and the query point is $\sqrt{(x_i-\dot{x}_j)^2+(y_i-\dot{y}_j)^2}$. Then the record matches the query if and only if $r_i^2\geq r_i^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2\geq 0$. As shown below, record matching can be detected by examining if $\langle\overrightarrow{U_i},\overrightarrow{V_j}\rangle$ is in the range $[0,\tau_2]$ as well.

$$\begin{aligned}
&r_i^2\geq r_i^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2\geq 0\\
\Leftrightarrow&\tau_2\geq\mu_i(r_i^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2)+\theta_i\geq 0 \text{ and}\\
&\tau_2\geq\tau_2-\mu_i(r_i^2-(x_i-\dot{x}_j)^2-(y_i-\dot{y}_j)^2)-\theta_i\geq 0\\
\Leftrightarrow&\tau_2\geq\langle\overrightarrow{U_i},\overrightarrow{V_j}\rangle\geq 0
\end{aligned}$$

For security reasons, the field order p in IPRE scheme is at least 160 bits, and $p\gg\langle\overrightarrow{U_i},\overrightarrow{V_j}\rangle\gg -p$ holds. Then we have

$$\tau_2\geq\langle\overrightarrow{U_i},\overrightarrow{V_j}\rangle\geq 0\Leftrightarrow\tau_2\geq\langle\overrightarrow{U_i},\overrightarrow{V_j}\rangle \bmod p\geq 0$$

Therefore, record matching can be detected by examining if $\langle\overrightarrow{U_i},\overrightarrow{V_j}\rangle \bmod p$ is in the range $[0,\tau_2]$.

Correctness of detecting the intersection of circular areas via inner product range. Similarly, for any area-intersecting predicate vector $\overrightarrow{U_i'}$ and any non-leaf node's attribute vector $\overrightarrow{V_j}$, we have the equation below.

$$\begin{aligned}
&\langle\overrightarrow{U_i'},\overrightarrow{V_j}\rangle\\
=&\begin{cases}\dot{\mu}_j((r_i'+\dot{r}_j)^2-(x_i'-\dot{x}_j)^2-(y_i'-\dot{y}_j)^2)/\phi^2+\dot{\theta}_j, & \text{if } \xi_i'=\dot{\xi}_j\\ \tau_2-\dot{\mu}_j((r_i'+\dot{r}_j)^2-(x_i'-\dot{x}_j)^2-(y_i'-\dot{y}_j)^2)/\phi^2-\dot{\theta}_j, & \text{otherwise}\end{cases}
\end{aligned}$$

The non-leaf node's circular area (been normalized) intersects with the query's normalized area if and only if $(r_i'+\dot{r}_j)^2\geq(r_i'+\dot{r}_j)^2-(x_i'-\dot{x}_j)^2-(y_i'-\dot{y}_j)^2\geq 0$. As shown below, the intersection of normalized areas can be detected by examining if $\langle\overrightarrow{U_i'},\overrightarrow{V_j}\rangle \bmod p$ is in the range $[0,\tau_2]$ as well. Please note that detecting intersection is to rule out subtrees not containing matched POIs. Expanding original areas to normalized areas only results in scanning more tree nodes. All matched POI records still can be found, and not matched records will not be included in the result.

$$\begin{aligned}
&(r_i'+\dot{r}_j)^2\geq(r_i'+\dot{r}_j)^2-(x_i'-\dot{x}_j)^2-(y_i'-\dot{y}_j)^2\geq 0\\
\Leftrightarrow&\tau_2\geq\dot{\mu}_j((r_i'+\dot{r}_j)^2-(x_i'-\dot{x}_j)^2-(y_i'-\dot{y}_j)^2)/\phi^2+\dot{\theta}_j\geq 0 \text{ and}\\
&\tau_2\geq\tau_2-\dot{\mu}_j((r_i'+\dot{r}_j)^2-(x_i'-\dot{x}_j)^2-(y_i'-\dot{y}_j)^2)/\phi^2-\dot{\theta}_j\geq 0\\
\Leftrightarrow&\tau_2\geq\langle\overrightarrow{U_i'},\overrightarrow{V_j}\rangle\geq 0\\
\Leftrightarrow&\tau_2\geq\langle\overrightarrow{U_i'},\overrightarrow{V_j}\rangle \bmod p\geq 0
\end{aligned}$$

C. EPLQ Design

Our EPLQ solution consists of two algorithms: system setup and spatial range search.

System setup. The LBS provider initializes the system by the steps below.

- *Step 1:* the LBS provider initializes the parameters and keys for the solution.
  The LBS provider initializes the public parameter and keys of the proposed IPRE scheme as well as the key of a standard encryption scheme (e.g. AES). Let $AK=(\alpha,\beta,d,M)$, $PK=(d,M)$ and $PP=((\mathbb{G}_1,\mathbb{G}_2,g,p,e),(\Omega_k)_{k=\tau_1}^{\tau_2})$ be the attribute encryption key, predicate encryption key and public parameter of IPRE scheme. PP is shared with the cloud. PK, $(\mathbb{G}_1,\mathbb{G}_2,g,p,e)$ and the key of the standard encryption scheme are shared with LBS users.
  Remark. The standard scheme will be used to encrypt POI records. IPRE scheme is for searching encrypted records. $(\Omega_k)_{k=\tau_1}^{\tau_2}$ in the public parameter is used for IPRE's $Check$ algorithm only, and LBS users don't need it to generate tokens.
- *Step 2:* the LBS provider builds an $\hat{ss}$-tree for the LBS database.
- *Step 3:* the LBS provider encrypts each POI record with the standard encryption scheme.
- *Step 4:* the LBS provider outsources all encrypted POI records and the $\hat{ss}$-tree to the cloud.

Spatial range search. Suppose an LBS user wants to find all POIs within a circular area centered at coordinates (x_i,y_i) with radius r_i. The privacy-preserving query is performed by following steps:

- *Step 1:* the LBS user generates two tokens for searching POI records with the proposed IPRE scheme.
  As elaborated earlier in Section V-B, to search $\hat{ss}$-tree, two tokens associated with the query area should be generated. The LBS user generates them following the way in Section V-B. Let $Ks[0]$ and $Ks[1]$ be the generated two tokens.
- *Step 2:* the user sends $(Ks[0],Ks[1])$ as a query to the cloud.
- *Step 3:* the cloud searches $\hat{ss}$-tree to find all leaf nodes matching the query from the user.
  The search algorithm has been given in Section V-B, and its pseudo-code is shown in Algorithm 1.
- *Step 4:* the cloud returns the corresponding POI records of matched leaf nodes to the user.
- *Step 5:* the LBS user decrypts received POI records with the shared key of the standard encryption scheme.

VI. Security Analysis

In this section, we analyze the security properties of the proposed EPLQ solution. Specifically, following the security requirements discussed earlier, our analysis will focus on how the proposed EPLQ solution can achieve the LBS data confidentiality and the user's location privacy.

The confidentiality of LBS data includes not only the confidentiality of POI records but also the confidentiality of location information in $\hat{ss}$-tree. On the other hand, user location privacy involves protecting sensitive location information in user queries and $\hat{ss}$-tree. The security of EPLQ solution depends on the underlying standard encryption scheme and IPRE scheme. The standard encryption scheme is responsible for preventing the cloud from learning POI records, while our IPRE scheme is responsible for protecting user location and POI location from the cloud. The current AES standard can be used as the standard scheme, and it is secure under ciphertext-only, known-sample and known-plaintext attacks. Thus, we focus on the analysis of user/POI location protection with IPRE scheme.

Security of query and POI index encryption. In EPLQ, user queries and the sensitive location information in $\hat{ss}$-tree are encrypted with IPRE scheme. A query consists of two tokens associated with two predicate vectors, which contains the LBS user's location information. For a predicate vector $\overrightarrow{U_i} = (u_{i,1}, u_{i,2}, ...u_{i,t})$, the corresponding token is $((g^{u''_{i,k}})_{k=1}^{n}, e(g,g)^{h_i})$ where $(u''_{i,1}, u''_{i,2}, ...u''_{i,n}) = Encode_U(\overrightarrow{U_i}, h_i)M \bmod p$. Because of the hardness of CDH problem, the attacker cannot reveal any exponent in the token even if knowing the predicate vector. Without the secret keys of IPRE (i.e. TK and AK), no one can reveal the predicate vector, the secret matrix M or the random number h_i. Because of the randomness and secretness of h_i, encrypting predicate vector to token is semantically secure. Therefore, it is secure under ciphertext-only and known-sample attacks. The sensitive location information in $\hat{ss}$-tree is concealed with attribute vector encryption. The encryption is very similar to predicate vector encryption, and their security properties are same. Therefore, we omit its security analysis.

Security under the attack on inner products. As discussed above, it's hard to reveal user queries and POI locations directly from the ciphertexts of attribute vectors and predicate vectors. Alternatively, the attacker may attempt to recover the inner products of predicate vectors and attribute vectors first, and then reveal the vectors containing information about user locations and POI locations. Next, we show how this attack works and its countermeasure.

The attacker may attempt to recover inner products through exhaustive attacks on $(\Upsilon_k = e(g,g)^{(\alpha \times perm(k)+\beta)^d})_{k=\tau_1}^{\tau_2}$. The exponents $((\alpha \times perm(k) + \beta)^d)_{k=\tau_1}^{\tau_2}$ could be viewed as d-degree polynomials of $d+1$ terms where the variables are α and β. The coefficients of the polynomials are in the range $[\tau_1^d, \tau_2^d]$. These polynomials are in the same vector space of dimension $d+1$. Any $d+1$ of these polynomials are linearly independent, and any $d+2$ of them are not. Thus, for any $\Upsilon_{k_1}, \Upsilon_{k_2}, ...\Upsilon_{k_{d+2}}$, there exists non-zero $\lambda_1, \lambda_2, ..., \lambda_{d+2}$ satisfying the equation below.

$$\prod_{l=1}^{d+2} \Upsilon_{k_l}^{\lambda_l} = \prod_{l=1}^{d+2} e(g,g)^{\lambda_l \times (\alpha \times perm(k_l)+\beta)^d} = e(g,g)^0$$

The attacker can find such $\lambda_1, \lambda_2, ..., \lambda_{d+2}$ through exhaustive search, and then recover $d+2$ inner products $perm(k_1)...perm(k_{d+2})$. Recall that τ_1 is set to 0 when using IPRE in EPLQ. There are $(\tau_2 + 1) \times \tau_2 ... \times (\tau_2 - d)$ possible values for the combination $(perm(k_1), perm(k_2), ..., perm(k_{d+2}))$. Therefore, the complexity of recovering inner products is $O(\tau_2^{d+2})$.

With enough known inner products, the attacker can further reveal sensitive location data by generating and solving an over-determined nonlinear polynomial equation system. Given the inner product $\rho_{i,j}$ of an unknown predicate vector $\overrightarrow{U}_i$ and attribute vector $\overrightarrow{V}_j$, the attacker can generate the equation $\langle \overrightarrow{U}_i, \overrightarrow{V}_j \rangle = \rho_{i,j}$. The unknowns are the location data and random numbers used to generate the vectors. From the inner products of ε encoded predicate vectors and w encoded attribute vectors, a system of $\varepsilon \times w$ equations and $O(\varepsilon + w)$ unknowns can be generated. If ε and w are big enough, the system is over-determined. There are polynomial time algorithms [13], [14] for solving over-determined nonlinear polynomial systems. Though the generated system has many solutions, an attacker knowing enough plaintext samples can reveal user locations from the solutions.

Noticing that the complexity of recovering inner products is $O(\tau_2^{d+2})$, we prevent the attack above by configuring big enough τ_2 and d. d is configurable parameter in our IPRE scheme, while, as demonstrated in our EPLQ solution, τ_2 can be configured by controlling the generation of predicate/attribute vectors. In our prototype, $\tau_2 = 10^8$ and $d = 2$. Then the complexity of the attack is over $O(2^{106})$.

VII. Performance Evaluation

In this section, we evaluate the performance of the proposed EPLQ solution in terms of communication cost, computational cost, storage cost and accuracy.

A. Implementation and Experimental Settings

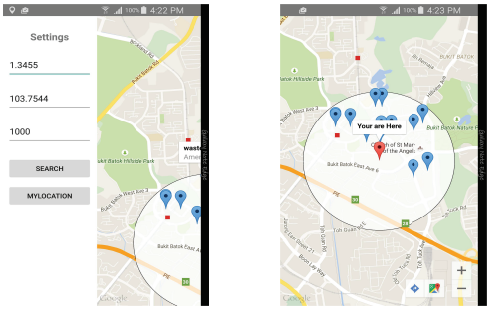


Fig. 5: User interface of EPLQ

We have implemented EPLQ in JAVA, and the user interface is shown in Fig. 5. We tested EPLQ's performance in a testbed of two workstations and one Android phone. These machines

play the roles of LBS provider, cloud and mobile LBS user, respectively. The hardware and software of these machines are shown in Table II(a), while the parameter settings are shown in Table II(b).

TABLE II: Experimental setting

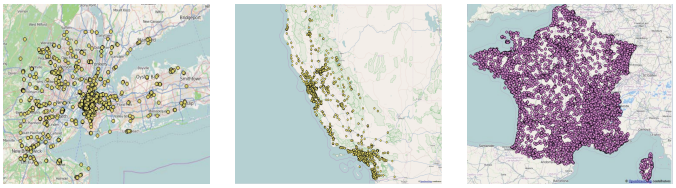
(a) Testbed setting

Role	Machine	Hardware & Software
LBS provider	workstation	CPU: Intel Xeon E5-2609 (quad-core 2.5 GHz) 64 GB memory; Windows 7 and Java EE 8
cloud	workstation	the same as above
LBS user	GALAXY Note Edge	CPU: Krait 400 (quad-core 2.7 GHz) 3 GB memory; Android 5.0.1

(b) Parameter setting

Parameter	$\|p\|$	$\|Hash()\|$	$\|\Omega_k\|$	t	d	n
Setting	160	64	64	7	2	37
Parameter	τ_1	τ_2	m_{max}	m_{min}	ϕ	r_{max1}
Setting	0	10^8	8	4	200	10^4

In addition, three datasets are extracted from the OpenStreetMap project (www.openstreetmap.org) for our experiments. The POIs in the datasets are the restaurants from New York, California and France. Their distributions are shown in Fig. 6, and the POI counts of the three datasets are 1676, 6994 and 28340, respectively.



(a) POIs in New York (b) POIs in California (c) POIs in France

Fig. 6: POI distributions of experimental datasets including restaurants in New York, California, and France

B. Computational Cost at User Side

To generate a query, an LBS user needs to encrypt two predicate vectors, which requires $2n$ modular exponentiations, about $2n^2$ multiplications and about $2n^2$ additions. n is the length of encoded vectors. To see whether the computational cost is acceptable for mobile LBS users or not, we measured the query generation latency. We let the Android phone in our testbed generate 1000 queries, and the average latency per query generation is about 0.9 second.

C. LBS Provider's Computational Cost

During system setup, the LBS provider needs to encrypt POI records, setup IPRE and build the $\hat{ss}$-tree. Because the cost of record encryption is the same as that in most solutions, we evaluate only the computational cost related to IPRE and $\hat{ss}$-tree here. The cost is evaluated by system setup latency, which is defined as: the time used to setup IPRE and build the $\hat{ss}$-tree. As shown in Table III, the latencies for the three datasets are between 1 and 3 hours. Considering that system setup is conducted only once, the computational cost is acceptable.

TABLE III: System setup latency

dataset	New York	California	France
latency (in minutes)	94	105	149

D. Cloud's Computational Cost

Recall that searching $\hat{ss}$-tree to find matched records requires to scan $O(\log N + R)$ trees nodes for the database with N records and the query having R matched records. Determining whether a record or tree node matches a query or not requires computing $Check(K_i, C_j)$. Computing the function requires $n = 37$ pairings and multiplications.

To see whether the computational cost of searching database is acceptable or not, we conducted experiments on three datasets. For each dataset, 1000 random query points are chosen. If a query point's location is not near any POI, the query is not realistic and query latency is lower than that in normal situations. To avoid that, in the experiments, each query point's location is the same as one random POI's. For each query point, we generate three queries with radiuses of 500m, 1km and 2km respectively. Therefore, 3000 queries are generated for each dataset. We measured the average search latency of these queries for each dataset, and the results are shown in Fig. 7. As expected, the latency increases very slow when increasing POI count and query radius. In the experiments, a workstation plays the role of cloud, and only four CPU cores can be utilized to do the computing. A real cloud has much more computing resources, and the query latency at a real cloud should be much lower.

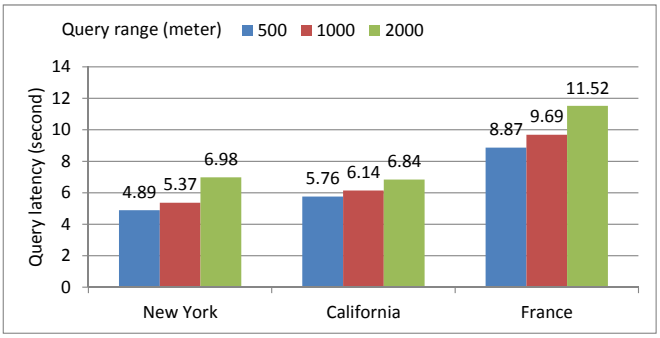


Fig. 7: POI query latency at cloud side. Note that the latency should be much lower once deployed at a real cloud.

E. Communication Cost and Storage Cost

To make a query, an LBS user sends two tokens to the cloud. The communication cost is $O(n \times \log p)$. Under the settings in Table II(b), the traffic is 4.75 KB, which is acceptable. The user has to store the attribute encryption key AK and pairing parameter locally. The storage usage is dominated by M, which is about 27 KB. This is negligible even for a mobile LBS user.

Let N be the number of POI records in the database. In addition to LBS data, the cloud also needs to store the public parameter of IPRE and an $\hat{ss}$-tree of less than $4N/3$ nodes (for the case of $m_{min} = 4$). The size of the public parameter is

dominated by $(\Omega_k)_{k=\tau_1}^{\tau_2}$, which is $(\tau_2-\tau_1+1)\times\lceil\log|Hash()|\rceil$ bits. It is about 763 MB for settings in Table II(b). A tree node's size is around 1.3 KB. Assume that there are 1 million records in the database. The cloud needs at most 2.42 GB storage space in total. The public parameter and $\hat{ss}$-tree can fit in the memory of even one single server. Therefore, the storage cost is acceptable. The LBS provider sends the cloud the public parameter and the tree only once. The communication cost is also acceptable.

F. Accuracy

As discussed in Section IV, using hash function in IPRE scheme reduces the size of public parameter, but introduces some false positives. This won't hurt the accuracy of EPLQ solution. The false positive rate is $(\tau_2 - \tau_1 + 1)/|Hash()|$, which is about 5.42×10^{-12}. $O(\log N + R)$ tree nodes are scanned during a query. This number is at most a few hundreds. Then the probability that a query result contains false positive(s) is at most a few hundred times of 5.42×10^{-12}, which is negligible.

VIII. Related Works

Our work is related to not only privacy-preserving LBS but also privacy-preserving query over outsourced encrypted data. In this section, we introduce some related works that can be used to realize privacy-preserving POI query, though some of them are not designed for POI query or LBS. In the literature, there are four kinds privacy-preserving queries over POIs: spatial range query [1], NN (Nearest Neighbour) query [15], KNN (K Nearest Neighbours) query [2], [16]–[20] and multi-dimensional range query [20]–[26]. Spatial range query cannot be replaced by NN query and KNN query, which all return the nearest neighbour(s) to a given location. They have different usages. Sometimes spatial range query may be replaced by multi-dimensional range query, which returns POIs in a rectangular area instead of a circular area. However, the inaccurate result is not desirable. Next, we review the works applicable to privacy-preserving spatial range query.

A. Solutions Applicable to Outsourced LBS

Privacy-preserving spatial range query based on coordinate transformation. In the solution based on coordinate transformation [1], the coordinates of queries and POIs in the orignal coordinate system are transformed to new coordinates in a new coordinate system. After the transformation, the distance information of any two points is still preserved. Coordinate transformation is very efficient, and the return results are accurate. However, solutions designed based on coordinate transformation would be vulnerable to known-sample attacks [2].

Privacy-preserving POI query based on PIR. As far as we know, only PIR-based solutions [3], [4] can protect the privacy in both public LBS and outsourced LBS. PIR (Private Information Retrieval) [5] is a privacy primitive hiding the retrieved data item's ID from the database server(s). Because the data items being retrieved are hidden from the database server(s), whether two queries' results are the same or not are undetectable. Therefore, PIR-based solutions are resilient to access-pattern attacks. PIR can be used to realize all the four kinds of POI queries. However, PIR is very communication and computationally costly [6] for the following reasons. PIR requires linearly scanning all POI records including their location data (coordinates and radiuses) and non-location data. Moreover, to use PIR in LBS, an LBS user must additionally access the LBS database's index data in a privacy-preserving manner. PIR can retrieve records if given their IDs. To support spatial range query, an LBS user should obtain nearby POIs' record IDs from index data in a privacy-preserving manner. PIR or other techniques may be used to obtain such IDs.

B. Solutions for Public LBS Only

Privacy-preserving LBS based on anonymous communication. In this kind of solutions [27], [28], one or more third parties relay messages between users and the LBS provider. This approach hides the linkage between user identities and messages from the LBS provider. The query area would be exposed to the LBS provider, but the user sending the query is hidden among a set of users.

Privacy-preserving LBS based on location obfuscation. In this kind of solutions [29], [30], to prevent the LBS provider from knowing users' precise locations, users submit low-precision locations or fake locations along with real locations. These solutions offer a weak level of privacy.

Privacy-preserving LBS based on spatial cloaking. This kind of solutions [31], [32] combine anonymous communication and location obfuscation techniques together. To the LBS provider, a user cannot be identified from a set of users in a cloaking area, and the cloaking area instead of users' precise locations are sent to the LBS provider.

All above solutions can be applied to a wide range of location-based services including POI query. However, their techniques don't allow the cloud to search encrypted data. Therefore, they cannot be used for outsourced LBS where LBS data in the cloud are encrypted.

IX. Conclusion

In this paper, we have proposed EPLQ, an efficient privacy-preserving spatial range query solution for smart phones, which preserves the privacy of user location, and achieves confidentiality of LBS data. To realize EPLQ, we have designed a novel predicate-only encryption scheme for inner product range named IPRE and a novel privacy-preserving index tree named $\hat{ss}$-tree. EPLQ's efficacy has been evaluated with theoretical analysis and experiments, and detailed analysis shows its security against known-sample attacks and ciphertext-only attacks. Our techniques have potential usages in other kinds of privacy-preserving queries. If the query can be performed through comparing inner products to a given range, the proposed IPRE and $\hat{ss}$-tree may be applied to realize privacy-preserving query. Two potential usages are privacy-preserving similarity query and long spatial range query. In the future, we will design solutions for these scenarios and identify more usages.

REFERENCES

[1] A. Gutscher, "Coordinate transformation - a solution for the privacy problem of location based services?" in *20th International Parallel and Distributed Processing Symposium (IPDPS 2006), Proceedings, 25-29 April 2006, Rhodes Island, Greece*, 2006. [Online]. Available: http://dx.doi.org/10.1109/IPDPS.2006.1639681

[2] W. K. Wong, D. W.-l. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases," in *SIGMOD*. ACM, 2009, pp. 139–152.

[3] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," in *SIGMOD*. ACM, 2008, pp. 121–132.

[4] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan, "Practical k nearest neighbor queries with location privacy," in *ICDE*. IEEE, 2014, pp. 640–651.

[5] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 965–981, 1998.

[6] F. Olumofin and I. Goldberg, "Revisiting the computational practicality of private information retrieval," in *Financial Cryptography and Data Security*. Springer, 2012, pp. 158–172.

[7] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings*, 2008, pp. 146–162. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-78967-3_9

[8] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, 2007, pp. 535–554. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-70936-7_29

[9] D. Boneh and M. K. Franklin, "Identity-based encryption from the weil pairing," *SIAM J. Comput.*, vol. 32, no. 3, pp. 586–615, 2003. [Online]. Available: http://dx.doi.org/10.1137/S0097539701398521

[10] D. A. White and R. Jain, "Similarity indexing with the ss-tree," in *ICDE*. IEEE, 1996, pp. 516–523.

[11] A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18-21, 1984*, 1984, pp. 47–57. [Online]. Available: http://doi.acm.org/10.1145/602259.602266

[12] T. K. Dang, J. Küng, and R. Wagner, "The sh-tree: A super hybrid index structure for multidimensional data," in *Database and Expert Systems Applications, 12th International Conference, DEXA 2001 Munich, Germany, September 3-5, 2001, Proceedings*, 2001, pp. 340–349. [Online]. Available: http://dx.doi.org/10.1007/3-540-44759-8_34

[13] B.-Y. Yang and J.-M. Chen, "All in the xl family: Theory and practice," in *ICISC*. Springer, 2004, pp. 67–86.

[14] G. Ars, J.-C. Faugere, H. Imai, M. Kawazoe, and M. Sugita, "Comparison between xl and gröbner basis algorithms," in *ASIACRYPT*. Springer, 2004, pp. 338–353.

[15] B. Yao, F. Li, and X. Xiao, "Secure nearest neighbor revisited," in *ICDE*. IEEE, 2013, pp. 733–744.

[16] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *ICDE*. IEEE, 2014, pp. 664–675.

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

[18] B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu, "Secure multidimensional range queries over outsourced data," *The VLDB Journal*, vol. 21, no. 3, pp. 333–358, 2012.

[19] I.-T. Lien, Y.-H. Lin, J.-R. Shieh, and J.-L. Wu, "a novel privacy preserving location-based service protocol with secret circular shift for k-nn search," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 6, pp. 863–873, 2013.

[20] M. L. Yiu, G. Ghinita, C. S. Jensen, and P. Kalnis, "Enabling search services on outsourced private spatial data," *VLDB J.*, vol. 19, no. 3, pp. 363–384, 2010. [Online]. Available: http://springerlink.metapress.com/content/68k6n4176t48q211/

[21] E. Shi, J. Bethencourt, T.-H. Chan, D. Song, and A. Perrig, "Multidimensional range query over encrypted data," in *S&P*. IEEE, 2007, pp. 350–364.

[22] B. Wang, Y. Hou, M. Li, H. Wang, and H. Li, "Maple: scalable multidimensional range search over encrypted cloud data with tree-based index," in *Proceedings of the 9th ACM symposium on Information, computer and communications security*. ACM, 2014, pp. 111–122.

[23] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *SIGMOD*. ACM, 2004, pp. 563–574.

[24] J. Shao, R. Lu, and X. Lin, "Fine: A fine-grained privacy-preserving location-based service framework for mobile devices," in *INFOCOM*. IEEE, 2014, pp. 244–252.

[25] A. Boldyreva, N. Chenette, Y. Lee, and A. Oneill, "Order-preserving symmetric encryption," in *EUROCRYPT*. Springer, 2009, pp. 224–241.

[26] P. Wang and C. Ravishankar, "Secure and efficient range queries on outsourced databases using rp-trees," in *ICDE*, 2013, pp. 314–325.

[27] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *ICPS*, vol. 2, no. 1, pp. 46–55, 2003.

[28] Y. Zhu, D. Ma, D. Huang, and C. Hu, "Enabling secure location-based services in mobile cloud computing," in *Proceedings of the second ACM SIGCOMM workshop on Mobile cloud computing*. ACM, 2013, pp. 27–32.

[29] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *ICPS*. IEEE, 2005, pp. 88–97.

[30] C. A. Ardagna, M. Cremonini, E. Damiani, S. D. C. Di Vimercati, and P. Samarati, "Location privacy protection through obfuscation-based techniques," in *Data and Applications Security XXI*. Springer, 2007, pp. 47–60.

[31] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st international conference on Mobile systems, applications and services*. ACM, 2003, pp. 31–42.

[32] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: query processing for location services without compromising privacy," in *VLDB*, 2006, pp. 763–774.

Lichun Li received the PhD. degree in computer science from Beijing University of Posts and Telecommunications in 2009, master degree in communication and information systems in China Academy of Telecommunication Technology in 2006, and bachelor degree in information engineering in in Beijing University of Posts and Telecommunications in 2002. He is currently a Postdoctoral Research Fellow with the INFINITUS laboratory, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests include privacy and security in cloud and big data.

Rongxing Lu (S'09-M'11-SM'15) received the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2006, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2012. From May 2012 to April 2013, he was a Postdoctoral Fellow with the University of Waterloo. Since May 2013, he has been an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests include computer network security, mobile and wireless communication security, and applied cryptography. Dr. Lu was the recipient of the Canada Governor General Gold Metal.

Huang Cheng received his B.Eng from Xidian University, China, in 2013, and is currently a Project Officer with the INFINITUS laboratory at the School of Electrical and Electronic Engineering, Nanyang Technological University. His research interests are in the areas of applied cryptography, cyber security and privacy.