On Location Privacy in Fingerprinting-based Indoor Positioning System: An Encryption Approach

Wenlu Wang
wenluwang@auburn.edu
Auburn University

Zhitao Gong
gong@auburn.edu
Auburn University

Ji Zhang
jizhang@auburn.edu
Auburn University

Hua Lu
luhua@cs.aau.dk
Aalborg University

Wei-Shinn Ku
weishinn@auburn.edu
Auburn University

ABSTRACT

Due to the inadequacy of GPS signals in indoor spaces, Indoor Positioning Services (IPSs) have drawn great attention. The popular smartphone localization technique relies on a centralized server to achieve localization, allowing the server to acquire a user's location in fine granularity. To ensure the privacy of IPS users, we propose an *Encrypted Indoor Positioning Service (EIPS)* model that protects users' privacy from the centralized server and maintains localization accuracy simultaneously. Our *EIPS* model enables users to encrypt and decrypt their query through an *Encryption and Decryption Server (EDS)* bi-directionally in a commutative way, so the users' locations remain private to both *EIPS* and *EDS*. We also propose *Query Split*, *Artificial Dimensions* and *Columns* to prevent Known Plaintext Attack (KPA). Our analytical and experimental evaluations show that our model is resilient to a variety of privacy attacks without loss of efficiency and accuracy.

CCS CONCEPTS

• **Information systems** → **Location based services**; • **Security and privacy** → **Management and querying of encrypted data**.

KEYWORDS

Indoor Positioning System, Privacy-Preserving Queries

ACM Reference Format:

Wenlu Wang, Zhitao Gong, Ji Zhang, Hua Lu, and Wei-Shinn Ku. 2019. On Location Privacy in Fingerprinting-based Indoor Positioning System: An Encryption Approach. In *27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '19)*, November 5–8, 2019, Chicago, IL, USA. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3347146.3359081>

1 INTRODUCTION

People spend a significant amount of time in indoor spaces, such as subway stations, shopping centers, libraries, airports, and office

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SIGSPATIAL '19, November 5–8, 2019, Chicago, IL, USA

© 2019 Association for Computing Machinery.

ACM ISBN 978-1-4503-6909-1/19/11...\$15.00

<https://doi.org/10.1145/3347146.3359081>

Figure 1: Conventional Indoor Positioning System (IPS)

buildings; therefore, location-related queries [22, 23] are of high interest and importance in indoor spaces [20, 21].

Contrasted to the satellite-based Global Positioning Systems (GPS), users often rely on smartphones to acquire locations (positions) through cloud-based Indoor Positioning Services (IPS). Such services utilize geolocation databases containing wireless fingerprints, magnetic signals, or light signals recorded from pre-selected reference locations off-line. Accordingly, an *IPS* determines the location of a user by comparing his/her online geolocation data with those of reference locations in the database. Taking WiFi routing as an example, whose database is a radio map, where each reference entry is a ⟨fingerprint, reference location⟩ tuple, the user needs to provide his/her online fingerprint (signal strengths) to compare against the radio map (Section 3.1). Figure 1 shows an example of a conventional WiFi router based IPS: upon request, the IPS receives the user's fingerprint and compares it against the radio map in order to find similar signal fingerprints using *kNN* [15] searching; then the *k* fingerprints closest to the user's fingerprint are returned, and the user is localized by the (weighted) convex combination of the *k* reference locations (associated with the *k* fingerprints).

As shown in Figure 1, an *IPS* has access to a user's fingerprints and locations, even in fine granularity, incurring severe privacy concerns. An *IPS* is able to monitor the precise trajectories of users without their knowledge, and individual trajectories could possibly reveal their interests, habits, and interactions with other people. A significant drawback of such cloud-based services is that the localization operations are uncontrollable by users since the geolocation databases are stored and operated on the server-side. An intuitive privacy-preserving IPS model is to load the whole radio map in memory and perform the localization (e.g., *kNN* search) on the smartphone; however, smartphone microprocessors lag behind the powerful computational resources available in servers. Such design incurs an extensive memory usage and execution inefficiencies, thus inapplicable in practice.

In general, a secure IPS needs to address two **challenges**: I. An *IPS* should not receive a user's query (e.g., fingerprints) in plain text; II. An *IPS* should not be able to infer a user's location based on

the query. For challenge I, we use encrypted databases that support encrypted query; for challenge II, our proposed communication scheme resists known plain text attack. In summary, we tackle the technical challenges of enabling users to acquire their positions from IPS while preserving their privacy.

An intuitive privacy-preserving IPS model with WiFi routing *TVM* is to retrieve b reference entries ($b > k$) from the radio map and pass all of them to the user, empowering users to perform nearest neighbor comparisons by themselves. Such strategy achieves a $\mathcal{K}$ anonymous solution, since the redundant reference entries guarantee at least $\mathcal{K}$ positive matches at the server-side. However, there are two weaknesses: (1) redundant fingerprint comparisons are computationally inefficient and memory consuming on mobile devices (e.g., smartphones) with a large $\mathcal{K}$; (2) *TVM* requires extra effort for continuous queries, while continuous queries are more common than snapshot queries.

Inspired by the drawbacks of an IPS, we propose to preserve the users' privacy with the help of a centralized server (who holds the encryption key). We alleviate the computation burden of smartphones, and transfer the workload to centralized servers, even though such transformation incurs extra I/Os, which is minimum compared with high computation cost on smartphones. Under the assumption that the centralized server which holds the encryption key is neutral, we can safely assume the users' locations remain private to users without loss of accuracy and efficiency.

We propose an **Encrypted Indoor Positioning Service (EIPS)**. The fingerprint of the user is encrypted before sent to *EIPS* (the encrypted radio map is stored in *EIPS* as well). To keep the encryption key inaccessible from users, we propose a communication scheme through an independent **Encryption and Decryption Server (EDS)**, whose functionality is to encrypt a query and decrypt a query result. There are two challenges in our scheme:

- i. *EDS* has access to users' location information through encryption and decryption.
- ii. Symmetric encryption through *EDS* introduces Known Plain-text Attack (KPA).

For the first challenge, we apply commutative ciphers and take advantage of their commutative properties to support the bi-directional communications between a user and *EDS* such that the user's real location remains hidden from *EDS*. For the second challenge, we take advantage of the critical property of Asymmetric Scalar-product Preserving Encryption (ASPE) [24]: *scaling the query does not affect localization accuracy*. We further enhance the design of ASPE to prevent KPA by supporting Artificial Dimensions and Columns.

Our contributions are summarized as follows: (1) we design a privacy-preserving model for fingerprinting-based IPS; (2) our model achieves high security without loss of accuracy and efficiency; (3) we conduct analyses as well as experiments to evaluate the performance and privacy protection capability of our approach.

2 RELATED WORK

2.1 Location Privacy-Preserving Techniques

Existing privacy-preserving techniques for location-based services are based on several basic concepts. *Sanitized locations approach* uses fake locations to protect users' privacy. *Spatial cloaking approach* [3, 6, 13, 16] blurs the user's location into a cloaked area.

However, it can only infer an approximate location, and such a strategy often applies in outdoor spaces. *Space transformation approach* [9, 11] transforms the user's real location into another space where their spatial relationships are maintained, such as the Hilbert space filling curve [10] and the Z curve. In a transformed space, points and distances are encoded. Some space transformation approaches incur transformation errors. *$\mathcal{K}$ -anonymity approach* achieves the goal that a user is blurred among at least $\mathcal{K}$ users [17, 18], so the probability of recognizing the user is lower than $\frac{1}{\mathcal{K}}$. Some strategies use a set of fake locations [12, 25] to achieve $\mathcal{K}$ -anonymity. For example, SpaceTwist [25] starts with a location different from the user's actual location; then its neighbors are retrieved incrementally until the query is answered. However, the adversary of the SpaceTwist system can infer a privacy region of the real user, which means the adversary can eventually infer an approximate location of the user. Some other $\mathcal{K}$ -anonymous solutions [1, 2] rely on historical data, and derive trajectories from real trajectories. Private information retrieval (PIR) [5] supports private location-dependent queries via cryptographic techniques. However, PIR targets for information retrieval rather than k NN search. *Encryption approach* Fully Homomorphic Encryption (FHE) [4] is useful to prevent server-side information leakages. However, it is impractical for real-world applications due to its high latency [19].

Secure location-based services have been well studied; however, the techniques of private *outdoor* location-based services [8] cannot apply to private *indoor* positioning systems directly.

2.2 Privacy-preserving Indoor Positioning

There are multiple indoor localization solutions, including Bluetooth, RFID, sensor networks, wireless LANs [7].

Temporal Vector Map (*TVM*) [14] is a $\mathcal{K}$ -anonymity indoor positioning algorithm using Wi-Fi radio map. The positioning of *TVM* is achieved by searching k nearest neighbors in the radio map records and using the weighted average reference locations as the user's location. *TVM* takes advantage of the bloom filter to perform localization without revealing the user's location to the localization server by retrieving part of the database (denoted as *pRM*) instead of the actual records. IPS is not able to deduct the real location of the user from *pRM*, due to the fact that $\mathcal{K}$ positive matches can be inferred from *pRM*. In order to maintain efficiency, *TVM* utilizes Hadoop and HBase for acceleration.

Cloaking-based privacy-preserving IPS [13] is another line of work. The authors propose an Anonymizing Spatial Region (ASR) involving $\mathcal{K}-1$ other users based on the hierarchical structure of the indoor spaces, which is a $\mathcal{K}$ -anonymous solution as well. One drawback of such a method is that $\mathcal{K}-1$ fake locations and the real location usually are in the same cloaking region, which has a low security level. Existing $\mathcal{K}$ -anonymous solution puts part of computation load to users and has to fake $\mathcal{K} - 1$ routes for continuous queries.

3 PRELIMINARIES

3.1 Fingerprinting-based Indoor Positioning

Wi-Fi fingerprints is a vector composed of Received Signal Strength (RSS) value ($[\lambda_1, \lambda_2, \dots, \lambda_n]$) from n Wi-Fi Access Points (AP). In an off-line phase, a logging application records the Wi-Fi fingerprints

at many reference locations (usually every 2-3 meters) in an indoor space, which are denoted as coordinate (l_x, l_y) . Supposing there are N reference locations, a *radio map* stores N records of the fingerprints and reference locations as $[\lambda_1^p, \lambda_2^p, \dots, \lambda_n^p, l_x, l_y]$.

Initially, users send their fingerprints to IPS (shown in Figure 1 as an inbound arrow). An IPS compares their online RSS fingerprints with the radio map using signal distance to infer their location. The signal distance between the user's RSS vector $q = [\lambda_1^q, \lambda_2^q, \dots, \lambda_n^q]$ and an RSS vector in the radio map $p = [\lambda_1^p, \lambda_2^p, \dots, \lambda_n^p]$ is computed as $SD(q, p)$

$$SD(q, p) = \left(\sum_{i=1}^n |\lambda_i^q - \lambda_i^p|^2 \right)^{\frac{1}{2}}$$

The nearest neighbor is the point with the smallest signal distance. After a k NN search among the radio map, IPS returns k reference locations to users (shown in Figure 1 as outbound arrow). If k nearest neighbors are found, the weighted average of the coordinates of k reference locations can be used as the estimation of the user's real location. Refer to the equation.

$$(l_x^u, l_y^u) = \left(\sum_{i=1}^k w_i l_{xi}, \sum_{i=1}^k w_i l_{yi} \right)$$

3.2 Privacy-Preserving k NN Search

The goal of private k NN is to protect the privacy of users while accurately searching for k NN from the encrypted data. Under the assumptions that the attacker cannot observe the plaintext queries in all cases, and the user can hold the encryption key, an Asymmetric Scalar-product Preserving Encryption (ASPE) approach [24] computes k nearest neighbors within an encrypted database. It requires a strong assumption that the attacker is not allowed to disguise as a user and submit queries to the database, which is too strong for real-world scenarios.

We provide a brief introduction of the ASPE method as follows (symbols are slightly changed to be consistent in our paper). Assuming v is an RSS record $v = [\lambda_1^p, \lambda_2^p, \dots, \lambda_n^p]$ with dimension n , p is a revision of v to support ASPE, and q is a query that is able to query the database. M is an $(n+1) \times (n+1)$ encryption matrix.

$$\begin{cases} p = (v^T, -0.5||v||^2)^T \\ E^{-1}(p) = M^T p \\ q = (v^T, 1)^T \\ E(q) = M^{-1} q \end{cases}$$

The database stores all $E^{-1}(p)$ records, and the k NN search is performed by computing $(E^{-1}(p_1) - E^{-1}(p_2)) \cdot E(q)$. If $(E^{-1}(p_1) - E^{-1}(p_2)) \cdot E(q) > 0$, p_1 is closer to q . Otherwise, p_2 is closer to q . Due to the uniqueness of such k NN search operation, an important property of ASPE is **dynamic scaling of q does not affect the localization accuracy**.

The weaknesses of ASPE are (1) infeasible strong assumption and (2) vulnerable towards KPA. We relax the assumption that the attacker cannot disguise as a user, and maintain the security by empowering a centralized server to hold the encryption key. We also make our modifications of ASPE to resist KPA, which are detailed in Section 5.

4 SYSTEM OVERVIEW

4.1 EIPS Privacy Preserving Scheme

Symbols	Description
q	an EIPS query
p	a radio map record
$[\lambda_1^q, \dots, \lambda_n^q]$	RSS fingerprints of a query
$[\lambda_1^p, \dots, \lambda_n^p]$	RSS fingerprints of a radio map record
l_x, l_y	the coordinate of a reference location (RL_i)
$[q_s^1, \dots, q_s^l]$	l split queries
$[q^1, \dots, q^l]$	l query partitions after dynamic scaling
$\tilde{q}^i$	i -th split of query with inserted artificial dimensions
$\tilde{p}$	a radio map record with inserted artificial dimensions
$\tilde{P}$	k nearest records with inserted artificial dimensions
$\mathcal{F}_t/\mathcal{F}_u$	Inbound cipher by EDS/users
DK	Downsize Key
RL	$[RL_1, \dots, RL_k]$ a matrix composed of k reference locations
$+D/-D$	Insert/Remove Artificial Dimensions
$+C/-C$	Insert/Remove Artificial Columns
$\mathcal{R}_t/\mathcal{R}_u$	Outbound cipher by EDS/users

Table 1: Notation Table

We propose an Encrypted IPS (EIPS) system involving a trusted server called Encryption and Decryption Server (EDS), whose functionality is holding the encryption key of the encrypted database. EIPS model is composed of two compatible schemes: Privacy-Preserving Scheme shown in Figure 2(a) and KPA Resistance Scheme shown in Figure 2(b).

EDS With the help of EDS, a user has no access to the central database's encryption key, nor can the user infer the decryption key held by EDS. We define “inbound” cipher as the process of EIPS receiving query information from users (shown in Figure 2 in green). The “outbound” cipher represents the process of EIPS finishing querying the encrypted database, and returning results to users (shown in Figure 2 in orange). We detail our **Privacy-Preserving Design** (Figure 2a) as follows

I. *Inbound cipher* $\mathcal{F}$ is designed to encrypt the query for privacy-preserving purpose. We define the encryption function by EDS as $\mathcal{F}_t$, and by users as $\mathcal{F}_u$ ($\mathcal{F}_u^{-1}$ is denoted as the decryption function of $\mathcal{F}_u$). $\mathcal{F}_u$ and $\mathcal{F}_u^{-1}$ are conducted by users to protect their privacy. $\mathcal{F}_t$ is the inbound encryption.

II. *Outbound cipher* $\mathcal{R}$ is designed to return k nearest neighbor records to users without loss of privacy. $\mathcal{R}_u$ and $\mathcal{R}_u^{-1}$ are the encryption and decryption function performed by users to protect their privacy. As with the previous setting, we denote $\mathcal{R}_t^{-1}$ as the outbound decryption function of reference locations performed by EDS.

In summary, as shown in Figure 2a, the functionality of $\mathcal{F}$ and $\mathcal{R}$ is to perform k NN search securely without leaking users' privacy to EDS or EIPS.

EIPS is responsible for storing encrypted databases and operating on encrypted databases. As introduced in Section 2, each database has a unique encryption key (held by EDS), and the database will be preprocessed (encrypted) by EDS, and stored in EIPS. EIPS needs to perform k NN search on the encrypted radio map.

In our EIPS setting, EDS holds the encryption key instead of users. EDS is responsible for generating the encrypted radio map

with the unique encryption key and sending the encrypted radio map to EIPS during the **Initialization Stage** before the starting of the service. In this way, EIPS only has access to the encrypted database without knowing the key nor the plaintext database.

4.2 Attack Model

There are two possible ways to attack an IPS: (1) Intentionally modify the database stored in IPS, or modify the response returned to users, called *active attacks*; (2) Another type of attack is *passive attacks*. IPS server can store multiple query records and infer critical information.

Our scheme resists active attack since the database is encrypted by EDS, and EIPS cannot tamper with the encrypted databases or ciphertext without affecting the accuracy of IPS services. For passive attacks, we define possible attacks in our proposed scheme.

Level 1 attacks

Attack by EDS. Our assumption is that EDS is a neutral server, who plays a similar role as a Certificate Authority (CA) or an Attribute Authority (AA) in the computer security field. We assume EDS would not collaborate with users or IPS. However, EDS may try to retrieve the users' real locations by either KPA or comparing against the radio map.

Attack by Users. A user tries to crack the encryption key using KPA.

Attack by EIPS. To prevent EIPS from acquiring the users' locations, we only store the encrypted radio map in EIPS. A user sends encrypted queries to EIPS to force executions on encrypted radio map. In this way, EIPS can either serve the user with encrypted radio maps or refuse to serve. We assume EIPS has no access to the original radio map.

Level 2 attacks

Collaborative Attack. We propose the highest level attack, where users and EIPS can collaborate. Even though this attack is rare in practice, EDS refreshes the encrypted database in EIPS periodically to prevent collaboration attacks. Since EDS encrypts the query in real-time, EIPS has to accept the refreshed encrypted radio map to support positioning tasks.

4.3 KPA Resistance

We propose the following techniques to prevent KPA, as shown in Figure 2b, and the details of these techniques are formally defined in Section 5.

Figure 2: The two building blocks of EIPS model.

Query Split Since EDS has access to the plaintext of the radio map, a user splits his/her query (RSS) and scales each part randomly to prevent EDS from retrieving the plaintext of the query (Sec 5.1.1).

Dynamic Scaling Due to the unique property of *ASPE* introduced in Section 3.2, EDS dynamically scales each query to prevent KPA from users (Sec 5.1.2).

During inbound communications, Query Split technique prevents Level 1 attack by EDS, while Dynamic Scaling technique prevents Level 1 attack by users.

Artificial Dimensions/Columns Since our communication model adopts commutative ciphers, we adopt a padding strategy to prevent KPA. Padding is a conventional encryption practice, which has been widely adopted, such as cryptographic hash functions, block cipher, and RSA. A strong padding scheme should prevent the attackers from exploiting the mathematical structure of the key by manipulating the plaintext, and we ensure such constraints by forcing the EDS to add padding and remove the padding and preventing users from acquiring plaintext.

The challenges of such scheme include: given the commutative nature of our model between EDS and users, we empower both EDS and users to pad the ciphertext they receive to ensure privacy, and both of them should receive the correct content that is not affected by the other side's padding operation. To ensure security as well as accuracy, we empower the EDS to embed haphazard (non-constant) artificial dimensions to ciphertext (Sec 5.1.2), and the users to embed haphazard artificial columns to ciphertext (Sec 5.3.1).

EDS holds the ground truth of which dimensions are artificial and which dimensions embed the reference locations. The users hold the ground truth of artificial columns, where their true location remains private. Correspondingly, EDS will remove artificial dimensions (Sec 5.3.2) and users will remove artificial columns (Sec 5.3.3). During outbound communications, inserting Artificial Dimensions prevents Level 1 attack by users, while inserting Artificial Columns prevents Level 1 attack by EDS.

5 EIPS DESIGN

Our design combines the perspectives of both Figure 2a and Figure 2b, which are perfectly coupled, and presented in Figure 3. First, a query initiated by a user goes through the inbound encryption

Figure 3: EIPS temporal workflow composed of Figure 2(a) and (b).

$\mathcal{F}_t$ of EDS (Sec 5.1), then the user sends its encrypted query to EIPS, where a **kNN search** is performed with reference locations (in cipher text) returned to the user (Sec 5.2). Finally, the ciphertext goes through the outbound decryption $\mathcal{R}_t^{-1}$ of EDS (Sec 5.3).

5.1 Inbound Scheme

There are n AP values for each query $\lambda^q = [\lambda_1, \lambda_2, \dots, \lambda_n]$, which means a fingerprint is an n -dimension vector. As we mentioned in Section 3, indoor localization relies on k NN search among the radio map. We (users) first revise the query:

$$q = [\lambda_1^q \dots \lambda_n^q \ 1]^T$$

Accordingly, EDS revises each radio map record ($\lambda^p = [\lambda_1^p \dots \lambda_n^p]^T$) to p in an effort to support k NN search by q (l_x, l_y is the coordinate of fingerprint λ^p)

$$p = [\lambda_1^p \dots \lambda_n^p \ -0.5||\lambda^p||^2 \ l_x \ l_y]^T$$

We define $\otimes$ as paired element-wise multiplication. If the operands are two vectors $A = [a_1 \dots a_n]$ and $B = [b_1 \dots b_n]$, $A \otimes B = [a_1 \cdot b_1 \dots a_n \cdot b_n]$. If the operands are an array $A = [a_1 \dots a_n]$ and a single item b , $A \otimes b = [a_1 \cdot b, \dots, a_n \cdot b]$. Here $(\cdot)$ refers to matrix multiplication. We also define the sum of an array $\sum A = \sum_{i=1}^n a_i$.

5.1.1 User Encryption. To prevent EDS from acquiring the original content of the fingerprint, we propose an encryption function conducted by users: $\mathcal{F}_u$ randomly (1) *splits* the original query q to l parts, and (2) *scales* the i_{th} part with a random variable α_i ($i = [1..l]$) before sending it to EDS, denoted as **Query Split**. The key of $\mathcal{F}_u$ is defined as V .

$$V = [V_1 \dots V_l] = [\alpha_1 I_{n+1} R_1 \dots \alpha_l I_{n+1} R_l]$$

where

$$R_i = \begin{bmatrix} r_1^i & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & r_{n+1}^i \end{bmatrix}_{(n+1) \times (n+1)}$$

$$\sum_{i=1}^l R_i = I_{n+1}$$

$$V_i = \alpha_i I_{n+1} R_i = \begin{bmatrix} \alpha_i r_1^i & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \alpha_i r_{n+1}^i \end{bmatrix}_{(n+1) \times (n+1)}$$

The purpose of R_i is to split the query, and α_i is to scale the query split randomly. r_j^i $i=[1..l]$ $j=[1..n+1]$ are random variables (i is the i_{th} partial query, j is the j_{th} dimension in the query). The element-wise sum of R_i $i=[1..l]$ is an identity matrix I .

Definition (Query Split) Operation $[R_1 \dots R_l] \otimes q$ splits query q to l parts, and split query is defined as $[q_s^1 \dots q_s^l]$. Correspondingly, the sum of the l splits is the original query q .

$$[q_s^1 \dots q_s^l] = [R_1 \dots R_l] \otimes q = [R_1 q \dots R_l q]$$

$$\sum [q_s^1 \dots q_s^l] = \sum_{i=1}^l R_i \otimes q = I_{n+1} q = q$$

The content of each split query is

$$q_s^i = R_i q = [r_1^i \lambda_1^q \dots r_n^i \lambda_n^q \ r_{n+1}^i]^T$$

After scaling, each partial query is

$$q^i = \alpha_i q_s^i = [\alpha_i r_1^i \lambda_1^q \dots \alpha_i r_n^i \lambda_n^q \ \alpha_i r_{n+1}^i]^T$$

We define the inbound encryption/decryption function of users using Query Split as

$$\mathcal{F}_u(q) = \mathcal{F}_u(q, V) = V \otimes q = [\alpha_1 q_s^1 \dots \alpha_l q_s^l] = [q^1 \dots q^l]$$

$$\mathcal{F}_u^{-1}(x) = \sum [\frac{1}{\alpha_1} I_{n+1} \dots \frac{1}{\alpha_l} I_{n+1}] \otimes x$$

For easy presentation, we use $\mathcal{F}_u(x)$ to present $\mathcal{F}_u(x, V)$. $\mathcal{F}_u^{-1}$ is to reverse scaling, followed by a summation.

Example We present an example of Query Split and its recovery with $n = 1$, $l = 2$.

$$q = [2, 1]^T$$

----- Encryption -----

$$[R_1 \ R_2] = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$[\alpha_1 I_2 \ \alpha_2 I_2] = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \quad \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$$

$$V = [\alpha_1 I_2 R_1 \ \alpha_2 I_2 R_2]$$

$$\mathcal{F}_u(q, V) = V \otimes q = \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \quad \begin{bmatrix} 6 \\ 0 \end{bmatrix}$$

----- Decryption -----

$$[\frac{1}{\alpha_1} I_2 \ \frac{1}{\alpha_2} I_2] = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \quad \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & \frac{1}{3} \end{bmatrix}$$

$$\mathcal{F}_u^{-1}(\begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \quad \begin{bmatrix} 6 \\ 0 \end{bmatrix}) = \sum \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \quad \begin{bmatrix} \frac{1}{3} & 0 \\ 0 & \frac{1}{3} \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \quad \begin{bmatrix} 6 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Purpose of *Query Split* is two-fold: (1) EDS receives l scaled partial queries and is unable to derive the original query with V unknown; (2) l remains unknown to EDS, to prevent EDS to infer which AP (dimension) has higher signal strength; a user could fake signal strength at any AP and send false queries to fool EDS.

5.1.2 EDS Encryption. EDS performs the following operations

- Insert artificial dimensions to q^i ($q^i \rightarrow^{+D} \hat{q}^i$) $i \in [1..l]$
- EDS encryption $\mathcal{F}_t(\hat{q}^i)$
- Dynamic scaling of encrypted query partitions to prevent KPA

a. Insert(+) Artificial Dimensions EDS extends the plaintext split query q^i as $q^i[n+2] = 0, q^i[n+3] = 0$ to support k NN search at EIPS (assuming $p[n+2] = l_x, p[n+3] = l_y$), then insert artificial dimensions.

To support k NN search on EIPS, we also define how to insert the corresponding artificial dimensions in p (our example shows an appendix padding). EDS has to determine the dimensions to insert artificial dimensions and pad every data record p in the radio map at an earlier stage since it has to send the complete encrypted radio map to EIPS before starting the service. There are two types of artificial dimensions (the operation of p applies to all the data record in the radio map), defined in Figure 2 (assuming we are inserting artificial dimension at dimension j).

We define $\hat{q}^i$ as query split q^i after artificial dimensions inserted:

$$[\alpha_i r_1^i \lambda_1^q \dots \alpha_i r_n^i \lambda_n^q \ \alpha_i r_{n+1}^i \ 0 \ 0 \ 0 \dots 0 \ e_{n+3}^{q^i} \dots e_j^{q^i}]^T$$

where $e_j^{q^i}$ is dynamic per split q^i , and varies per dimension j .

Correspondingly, the data record in the radio map needs to insert dimensions to support k NN. First we deduct the format of a query

Type I	q^i	Insert random variable $e_j^{q^i}$ at dimension j .
	p	Insert random variable e_j^p at dimension j , where all the records in the radio map at dimension j share the same e_j^p .
Type II	q^i	Insert 0 in q^i at dimension j .
	p	Insert random variable r_j^p in each data record p at dimension j and each record in the radio map could be different.

Table 2: Two types of artificial dimensions.

in its ciphertext received by EIPS. Users receive the ciphertext of $\hat{q}^i$, after reversed scaling, $\frac{1}{\alpha_i} I_d \hat{q}^i$ is transformed as

$$[r_1^i \lambda_1^q \dots r_n^i \lambda_n^q \dots r_{n+1}^i \lambda_{n+1}^q \dots 0 \dots 0 \dots \frac{e_{n+4}^{q^i}}{\alpha_i} \dots \frac{e_j^{q^i}}{\alpha_i}]^T$$

The sum of reversed scaled query splits (for an arbitrary Type I

artificial dimension j , $\sum_{i=1}^l \frac{e_j^{q^i}}{\alpha_i} = e_j^q$ is

$$\hat{q} = [\lambda_1^q \dots \lambda_n^q \ 1 \ 0 \ 0 \ 0 \dots 0 \ e_{n+4}^q \dots e_j^q]^T$$

$\hat{q}$ is the encrypted query received by EIPS in its plaintext. Accordingly, EDS should pad each data record of the radio map in plaintext $\hat{p}$ as

$$[\lambda_1^p \dots \lambda_n^p \ -0.5||\lambda^p||^2 \ l_x \ l_y \ r_{j+1}^p \dots r_d^p \ e_{n+4}^p \dots e_j^p]^T$$

where $||\lambda^p||^2 = \lambda^p \cdot \lambda^p$.

EDS selects the specific dimensions to insert Type I and II artificial dimensions, which are inserted to the radio map during the Initialization Phase. During services, EDS inserts random variables (Type I) or 0 (Type II) at pre-defined dimensions in q^i .

The advantages of artificial dimensions are three-fold: (1) Since EDS performs padding for both p (Initialization Phase) and q (EDS Encryption), padding can take place in any dimension (our example shows appendix padding). In other words, the sequence of p and q can be shuffled, as long as they are shuffled in the same way (controlled by EDS); (2) Since users (EIPSs) receive *ciphertext* of $\hat{q}^i$ ($\hat{q}$), thus they cannot infer the dimensional information of artificial dimensions; and (3) KPA resistance.

b. Encryption Function We define $M_{d \times d}$ an invertible matrix (columns of M are linear independent), M^{-1} the inverse of M , and γ a random variable. We define $\mathcal{F}_t$ and $\mathcal{F}_t^{-1}$ as:

$$\mathcal{F}_t(x) = \mathcal{F}_t(x, M^{-1}) = M^{-1} \cdot x \quad \mathcal{F}_t^{-1}(x) = \mathcal{F}_t^{-1}(x, M^T) = M^T \cdot x$$

In summary, EDS encrypts each padded query split q^i to $\hat{q}^i$, which is encrypted with $\mathcal{F}_t(\hat{q}^i) = \mathcal{F}_t(\hat{q}^i, M^{-1}) = M^{-1} \cdot \hat{q}^i$. Correspondingly, each data record in the radio map $\hat{p}$ will be encrypted as $\mathcal{F}_t^{-1}(\hat{p}) = \mathcal{F}_t^{-1}(\hat{p}, M^T) = M^T \cdot \hat{p}$ as stored in EIPS.

c. Dynamic Scaling EDS multiplies a Dynamic Scaling factor γ to each encrypted query $\mathcal{F}_t(\hat{q}^i)$, γ is *shared among l query partitions and changed per query*.

$$\mathcal{F}_t(\hat{q}^i) \rightarrow \gamma \mathcal{F}_t(\hat{q}^i)$$

$$\mathcal{F}_u^{-1} \cdot \mathcal{F}_t \cdot \mathcal{F}_u(q) = \sum_{i=1}^l \frac{1}{\alpha_i} \cdot I_d \cdot \gamma \mathcal{F}_t(\hat{q}^i) = \gamma \mathcal{F}_t(\hat{q})$$

Here γ is generated by EDS dynamically, which is not part of the encryption key. Our dynamic scaling scheme does not alter the result with respect to the kNN search.

KPA resistance As we know, matrix transformation does not resist KPA, and encrypting through EDS introduces plaintext q^i and ciphertext $\mathcal{F}_t(q^i)$ pair to users easily. To prevent KPA by users, EDS has the following strategies:

- *Dynamic Scaling*. By introducing randomness to each query, the ciphertext is altered randomly. Due to the alteration of the ciphertext, the cracked encryption key using KPA does not reveal any information of the plaintext.
- *Artificial Dimensions*. By adding artificial dimensions in anonymous dimensions, EDS not only increases the dimensions of the plaintext but also introduces more unknowns to the encryption process. In this way, a user fails to create a (plaintext, ciphertext) pair.

5.2 Enhanced kNN Search

Upon receiving of the cipher text, a user decrypts the ciphertext by $\mathcal{F}_u^{-1}$ (Sec 5.2.1), and sends the encrypted query to EIPS (Sec 5.2.2). Since reference locations are embedded in the encrypted radio map, EIPS performs a kNN search and retrieve k reference locations (Sec 5.2.3). Note that, a weighted kNN search is also possible depending on the pre-setting of the EIPS.

5.2.1 User Decryption. From EDS, a user receives $\gamma \mathcal{F}_t(\hat{q}^i)_{i=[1..l]}$, which is further decrypted by user decryption $\mathcal{F}_u^{-1}$ (a reversed scaling followed by a summation). Here we prove that $\mathcal{F}_u$ and $\mathcal{F}_t$ are commutative

PROOF.

$$\begin{aligned} & \mathcal{F}_u^{-1} \cdot \gamma \mathcal{F}_t \cdot \mathcal{F}_u(q) \\ &= \mathcal{F}_u^{-1} \cdot \gamma \mathcal{F}_t(\hat{q}^i)_{i=[1..l]} \\ &= \gamma M^{-1} \sum_{i=1}^l [r_1^i \lambda_1^q \dots r_n^i \lambda_n^q \dots r_{n+1}^i \lambda_{n+1}^q \dots 0 \dots 0 \dots \frac{e_{n+4}^{q^i}}{\alpha_i} \dots \frac{e_j^{q^i}}{\alpha_i}]^T \\ &= \gamma M^{-1} \left[\sum_{i=1}^l r_1^i \lambda_1^q \dots \sum_{i=1}^l r_n^i \lambda_n^q \dots \sum_{i=1}^l r_{n+1}^i \lambda_{n+1}^q \dots 0 \dots 0 \dots \sum_{i=1}^l \frac{e_{n+4}^{q^i}}{\alpha_i} \dots \sum_{i=1}^l \frac{e_j^{q^i}}{\alpha_i} \right]^T \\ &= \gamma M^{-1} [\lambda_1^q \dots \lambda_n^q \ 1 \ 0 \ 0 \ 0 \dots 0 \ e_{n+4}^q \dots e_j^q]^T \\ &= \gamma \mathcal{F}_t(\hat{q}) \end{aligned} \quad \square$$

5.2.2 kNN search revisit. We now explain how EIPS supports kNN search on an encrypted database. We define

$$f = (\mathcal{F}_t^{-1}(\hat{p}_1) - \mathcal{F}_t^{-1}(\hat{p}_2)) \cdot \gamma \mathcal{F}_t(\hat{q})$$

We prove in the following equation that the sign of f indicates spatial dominance

PROOF.

$$\begin{aligned} f &= (\mathcal{F}_t^{-1}(\hat{p}_1) - \mathcal{F}_t^{-1}(\hat{p}_2)) \cdot \gamma \mathcal{F}_t(\hat{q}) \\ &= (M^T \hat{p}_1 - M^T \hat{p}_2) \cdot \gamma M^{-1} \hat{q} \\ &= \gamma \left[\lambda_1^{p1} - \lambda_1^{p2} \dots 0.5||\lambda^{p2}||^2 - 0.5||\lambda^{p1}||^2 \ l_{x1} - l_{x2} \ l_{y1} - l_{y2} \right. \\ &\quad \left. r_{j+1}^{p1} - r_{j+1}^{p2} \dots r_d^{p1} - r_d^{p2} \ e_{n+1}^p - e_{n+1}^p \dots e_j^p - e_j^p \right]^T \\ &\quad [\lambda_1^q \dots 1 \ 0 \ 0 \ 0 \dots 0 \ e_{n+1}^q \dots e_j^q] \\ &= \gamma \left((\lambda_1^{p1} - \lambda_1^{p2}) * \lambda_1^q + \dots - 0.5(||\lambda^{p1}||^2 - ||\lambda^{p2}||^2) * 1 \right. \\ &\quad \left. + (l_{x1} - l_{x2}) * 0 + (l_{y1} - l_{y2}) * 0 \right. \\ &\quad \left. + (r_{j+1}^{p1} - r_{j+1}^{p2}) * 0 + \dots + (r_d^{p1} - r_d^{p2}) * 0 \right) \end{aligned}$$

$$\begin{aligned}
& + (e_{n+1}^p - e_{n+1}^q) * e_{n+1}^q + \dots + (e_d^p - e_d^q) * e_d^q \\
& = \gamma(\lambda^{p_1} \cdot \lambda^q - \lambda^{p_2} \cdot \lambda^q - 0.5||\lambda^{p_1}||^2 + 0.5||\lambda^{p_2}||^2) \\
& = 0.5\gamma(d(\lambda^{p_2}, \lambda^q) - d(\lambda^{p_1}, \lambda^q))
\end{aligned}$$

Artificial dimensions do not affect the correctness of k NN since

$$\begin{aligned}
& (r_{j+1}^{p_1} - r_{j+1}^{p_2}) * 0 + \dots + (r_d^{p_1} - r_d^{p_2}) * 0 = 0 \\
& (e_{n+1}^p - e_{n+1}^q) * e_{n+1}^q + \dots + (e_d^p - e_d^q) * e_d^q = 0
\end{aligned}$$

If $d(p_2, q) > d(p_1, q)$ ($f > 0$), p_1 is closer to q . Otherwise, p_2 is closer to q .

Each encrypted radio map record is stored as $\mathcal{F}_t^{-1}(\hat{p})$ in EIPS. The k nearest neighbor could be retrieved with a linear scan. We denote the k records as $\mathcal{F}_t^{-1}(\hat{P})$, where $\hat{P}$ is collection of k nearest $\hat{p}$: $\hat{P} = [\hat{p}_1 \ \hat{p}_2 \ \dots \ \hat{p}_k]$.

$$\mathcal{F}_t^{-1}(\hat{P}) = [\mathcal{F}_t^{-1}(\hat{p}_1) \ \dots \ \mathcal{F}_t^{-1}(\hat{p}_k)]_{d \times k} = M^T [\hat{p}_1 \ \dots \ \hat{p}_k]_{d \times k}$$

5.2.3 Reference Location Retrieval. After k NN search, k nearest data records $\mathcal{F}_t^{-1}(\hat{P})$ are retrieved, where reference locations are embedded and encrypted. To prevent sending the whole encrypted data record (ciphertext) to users, our model forces EIPS to apply another transformation on the ciphertext to reduce its dimension. The transformation is performed using **Downsize Key (DK)**, whose functionality is downsizing and transforming the ciphertext, so that neither users nor EDS has access to the actual data records stored in EIPS. Such transformation is necessary since EDS is able to decrypt the user's location by comparing against the database if data records (ciphertext) are passed to EDS as their original format. EIPS applies DK to $\mathcal{F}_t^{-1}(\hat{P})$ and return resultant ciphertext to users. EIPS is forced to perform DK transformation since the users can audit its action by checking the dimensions, which guarantees that EDS does not received a full-length record.

Definition (Downsize Key) During the Initialization Phase, EDS first generates a matrix M_r , which will be embedded in query q_{RL} .

$$M_r = \begin{bmatrix} r_{11} & \dots & r_{1I} \\ \vdots & \ddots & \vdots \\ r_{I1} & \dots & r_{II} \end{bmatrix}_{I \times I}$$

EDS then embed M_r into query q_{RL}

$$\begin{aligned}
q_{RL} &= \begin{bmatrix} 0 & \dots & r_{11} & r_{1(i+1)} & \dots & r_{1I} & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & r_{I1} & r_{I(i+1)} & \dots & r_{II} & \dots & 0 \end{bmatrix}_{I \times d} \\
\hat{p} &= [\lambda_1^p \ \dots \ l_x \ \dots \ l_y \ \dots \ r_t^p \ \dots \ r_d^p]^T
\end{aligned}$$

q_{RL} is a padded matrix of M_r with dimension $I \times d$ ($I \geq 2$), and all the padding is 0. EDS will choose which dimensions to pad. Since EDS has access to the dimensions of l_x, l_y in $\hat{p}$ ($n+2, n+3$ in our running example), the corresponding dimension of l_x, l_y in q_{RL} can not be 0. The reason is that q_{RL} is designed to retrieve l_x, l_y , if the corresponding dimensions of l_x, l_y are 0, we will lose the essential information. We formally define DK as

$$DK = (M^{-1} \cdot q_{RL}^T)^T$$

Since EIPS has no access to the dimension of l_x and l_y , q_{RL} is secure under the worst assumption that M_r is decrypted. There are A_d^I ways to embed M_r , the probability of retrieving q_{RL} is 0.01% with $d = 120, I = 2$. After the encryption, $DK = q_{RL} \cdot M^{-1}$. The benefit of DK is three-fold: (1) the dimension of ciphertext is $I \times k$ with each

data record downsized from dimension of d to I while preserving reference location (l_x, l_y) related information. (2) downsizing the dimensions of ciphertext prevents EDS from deducting the data records by comparing against the radio map. A user inserts the artificial columns to the ciphertext, which further increases the difficulties for EDS to deduct the original data records. (3) q_{RL} is encrypted with M^{-1} without revealing the dimensions of reference locations. EDS needs to generate DK and send it to EIPS before service.

After EIPS locates k nearest records, the reference location l_i^x, l_i^y are retrieved via the following operation:

$$DK \cdot \mathcal{F}_t^{-1}(\hat{P}) = (M^{-1} \cdot q_{RL}^T)^T \cdot M^T \cdot \hat{P} = M_r \cdot RL^{+D} = \mathcal{R}_t(RL^{+D})$$

where RL^{+D} represents partial records of $\hat{P}$ (corresponding to non-zero dimensions in q_{RL}). The dimensions in **gray** are the artificial dimensions that returned to users, which are either Type I or Type II artificial dimensions^{+D}. The dimensions of artificial dimensions are arbitrary as long as they are pre-defined by EDS. We only show Type II appendix padding in our example due to limited space.

$$RL^{+D} = [RL_1^{+D} \ RL_2^{+D} \ \dots \ RL_k^{+D}] = \begin{bmatrix} l_1^x & \dots & l_k^x \\ l_1^y & \dots & l_k^y \\ \vdots & \ddots & \vdots \\ r_t^{p_1} & \dots & r_t^{p_k} \end{bmatrix}_{I \times k}$$

Example We provide a running example with $k = 2, I = 3, d = 6$ (reference locations are highlighted)

$$\begin{aligned}
\hat{p} &= \begin{bmatrix} 2 & 6 & 20 & \textcolor{blue}{2} & \textcolor{blue}{3} & 9 \\ 4 & 2 & 10 & \textcolor{blue}{2} & \textcolor{blue}{4} & 3 \end{bmatrix}^T \\
M_r &= \begin{bmatrix} 2 & 3 & 1 \\ 1 & 2 & 8 \\ 3 & 5 & 3 \end{bmatrix}_{3 \times 3} \\
q_{RL} &= \begin{bmatrix} 0 & 0 & 0 & 2 & 3 & 1 \\ 0 & 0 & 0 & 1 & 2 & 8 \\ 0 & 0 & 0 & 3 & 5 & 3 \end{bmatrix}_{3 \times 6} \\
DK \cdot \mathcal{F}_t^{-1}(\hat{P}) &= \mathcal{R}_t(RL^{+D}) = \begin{bmatrix} 22 & 19 \\ 80 & 34 \\ 48 & 35 \end{bmatrix} \\
RL^{+D} &= \begin{bmatrix} \textcolor{blue}{2} & \textcolor{blue}{3} & 9 \\ \textcolor{blue}{2} & \textcolor{blue}{4} & 3 \end{bmatrix}^T
\end{aligned}$$

5.3 Outbound Scheme

In this section, we introduce our outbound encryption scheme denoted as $\mathcal{R}$. The functionality of $\mathcal{R}$ is to retrieve the k nearest fingerprint neighbors' corresponding reference locations. We define $\mathcal{R}_u$ as $\mathcal{R}$ conducted by users. In the outbound scheme, users adopt commutative cipher $\mathcal{R}_u$ to protect their privacy from EDS, and EDS uses $\mathcal{R}_t$ to prevent EIPS from acquiring the users' locations. Both $\mathcal{R}_u$ and $\mathcal{R}_t$ could introduce KPA, however, the key of $\mathcal{R}_u$ is dynamic, and we take advantage of the anonymity of the dimensions in a query as well as in data record (of the radio map) to prevent KPA towards the key of $\mathcal{R}_t$. First of all, we introduce the definitions:

$$\begin{aligned}
\mathcal{R}_t(x) &= \mathcal{R}_t(x, M_r) = M_r \cdot x & \mathcal{R}_t^{-1}(x) &= \mathcal{R}_t^{-1}(x, M_r) = M_r^{-1} \cdot x \\
\mathcal{R}_u(x) &= \mathcal{R}_u(x, A_k) = x \cdot A_k & \mathcal{R}_u^{-1}(x) &= \mathcal{R}_u^{-1}(x, A_k^{-1}) = x \cdot A_k^{-1}
\end{aligned}$$

5.3.1 User Encryption. In order to protect users' privacy, a user performs the following steps

- Encrypt ciphertext from EIPS with user's key $\mathcal{R}_u$.
- Inserting Artificial Columns.

The general idea is to prevent EDS from comparing the ciphertext with the encrypted database.

a. Encryption Function A user encryption function is defined as

$$\begin{aligned} & \mathcal{R}_u \cdot \mathcal{R}_t(RL^{+D}) \\ &= \mathcal{R}_t(RL^{+D}) \cdot A_k \\ &= M_r RL^{+D} A_k \\ &= [M_r \cdot \alpha_1 \cdot RL_1^{+D} \quad \dots \quad M_r \cdot \alpha_k \cdot RL_k^{+D}]_{I \times k} \end{aligned}$$

where a user key is defined as $A_k = \begin{bmatrix} \alpha_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \alpha_k \end{bmatrix}_{k \times k}$

b. Insert(+) Artificial Column Users insert m artificial columns, with a slight abuse of denotation, denoted as $M_r C_i^a$ $i \in [1..m]$. The content of each artificial column is random values generated by users on-the-fly.

$$\begin{aligned} & \mathcal{R}_u \cdot \mathcal{R}_t(RL^{+D})^C = \\ & [M_r \cdot \alpha_1 \cdot RL_1^{+D} \quad \dots \quad M_r C_1^a \quad \dots \quad M_r C_m^a \quad \dots \quad M_r \cdot \alpha_k \cdot RL_k^{+D}] \end{aligned}$$

Level 1 Attack by EDS Before adding artificial columns, user encryption A_k is a simple scaling of each ciphertext. EDS can infer the k target records by comparing with all the database records. However, inserting artificial columns in the ciphertext at random columns, without affecting the correctness of the encryption scheme, prevents EDS from comparing with the database due to the randomness introduced.

5.3.2 EDS Decryption. EDS decryption is a matrix transformation $\mathcal{R}_t^{-1}$ followed by removing artificial dimensions. We define

$$C_i^a = \begin{bmatrix} c_{1i}^a \\ \vdots \\ c_{ti}^a \\ c_{li}^a \end{bmatrix}$$

$$\begin{aligned} & \mathcal{R}_t^{-1} \cdot \mathcal{R}_u \cdot \mathcal{R}_t(RL^{+D})^C \\ &= M_r^{-1} \mathcal{R}_u \cdot \mathcal{R}_t(RL^{+D})^C \\ &= [\alpha_1 \cdot RL_1^{+D} \quad \dots \quad M_r^{-1} M_r C_1^a \quad \dots \quad M_r^{-1} M_r C_m^a \quad \dots \quad \alpha_k \cdot RL_k^{+D}] \\ &= \begin{bmatrix} \alpha_1 l_1^x & \dots & c_{11}^a & \dots & c_{1m}^a & \dots & \alpha_k l_k^x \\ \alpha_1 l_1^y & \dots & c_{21}^a & \dots & c_{2m}^a & \dots & \alpha_k l_k^y \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \alpha_1 l_1^{p_1} & \dots & c_{t1}^a & \dots & c_{tm}^a & \dots & \alpha_k l_k^{p_k} \end{bmatrix}_{I \times (k+m)} \end{aligned}$$

However, since EDS knows the dimensions where l_x l_y are located, EDS removes artificial dimensions (- **Artificial Dimensions**) and returns

$$\mathcal{R}_u(RL)^C = \begin{bmatrix} \alpha_1 l_1^x & \dots & c_{11}^a & \dots & c_{1m}^a & \dots & \alpha_k l_k^x \\ \alpha_1 l_1^y & \dots & c_{21}^a & \dots & c_{2m}^a & \dots & \alpha_k l_k^y \end{bmatrix}_{2 \times k}$$

Since artificial columns are randomly generated, $c_{11}^a \dots c_{1m}^a$ and $c_{21}^a \dots c_{2m}^a$ are random values as well. Due to the anonymity of the columns, only users have access to the actual columns of reference locations.

5.3.3 User Decryption. A user has inserted artificial columns in the ciphertext during the encryption process, which are removed here in the decryption process. After removing artificial columns (- **Artificial Columns**), a user performs

$$\begin{aligned} RL &= \mathcal{R}_u^{-1} \cdot \mathcal{R}_u(RL) = \begin{bmatrix} \alpha_1 l_1^x & \dots & \alpha_k l_k^x \\ \alpha_1 l_1^y & \dots & \alpha_k l_k^y \end{bmatrix}_{2 \times k} \begin{bmatrix} \frac{1}{\alpha_1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \frac{1}{\alpha_k} \end{bmatrix}_{k \times k} \\ \begin{bmatrix} l_x \\ l_y \end{bmatrix} &= \frac{\sum RL}{k} = \begin{bmatrix} \sum_{i=1}^k l_i^x / k \\ \sum_{i=1}^k l_i^y / k \end{bmatrix} \end{aligned}$$

Example We continue our running example after a user receives the following cipher text

$$\begin{aligned} & \mathcal{R}_t(RL^{+D}) = \begin{bmatrix} 22 & 19 \\ 80 & 34 \\ 48 & 35 \end{bmatrix} \\ & \text{----- User Encryption -----} \\ & \text{key } A_k = \begin{bmatrix} 3 & 0 \\ 0 & 0.5 \end{bmatrix}_{2 \times 2} \\ & \mathcal{R}_u \cdot \mathcal{R}_t(RL^{+D}) = \mathcal{R}_t(RL^{+D}) \cdot A_k = \begin{bmatrix} 66 & 9.5 \\ 240 & 17 \\ 144 & 17.5 \end{bmatrix} \\ & \quad \quad \quad + \text{Artificial Columns} \\ & \mathcal{R}_u \cdot \mathcal{R}_t(RL^{+D})^C = \begin{bmatrix} 66 & 30 & 9.5 \\ 240 & 60 & 17 \\ 144 & 57 & 17.5 \end{bmatrix} \\ & \text{----- EDS Decryption -----} \\ & \mathcal{R}_t^{-1} \cdot \mathcal{R}_u \cdot \mathcal{R}_t(RL^{+D})^C = M_r^{-1} \cdot \mathcal{R}_u \cdot \mathcal{R}_t(RL^{+D})^C \\ & \quad \quad \quad = \begin{bmatrix} 6 & 1 & 1 \\ 9 & 7.5 & 2 \\ 27 & 5.5 & 1.5 \end{bmatrix} \\ & \quad \quad \quad - \text{Artificial Dimensions} \\ & \mathcal{R}_u(RL)^C = \begin{bmatrix} 6 & 1 & 1 \\ 9 & 7.5 & 2 \end{bmatrix} \\ & \text{----- User Decryption -----} \\ & \quad \quad \quad - \text{Artificial Columns} \\ & \mathcal{R}_u(RL) = \begin{bmatrix} 6 & 1 \\ 9 & 2 \end{bmatrix} \quad RL = \mathcal{R}_u(RL) \cdot A_k^{-1} = \begin{bmatrix} 2 & 2 \\ 3 & 4 \end{bmatrix} \end{aligned}$$

KPA resistance There is minimum KPA threat towards $\mathcal{R}_u$ due to artificial columns and its dynamic nature (could be changed per query). A user could potentially decrypt $\mathcal{R}_t$ by KPA; however, instead of full plaintext, only partial plaintext (artificial dimensions removed) is returned to users.

6 EMPIRICAL EVALUATION

6.1 Experiments Setting

We compare our EIPS method with the state-of-the-art solution TVM [14]. Since in our EIPS model, a continuous query merely is a sequence of consecutive snapshot queries without further loss of privacy or efficiency, we only compare snapshot queries in this experiment. In our EIPS setting, each query is independent, and the attacker is not able to infer the closeness of two consecutive

queries. We conduct real-world experiments with Android mobile applications on the Samsung Galaxy S4¹.

Unlike *TVM* [14] uses Hadoop/Hbase, our experimental centralized server runs on a single-node (2.7 GHz Intel Core i5 with 16 GB memory). We change the setting of *TVM* to a **single server** version. As EIPS is efficient enough that a single-node setting is adequate for normal users, we did not adopt acceleration schemes such as Hadoop and HBase. Both *TVM* and EIPS are more efficient with accelerating schemes. In our experimental settings, we focus on *comparing the pros and cons* of each method rather than merely comparing the execution time.

6.2 Localization Accuracy of EIPS

k	Number of RM records (10^3)				
	0.5	1	1.5	2	2.5
3	-8e-12	-8e-12	-8e-12	-8e-12	-8e-12
5	-7e-12	-7e-12	-7e-12	-7e-12	-7e-12
10	-6.8e-12	-6.8e-12	-6.8e-12	-6.8e-12	-6.8e-12
15	-7.1e-12	-7.1e-12	-7.1e-12	-7.1e-12	-7.1e-12
20	-6.7e-12	-6.7e-12	-6.7e-12	-6.7e-12	-6.7e-12

Table 3: Accuracy errors between *TVM* and EIPS in meters.

In this section, we verify the localization accuracy of our EIPS system. The accuracy of WiFi positioning has been extensively discussed in [15]. The only difference between EIPS positioning and conventional WiFi positioning in [15] is encryption. In this section, we use *TVM* positioning as ground truth and explore the accuracy cost of the encryption.

We test the average localization error between *TVM* and EIPS by varying the size of the radio map and k . As we know, both *TVM* and EIPS use k NN to estimate the user's location, which means the inherent error should be the same. Since EIPS is conducted on an encrypted database, we want to make sure that the encryption does not introduce more errors. As we vary the size of the radio map and k respectively, the accuracy error is within the order of 10^{-12} meters. We omit these errors in real-world scenarios.

6.3 Time Efficiency of EIPS

(a) Total execution time. (b) Correspondence |pRM|.

Figure 4: Execution time varying RM size.

6.3.1 Time Efficiency with Varying Radio Map Sizes. We explore the time efficiency by varying the size of the radio map. As the whole localization process is operated on the cloud server efficiently, the primary time cost for EIPS is the I/O cost. While in *TVM*, the most time-consuming operation is k NN search.

¹Our code is publicly available at <https://github.com/VV123/EIPS>

Figure 4a shows the total response time by varying the radio map size. As we know, *TVM* retrieves *partial radio map* (pRM) to the mobile phone. As the size of radio map changes, the size of pRM (|pRM|) changes correspondingly. Some queries are not affected by the change of radio map size, so we use different notations for those two types of queries:

- *TVM*_{incr} represents the queries with increasing |pRM|.
- *TVM*_{const} represents the queries with constant |pRM|.

Figure 4b shows the size of |pRM| in correspondence to each type. The difference of *TVM*_{incr} and *TVM*_{const} shows that the time efficiency of *TVM* relies on |pRM|, not the size of radio map.

As we can see from Figure 4a, for EIPS, the execution time is constant. The radio map size does not affect the execution time of EIPS. To further prove our theory, we test the pure NN execution time of EIPS in Figure 5b. Nearest neighbor query execution time is constant as well as the total response time because EIPS conducts a linear scan on the whole radio map on the server-side. The server is much more powerful than mobile devices. Consequently, $O(n)$ execution does not cause a recognizable increment with the increasing of n in our experimental settings.

For *TVM* method, we compare the |pRM| in Figure 4b. As EIPS does not retrieve partial radio map, EIPS is not included in the figure. The execution time of *TVM* is correlated with |pRM|. On average, larger radio map size will cause larger |pRM|. We also test the NN execution time of *TVM* in Figure 5b. Unlike EIPS, *TVM* conducts nearest neighbor queries on the user side. As we can see from Figure 5b, *TVM* NN execution time is correlated with |pRM|. It means that even for $O(n)$ computation, the increment of n will cause substantial increment on execution time in our experimental settings. *TVM* efficiency relies highly on NN execution time in general. As a result, the execution time of *TVM* will increase with larger radio map on average.

On average, with the single-node server setting, the execution time of *TVM* does not perform better than EIPS. We can also see that *TVM* is drastically affected by the size of the radio map. On the contrary, EIPS is scalable varying the size of the radio map or k .

6.3.2 Time Efficiency with Varying k . We explore the scalability of *TVM* and EIPS by varying k . In Figure 5a, *TVM*-total and *EIPS*-total represent the total execution time by varying k from 3 to 20. *TVM*-NN and *EIPS*-NN represent the pure nearest neighbor execution time.

The total execution time of *TVM* is composed of data transfer and the nearest neighbor execution. We fix the size of the radio map, so that data transfer time is fixed as well. We only focus on the

(a) Varying k .

(b) Varying the size of radio map.

Figure 5: kNN execution time.

(a) Energy with varying dataset. (b) Energy with varying k .

Figure 6: Energy consumption.

trend of nearest neighbor execution. As we can see from Figure 4a, both *TVM* and *EIPS* are constant in terms of total execution time. The trends of NN execution time are slightly different between *TVM* and *EIPS*. *TVM*-NN does not increase with increasing k because the user retrieves partial radio map (pRM) from the central server, and NN search is executed on pRM, whose size is relatively small.

We do not observe a noticeable increasing trend on *EIPS*-NN. Even though the nearest neighbor search is among the whole radio map instead of part of it, the computation is at the server-side, which is more efficient than on the user side.

6.4 Energy Consumption Evaluation

We follow the evaluation metrics used in the previous study [14]. We compare the performance of *TVM* and *EIPS* in terms of energy (in Joules) consumed by smartphones. Similar to [14], we ignore server-perceived energy costs. The energy consumed by a smartphone is measured with the help of PowerTutor². We measure the energy consumed by the CPU.

As we can see from Figure 6, when increasing data size, the energy consumption by *EIPS* does not show tremendous change. The energy consumption of *TVM* slightly increases with larger data size, which is consistent with the results in [14]. With larger k , both *EIPS* and *TVM* consume more energy. However, in all the cases mentioned above, *TVM* consumes more energy than *EIPS* on the user side. This observation is consistent with our theory in Section 2.2, that *EIPS* transfers the workload to the centralized server, while *TVM* maintains most of the workload on the user side.

7 CONCLUSION

In this paper, we propose an encryption-based privacy-preserving indoor positioning system, called Encryption-based IPS (*EIPS*). The proposed *EIPS* enables a user to enjoy efficient and accurate localization without leaking his/her location to the IPS server. Our scheme works for both snapshot and continuous queries without loss of efficiency. We present a detailed design of the architecture and components of *EIPS*. We derive theoretical guarantees for the design and give analyses on the communication cost and overall security of *EIPS*. We conduct an experimental evaluation using real-world datasets, and the results show that *EIPS* is able to provide accurate and efficient services while maintaining privacy guarantees.

Acknowledgement This research has been funded in part by the U.S. National Science Foundation grants IIS-1618669 (III) and ACI-1642133 (CICI). Hua Lu's work was partly supported by Independent Research Fund Denmark under grant no. 8022-00366B.

²<http://ziyang.eecs.umich.edu/projects/powertutor>

REFERENCES

- [1] O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Proceedings of the 24th ICDE*, pages 376–385, 2008.
- [2] O. Abul, F. Bonchi, and M. Nanni. Anonymization of moving objects databases by clustering and perturbation. *Inf. Syst.*, 35(8):884–910, 2010.
- [3] C. Chow, M. F. Mokbel, and X. Liu. A peer-to-peer spatial cloaking algorithm for anonymous location-based service. In *14th ACM International Symposium on Geographic Information Systems, ACM-GIS, Proceedings*, pages 171–178, 2006.
- [4] C. Gentry and S. Halevi. Implementing gentry's fully-homomorphic encryption scheme. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Proceedings*, pages 129–148, 2011.
- [5] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K. Tan. Private queries in location based services: anonymizers are not necessary. In *Proceedings of the ACM SIGMOD*, pages 121–132, 2008.
- [6] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the First International Conference on Mobile Systems, Applications, and Services, MobiSys*, pages 31–42, 2003.
- [7] Y. Gu, A. C. C. Lo, and I. G. Niemegeers. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys and Tutorials*, 11(1):13–32, 2009.
- [8] S. U. Hussain and F. Koushanfar. Privacy preserving localization for smart automotive systems. In *Proceedings of the 53rd Annual Design Automation Conference, DAC*, pages 26:1–26:6, 2016.
- [9] P. Indyk and D. P. Woodruff. Polylogarithmic private approximations and efficient matching. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC, Proceedings*, pages 245–264, 2006.
- [10] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing location-based identity inference in anonymous spatial queries. *IEEE Trans. Knowl. Data Eng.*, 19(12):1719–1733, 2007.
- [11] A. Khoshgozaran and C. Shahabi. Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In *10th International SSTD, Proceedings*, pages 239–257, 2007.
- [12] H. Kido, Y. Yanagisawa, and T. Satoh. An anonymous communication technique using dummies for location-based services. In *Proceedings of ICPS*, pages 88–97, 2005.
- [13] J. Kim, Y. Han, and K. Li. K-anonymity in indoor spaces through hierarchical graphs. In *Proceedings of the Fourth ACM SIGSPATIAL Workshop on ISA*, pages 21–28, 2012.
- [14] A. Konstantinidis, G. Chatzimiloudis, D. Zeinalipour-Yazti, P. Mpei, N. Pelekis, and Y. Theodoridis. Privacy-preserving indoor localization on smartphones. *IEEE Trans. Knowl. Data Eng.*, 27(11):3042–3055, 2015.
- [15] B. Li, J. Salter, A. G. Dempster, and C. Rizos. Indoor positioning techniques based on wireless lan. In *LAN, first IEEE international conference on wireless broadband and ultra wideband communications*, 2006.
- [16] M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The new casper: Query processing for location services without compromising privacy. In *Proceedings of the 32nd international conference on Very large data bases*, pages 763–774. VLDB Endowment, 2006.
- [17] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. Knowl. Data Eng.*, 13(6):1010–1027, 2001.
- [18] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.
- [19] W. Wang, Y. Hu, L. Chen, X. Huang, and B. Sunar. Exploring the feasibility of fully homomorphic encryption. *IEEE Trans. Computers*, 64(3):698–706, 2015.
- [20] W. Wang and W.-S. Ku. Dynamic indoor navigation with bayesian filters. *SIGSPATIAL Special*, 8(3):9–10, 2017.
- [21] W. Wang and W.-S. Ku. Recommendation-based smart indoor navigation. In *Proceedings of the Second International Conference on Internet-of-Things Design and Implementation*, pages 311–312. ACM, 2017.
- [22] W. Wang, J. Zhang, M.-T. Sun, and W.-S. Ku. Efficient parallel spatial skyline evaluation using mapreduce. In *Proceedings of the 20th international conference on extending database technology*, 2017.
- [23] W. Wang, J. Zhang, M.-T. Sun, and W.-S. Ku. A scalable spatial skyline evaluation system utilizing parallel independent region groups. *The VLDB Journal The International Journal on Very Large Data Bases*, 28(1):73–98, 2019.
- [24] W. K. Wong, D. W.-I. Cheung, B. Kao, and N. Mamoulis. Secure knn computation on encrypted databases. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 139–152. ACM, 2009.
- [25] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu. Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In *In proceedings of 24th IEEE ICDE*, pages 366–375, 2008.