PMRQ: Achieving Efficient and Privacy-Preserving Multi-Dimensional Range Query in eHealthcare

Yandong Zheng, Rongxing Lu, *Fellow, IEEE,* Songnian Zhang, Yunguo Guan, Jun Shao, *Senior Member, IEEE,* Fengwei Wang, *Member, IEEE,* and Hui Zhu, *Senior Member, IEEE*

***Abstract*—Healthcare data explosion and cloud computing booming have motivated healthcare centers to outsource their healthcare data and data-driven services to a powerful cloud. Nevertheless, due to privacy concerns, the data are usually encrypted before being outsourced, which will degrade the data utility and make it challenging to implement data-driven services. Although the multi-dimensional range query over encrypted data, as one of the most popular outsourced services in eHealthcare, has been extensively studied, existing solutions still have some limitations in efficiency, privacy, and practicality. Aiming at this challenge, in this paper, we design an efficient and privacy-preserving multi-dimensional range query (PMRQ) scheme. We first build an R-tree to index the dataset and reduce the R-tree-based range queries to the multi-dimensional range intersection problem. Then, by delicately designing a data comparison algorithm and a homomorphic encoding technique, we present an encoding-based range intersection algorithm. After that, by employing matrix encryption to protect the privacy of the encoding-based range intersection algorithm, we design a multi-dimensional range intersection predicate encryption (MRIPE) scheme. Based on the MRIPE scheme, we then propose our PMRQ scheme. Detailed security analysis illustrates that our PMRQ scheme is privacy-preserving, and experimental results demonstrate that it is computationally efficient.**

***Index Terms*—Multi-dimensional range query, R-tree, single-dimensional privacy, eHealthcare, homomorphic encoding.**

I. Introduction

The aging population, digitization of eHealthcare systems, evolution of wireless networks, and advance of machine learning have jointly stimulated the exponential growth of the medical data at the healthcare centers [1]–[3]. These accumulated medical data have been widely utilized to offer various query services, e.g., range queries, similarity queries [4], and skyline queries [5], to doctors. Among them, the multi-dimensional range query, which retrieves data records within a query range, is highly regarded due to its fast-growing applications in disease diagnosis and healthcare monitoring [6]. For better understanding, we show an example to illustrate multi-dimensional range queries.

Example 1: Suppose that a medical dataset has four records with two attributes, i.e., (age and blood glucose), denoted by $\mathbb{D} = \{\mathbf{x}_1 = (38, 138), \mathbf{x}_2 = (34, 180), \mathbf{x}_3 = (25, 146), \mathbf{x}_4 = (50, 50)\}$. Let $\mathcal{Q} = [20, 35] \times [120, 150]$ be a range query, which aims to retrieve data records, whose age is in $[20, 35]$ and blood glucose is in $[120, 150]$. The query result of $\mathcal{Q}$ over $\mathbb{D}$ will be $\mathbf{x}_3 = (25, 146)$.

As the medical data volumes grow, healthcare centers progressively choose to outsource their medical data and the multi-dimensional range query service to a powerful cloud. Taking the privacy into account, healthcare centers usually encrypt the medical data and outsource the corresponding ciphertexts to the cloud. However, data encryption will degrade the data utility and make it challenging to perform multi-dimensional range queries. Aiming at this challenge, various schemes [7]–[15] have been reported, but they still have some limitations in efficiency, privacy, and practicality:

- *Search efficiency:* Schemes [7]–[9] are inefficient, because (i) all of them are designed using the computationally expensive cryptographic primitives; and (ii) the search efficiency of schemes in [7], [8] is linear to the size of the dataset.
- *Privacy:* Schemes in [7], [9] cannot preserve the query privacy (i.e., the plaintext of query requests). Schemes in [10]–[12] leak the single-dimensional privacy, which refers to the information on which records satisfy the query request in each dimension. For example, the single-dimensional privacy of $\mathcal{Q} = [20, 35] \times [120, 150]$ in Example 1 refers to the private information that $\{\mathbf{x}_2 = (34, 180), \mathbf{x}_3 = (25, 146)\}$ satisfy $\mathcal{Q}$ in the age dimension. As discussed in [16], the leakage of the single-dimensional privacy may have a disastrous consequence on the privacy of query requests and dataset, e.g., leaking the plaintext information of the entire dataset. As a result, it is critical to preserve the single-dimensional privacy in multi-dimensional range queries.
- *Practicality:* The scheme in [13] is designed in a two-server model and secure under the non-colluding assumption between two servers. However, this assumption is too strict in some practical scenarios. Schemes in [14], [15] are designed based on the bucketization method, which are impractical because their query results may contain false positive records.

To address the above challenges, in this paper, we present an efficient, privacy-preserving, and practical multi-dimensional range query (PMRQ) scheme under a single-server setting. In our scheme, since R-tree is widely used to index the dataset and support multi-dimensional range queries in the database systems [13], we will first build an R-tree to index the dataset. Then, we reduce the R-tree-based range queries to the multi-dimensional range intersection problem, which determines whether two multi-dimensional ranges $\mathcal{P}, \mathcal{Q}$ intersect or not,

Y. Zheng, R. Lu, S. Zhang, and Y. Guan are with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: yzheng8@unb.ca, rlu1@unb.ca, szhang17@unb.ca, yguan4@unb.ca).

J. Shao is with School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, 310018, China (e-mail: chn.junshao@gmail.com).

F. Wang and H. Zhu are with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, 710071, China (e-mail: wangfengwei@xidian.edu.cn, zhuhui@xidian.edu.cn).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3158321, IEEE Internet of Things Journal

i.e., $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. After that, we design a multi-dimensional range intersection predicate encryption (MRIPE) scheme to privately check $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. In the following, we discuss two challenges, which we will face when designing the MRIPE scheme, together with our countermeasures.

Challenge I: How to efficiently conduct multiple inequalities as a whole? Let $\mathcal{P} = P_1 \times P_2 \times \cdots \times P_d$ and $\mathcal{Q} = Q_1 \times Q_2 \times \cdots \times Q_d$ be two d-dimensional ranges, where $P_k = [p_{k,l}, p_{k,r}]$ and $Q_k = [q_{k,l}, q_{k,r}]$ for $1 \leq k \leq d$. We have

$$\mathcal{P} \cap \mathcal{Q} \neq \emptyset \Leftrightarrow \{p_{k,r} \geq q_{k,l} \text{ and } p_{k,l} \leq q_{k,r}\}_{k=1}^{d}. \quad (1)$$

Then, determining $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ is equivalent to determining whether $\mathcal{P}$ and $\mathcal{Q}$ satisfy $2d$ inequalities $\{p_{k,r} \geq q_{k,l}$ and $p_{k,l} \leq q_{k,r}\}_{k=1}^{d}$. A straightforward method to perform $2d$ inequalities is to separately determine each inequality, but this method unavoidably causes the single-dimensional privacy leakage. To avoid this leakage, $2d$ inequalities are required to be determined as a whole. That is, an adversary (i.e., the cloud server in our scheme) is only allowed to know whether $\mathcal{P}$ and $\mathcal{Q}$ satisfy the $2d$ inequalities in Eq. (1) or not. When $\mathcal{P}$ and $\mathcal{Q}$ are not satisfied, the cloud server cannot know which inequalities do not hold. However, to the best of our knowledge, there is no efficient solution to conduct multiple inequalities with the single-dimensional privacy.

Countermeasure I: To solve ***Challenge I***, we design a data comparison algorithm to compare two integers p and q. The main idea is to represent p and q to vectors $\{\widehat{\mathbf{p}}_i, \overline{\mathbf{p}}_i\}_{i=1}^{n}$ and $\{\mathbf{q}_i\}_{i=1}^{n}$, where n is the bit length of values in $\mathcal{P}, \mathcal{Q}$. More details about the vector representation of p and q are provided in Section IV-A. Then, comparing p and q can be transformed to an equality test, i.e.,

$$\text{(i) } p < q \Leftrightarrow f(p,q) = 1; \text{ (ii) } p \geq q \Leftrightarrow f(p,q) = 0,$$

where $f(p,q) = \sum_{i=1}^{n}((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T)$. The equality test based data comparison algorithm enables us to integrate the $2d$ data comparisons in the computation of $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ to one equality test such that we can design an efficient multi-dimensional range intersection algorithm. After that, we should consider how to privately compute $f(p,q)$. Then, we will encounter ***Challenge II*** as follows.

Challenge II: How to privately compute the multiplication of multiple vectors? When computing $f(p,q) = \sum_{i=1}^{n}((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T)$, the most challenging work is to privately compute $f_i(p,q) = (\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T$, i.e., computing the multiplication of multiple vectors. Although fully homomorphic encryption techniques [17], [18] can be naturally used to implement this computation, the computational costs are prohibitively large. A promising solution is to use the extended Hill cipher encryption (also called matrix encryption) in [19], [20] to protect the privacy of the computation $f_i(p,q)$. However, since $\{\widehat{\mathbf{p}}_j, \overline{\mathbf{p}}_j, \mathbf{q}_j\}$ are binary vectors and even some of them are zero vectors, using the matrix encryption to encrypt them will suffer from the matrix rank attack. That is, the attacker can use the rank of ciphertexts to infer the underlying plaintext data. More details on the extended Hill cipher encryption and the matrix rank attack are provided in Section IV-B.

Countermeasure II: To address ***Challenge II***, we design a homomorphic encoding technique to encode binary vectors $\{\overline{\mathbf{p}}_j, \widehat{\mathbf{p}}_j, \mathbf{q}_j\}$ into non-zero random vectors. Since the encoded vectors do not have 0, the rank of ciphertexts will not leak the plaintext data. Hence, our scheme is free of the matrix rank attack. Besides, the homomorphic property of the encoding technique can ensure that $f(p,q)$ can be correctly computed.

In summary, our contributions are three folds as follows.

- First, we design a data comparison algorithm to compare two integers p and q. Then, we present a homomorphic encoding technique to encode data. Based on them, we construct an encoding-based multi-dimensional range intersection algorithm, which can efficiently determine whether $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$.
- Second, by employing matrix encryption to protect the privacy of the encoding-based range intersection algorithm, we design a multi-dimensional range intersection predicate encryption (MRIPE) scheme to privately determine $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. Based on MRIPE scheme, we propose an efficient and privacy-preserving multi-dimensional range query (PMRQ) scheme.
- Finally, we prove that our PMRQ scheme is able to preserve the data privacy, query privacy, and single-dimensional privacy simultaneously. In addition, the extensive experiments demonstrate that our scheme is computationally efficient.

The remainder of our paper is organized as follows. We formalize our system and security models in Section II and recall the preliminaries in Section III. We introduce some building block techniques of our scheme in Section IV and propose our PMRQ scheme in Section V. In Section VI, we analyze the security of our scheme, followed by its performance evaluation in Section VII. We review some related works in Section VIII and conclude our work in Section IX.

II. System Model and Security Model

In this section, we define the system and security models considered in our work.

A. System Model

In the system model, we consider a single-server-based multi-dimensional range query model in eHealthcare, which involves a healthcare center, a cloud server, and multiple query doctors, as shown in Fig. 1.

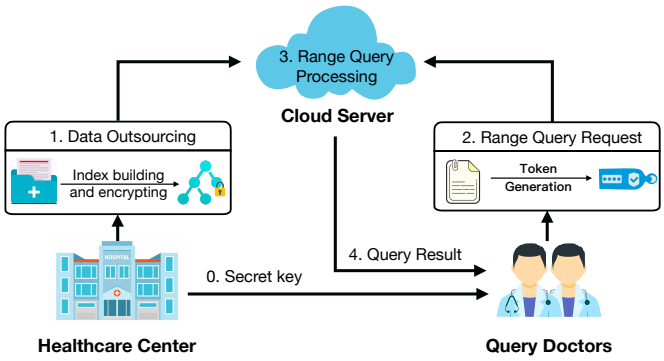


Fig. 1. System model under consideration

- Healthcare Center (HC): The HC has a multi-dimensional medical dataset and leverages it to offer the multi-dimensional

range query service to query doctors. Constrained by the computing capability, it outsources the dataset and the multi-dimensional range query service to a powerful cloud. To speed up the query efficiency and protect the data privacy, the HC indexes the dataset with an R-tree and outsources the R-tree to the cloud in an encrypted form.

• Cloud Server: The cloud server possesses abundant computing and storage resources. It stores the encrypted R-tree that is outsourced by the HC, and offers the multi-dimensional range query service to query doctors. Concretely, on receiving a range query request from a query doctor, the cloud server will search on the encrypted R-tree for records satisfying the query requests and respond them to the requesting doctor.

• Query Doctors: In the system, there are many query doctors, and each one is authorized by the HC with an authorized key. Authorized doctors can enjoy the range query service from the cloud server. To protect the privacy of query requests, doctors are required to send encrypted query requests to the cloud server.

B. Security Model

Regarding the security model, we assume that the HC is *trusted* because it sets up the system and has no incentive to deviate from the range query service. As for the cloud server, we assume that it is *semi-honest*, namely, it sincerely follows our scheme to offer the multi-dimensional range query service to doctors but may attempt to deduce (i) the plaintext of dataset records and query requests; and (ii) the single-dimensional privacy of range queries. For the query doctors, since they have been authorized, we assume that they are *honest*, namely, they will faithfully encrypt multi-dimensional range queries into query tokens and send the query tokens to the cloud as the query requests. In addition, we assume that the cloud server does not collude with any query doctor due to conflicts of interest. It is worth noting that other active attacks, e.g., impersonation, may be launched by adversaries. Since this work focuses on privacy, those attacks are beyond the scope of this paper, and will be discussed in our future work.

III. Preliminaries

In this section, we recall R-tree data structure together with the R-tree based multi-dimensional range query algorithm.

R-tree. R-tree is a classical tree index and can be used to represent multi-dimensional dataset [21]. It is usually built by recursively grouping nearby data records and using a minimum bounding rectangle (MBR) to represent them. Specifically, given a dataset $\mathbb{D}$, we can represent it to an R-tree T with internal nodes and leaf nodes. Each leaf node stores a multi-dimensional data record $\mathbf{x} \in \mathbb{D}$, and each internal node stores an MBR $\mathcal{P}$ and a set of child nodes, where $\mathcal{P}$ can cover all data records of its child nodes.

R-tree based multi-dimensional range query. R-tree can efficiently support multi-dimensional range queries. Let $\mathcal{Q}$ be a query range and T be an R-tree. Then, we can search on T for data records within $\mathcal{Q}$. As described in Algorithm 1, the query algorithm searches on T in a depth-first manner. Based on the type of the current searched nodes, we consider two cases in the query algorithm.

• Case 1: When the current node is a leaf node with the data record $\mathbf{x}$, we need to determine whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$. If $\mathbf{x} \in \mathcal{Q}$, we put $\mathbf{x}$ into the query result, i.e., $\mathbb{C} = \mathbb{C} \cup \{\mathbf{x}\}$.

• Case 2: When the current node is an internal node with an MBR $\mathcal{P}$. We need to determine whether $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. If $\mathcal{P} \cap \mathcal{Q} \neq \emptyset$, we continue to search its child nodes.

Algorithm 1 RQuery(Node $node$, Query range $\mathcal{Q}$)

```
// Let C be the query result.
if node is a leaf node with x then
    if x ∈ Q then
        C = C ∪ {x};
if node is an internal node with P then
    if P ∩ Q ≠ ∅ then
        for each child node (i.e., child) of node do
            RQuery(child, Q)
return C;
```

We can observe that R-tree based multi-dimensional range queries have two basic operations, i.e., (i) point intersection: determine whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$; and (ii) range intersection: determine $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. Meanwhile, the point intersection determination can be achieved by the range intersection determination. Specifically, let $\mathbf{x} = (x_1, x_2, \cdots, x_d)$ be a data point and $\mathcal{Q} = [q_{1,l}, q_{1,r}] \times [q_{2,l}, q_{2,r}] \times \cdots [q_{d,l}, q_{d,r}]$ be a query range. If we regard $\mathbf{x}$ as a range $\mathcal{P} = [x_1, x_1] \times [x_2, x_2] \times \cdots \times [x_d, x_d]$, determining $\mathbf{x} \in \mathcal{Q}$ is equivalent to determine $\mathcal{P} \cap \mathcal{Q} \neq \emptyset$. Thus, in our scheme, we only focus on the implementation of the multi-dimensional range intersection determination.

IV. Building Block Techniques

In this section, we first present our novel data comparison algorithm and homomorphic encoding technique. Then, we employ them to propose an encoding-based multi-dimensional range intersection algorithm.

A. Data Comparison Algorithm

Our novel data comparison algorithm is used to efficiently compare two non-negative integers p and q, where $p, q \in \{0,1\}^n$. The main idea is to represent p and q to vectors such that we can compare them via an equality test over the represented vectors. As shown in Algorithm 2, p and q can be compared as follows.

• ***Step 1:*** We represent p to its binary representation, i.e., $a_1 a_2 \cdots a_n = \mathrm{binary}(p)$. For example, the binary representation of 5 is 101, i.e., $\mathrm{binary}(5) = 101$. Based on $a_1 a_2 \cdots a_n$, we construct $2n$ vectors $\{\widehat{\mathbf{p}}_i, \overline{\mathbf{p}}_i\}_{i=1}^n$. Specifically, for each a_i $(1 \leq i \leq n)$, we construct a pair of vectors $\widehat{\mathbf{p}}_i$ and $\overline{\mathbf{p}}_i$ as

$$\widehat{\mathbf{p}}_i = \begin{cases} [1,0] & a_i = 0 \\ [0,1] & a_i = 1 \end{cases} \text{ and } \overline{\mathbf{p}}_i = \begin{cases} [0,1] & a_i = 0 \\ [0,0] & a_i = 1. \end{cases} \quad (2)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3158321, IEEE Internet of Things Journal

• ***Step 2:*** Let the binary representation of q be $b_1 b_2 \cdots b_n$. Based on $b_1 b_2 \cdots b_n$, we construct n vectors $\{\mathbf{q}_i\}_{i=1}^n$. Specifically, for each b_i ($1 \le i \le n$), we construct a vector $\mathbf{q}_i$ as

$$\mathbf{q}_i = \begin{cases} [1,0] & b_i = 0 \\ [0,1] & b_i = 1. \end{cases} \tag{3}$$

• ***Step 3:*** We compare p and q through the constructed vectors $\{\widehat{\mathbf{p}}_i, \overline{\mathbf{p}}_i\}_{i=1}^n$ and $\{\mathbf{q}_i\}_{i=1}^n$. Specifically, let $f_i(p,q) = (\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T$ and $f(p,q) = \sum_{i=1}^n f_i(p,q)$. We have

$$p < q \Leftrightarrow f(p,q) = 1;\ p \ge q \Leftrightarrow f(p,q) = 0. \tag{4}$$

Algorithm 2 Comparison(int p, int q)

$a_1 a_2 \cdots a_n = \text{binary}(p)$; $b_1 b_2 \cdots b_n = \text{binary}(q)$;
for i = 1 to n **do**
  if $a_i = 0$ **then**
    $\widehat{\mathbf{p}}_i = [1,0]$; $\overline{\mathbf{p}}_i = [0,1]$;
  else
    $\widehat{\mathbf{p}}_i = [0,1]$; $\overline{\mathbf{p}}_i = [0,0]$;
  if $b_i = 0$ **then**
    $\mathbf{q}_i = [1,0]$;
  else
    $\mathbf{q}_i = [0,1]$;
$f(p,q) = \sum_{i=1}^n f_i(p,q) = \sum_{i=1}^n ((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T)$;
if $f(p,q) = 1$ **then**
  return "$p < q$";
else
  return "$p \ge q$";

Theorem 1: The data comparison algorithm is correct.

Proof. The data comparison algorithm is correct *iff* Eq. (4) is correct, which can be proved from three cases as follows.

Case 1: $p < q$. When $p < q$, there exists a k such that $a_k < b_k$, and $\{a_i = b_i | i = 1, 2, \cdots, k-1\}$. From $a_k < b_k$, we can deduce that $a_k = 0$, and $b_k = 1$. Moreover, we have

$$\widehat{\mathbf{p}}_k \mathbf{q}_k^T = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0.$$

We further have $\sum_{i=k+1}^n ((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T) = 0$ and

$$f(p,q) = \sum_{i=1}^{n} ((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T) = \sum_{i=1}^{k} ((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T).$$

From $\{a_i = b_i | i = 1, 2, \cdots, k-1\}$, we can deduce that $a_i = b_i = 0$ or $a_i = b_i = 1$ for $i = 1, 2, \cdots, k-1$.

(1) When $a_i = b_i = 0$, we have

$$\overline{\mathbf{p}}_i \mathbf{q}_i^T = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 0; \quad \widehat{\mathbf{p}}_i \mathbf{q}_i^T = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 1.$$

(2) When $a_i = b_i = 1$, we have

$$\overline{\mathbf{p}}_i \mathbf{q}_i^T = \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0; \quad \widehat{\mathbf{p}}_i \mathbf{q}_i^T = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 1.$$

Thus, we can infer that $\overline{\mathbf{p}}_i \mathbf{q}_i^T = 0$ and $\widehat{\mathbf{p}}_i \mathbf{q}_i^T = 1$ for $1 \le i \le k-1$. Then, we have

$$f(p,q) = \sum_{i=1}^{k} ((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T) = (\prod_{j=1}^{k-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_k \mathbf{q}_k^T = \overline{\mathbf{p}}_k \mathbf{q}_k^T.$$

Since $a_k < b_k$, we have $a_k = 0$, $b_k = 1$, and $\overline{\mathbf{p}}_k \mathbf{q}_k^T = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 1$. That is, $f(p,q) = \overline{\mathbf{p}}_k \mathbf{q}_k^T = 1$.

Case 2: $p > q$. When $p > q$, there exists a k such that $a_k > b_k$, and $\{a_i = b_i | i = 1, 2, \cdots, k-1\}$. Similar to **Case 1**, we can deduce that $f(p,q) = \overline{\mathbf{p}}_k \mathbf{q}_k^T$. Since $a_k > b_k$, we can deduce that $a_k = 1$ and $b_k = 0$. Then, we have $\overline{\mathbf{p}}_k \mathbf{q}_k^T = \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 0$. That is, $f(p,q) = \overline{\mathbf{p}}_k \mathbf{q}_k^T = 0$.

Case 3: $p = q$. When $p = q$, we have $\{a_i = b_i\}_{i=1}^n$. Then, we can deduce that $\overline{\mathbf{p}}_i \mathbf{q}_i^T = 0$ and $\widehat{\mathbf{p}}_i \mathbf{q}_i^T = 1$ for $1 \le i \le n$. We further have $f(p,q) = \sum_{i=1}^n ((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T) = 0$. Therefore, Eq. (4) is correct. □

B. Matrix Rank Attack on Extended Hill Cipher Encryption

In this subsection, we first recall the extended Hill cipher encryption and then introduce the matrix rank attack on the extended Hill cipher encryption.

Extended Hill Cipher Encryption. The Hill cipher [22] is a matrix encryption technique that can encrypt a vector using an inverse matrix. Based on the idea of the Hill cipher, an extended Hill cipher encryption in [19] is utilized to protect the data privacy, where a message matrix is encrypted by an invertible matrix. Meanwhile, the extended Hill cipher encryption can be further extended to encrypt a message with two invertible matrices, which contains three algorithms, i.e., key generation, encryption, and decryption.

• $\mathsf{KeyGen}(d_1, d_2)$: Suppose the message matrix has the size of $d_1 \times d_2$. Then, in the key generation algorithm, we will generate two random matrices $\mathbf{M} \in \mathbb{R}^{d_1 \times d_1}$ and $\mathbf{W} \in \mathbb{R}^{d_2 \times d_2}$ as the secret keys, where $\mathbb{R}$ denotes the real domain.

• $\mathsf{Enc}(\mathbf{P}, \mathbf{M}, \mathbf{W})$: A message matrix $\mathbf{P}$ with the size of $d_1 \times d_2$ can be encrypted as $\text{CT}_\mathbf{P} = \mathbf{M}^{-1} * \mathbf{P} * \mathbf{W}$.

• $\mathsf{Dec}(\text{CT}_\mathbf{P}, \mathbf{M}, \mathbf{W})$: The message matrix $\mathbf{P}$ underlying a ciphertext $\text{CT}_\mathbf{P}$ can be recovered as $\mathbf{P} = \mathbf{M} * \text{CT}_\mathbf{P} * \mathbf{W}^{-1}$.

Matrix Rank Attack. The matrix rank attack is used to obtain some information about the message matrix by leveraging the rank of the encrypted matrix. Specifically, based on the property of the matrix multiplication, we have

$$\texttt{rank}(\text{CT}_\mathbf{P}) = \texttt{rank}(\mathbf{M}^{-1} * \mathbf{P} * \mathbf{W}) \le \texttt{rank}(\mathbf{P}).$$

If $\mathbf{P}$ is a full rank matrix, i.e., $\texttt{rank}(\mathbf{P}) = \min\{d_1, d_2\}$, $\text{CT}_\mathbf{P}$ will be a full rank matrix with a high probability. Otherwise, if $\mathbf{P}$ is not a full rank matrix, $\text{CT}_\mathbf{P}$ must not be a full rank matrix. In this case, if the domain of the ciphertext message is small, the rank may leak the plaintext information. For example, if a 5×7 matrix $\mathbf{P}$ is in the form of $\begin{bmatrix} \overline{\mathbf{p}}_j & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \overline{\mathbf{p}}_j & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{R}_j \end{bmatrix}$, where (i) $\overline{\mathbf{p}}_j$ is a 1×2 vector and is either $[0,0]$ or $[0,1]$; and (ii) $\mathbf{R}_j$ is a random 3×3 matrix. In this case, if we use the extended Hill cipher encryption to encrypt $\mathbf{P}$, $\texttt{rank}(\text{CT}_\mathbf{P})$ will leak the plaintext of $\overline{\mathbf{p}}_j$. Specifically, when $\overline{\mathbf{p}}_j = [0,0]$, $\texttt{rank}(\text{CT}_\mathbf{P})$ will be 3 with a high probability. When $\overline{\mathbf{p}}_j = [1,0]$, $\texttt{rank}(\text{CT}_\mathbf{P})$ will be 5 with a high probability. According to the description of our scheme in Section V, if we directly use the extended Hill cipher encryption to protect the privacy for the computation of $f(p,q) = \sum_{i=1}^n ((\prod_{j=1}^{i-1} \widehat{\mathbf{p}}_j \mathbf{q}_j^T) * \overline{\mathbf{p}}_i \mathbf{q}_i^T)$,

the information of the plaintext, such as $\overline{\mathbf{p}}_j$, will be leaked. Therefore, we design a homomorphic encoding technique in the next subsection to prevent the matrix rank attack on the extend Hill cipher encryption.

C. Homomorphic Encoding Technique

The homomorphic encoding technique is designed for encoding data. Since it can encode the zero value to non-zero random integers, we will use it as a building block to prevent our PMRQ scheme from the matrix rank attack in Subsection IV-B. Formally, the scheme can be defined as $\Pi_{\mathsf{HE}} = (\mathsf{HE.Setup}, \mathsf{HE.Encode}, \mathsf{HE.Decode})$.

- $\mathsf{HE.Setup}(w)$: On input an encoding parameter w, the setup algorithm chooses a prime integer $\mathtt{L} \in \{0,1\}^w$ as the encoding key. Then, it sets the message space as $\mathcal{M} = \{\mathtt{m} | 0 \leq \mathtt{m} < \mathtt{L}\}$. Finally, it outputs $\{\mathtt{L}, \mathcal{M}\}$.
- $\mathsf{HE.Encode}(\mathtt{L}, \mathtt{m})$: The encoding algorithm uses $\mathtt{L}$ to encode a plaintext message $\mathtt{m}$ as $\mathtt{E}_{\mathtt{m}} = \mathtt{m} + \mathtt{r} * \mathtt{L}$, where $\mathtt{r}$ is a non-zero integer.
- $\mathsf{HE.Decode}(\mathtt{L}, \mathtt{E}_{\mathtt{m}})$: The decoding algorithm uses $\mathtt{L}$ to decode an encoded value $\mathtt{E}_{\mathtt{m}}$ as $\mathtt{m} = \mathtt{E}_{\mathtt{m}} \mod \mathtt{L}$.

Correctness. The encoding technique is correct because $\mathtt{E}_{\mathtt{m}} \mod \mathtt{L} = (\mathtt{m} + \mathtt{r} * \mathtt{L}) \mod \mathtt{L} = \mathtt{m}$ $(\because 0 \leq \mathtt{m} < \mathtt{L})$.

Homomorphic properties. Given two encoding values $\mathtt{E}_{\mathtt{m}_1}$ and $\mathtt{E}_{\mathtt{m}_2}$, they satisfy (i) homomorphic addition property: $\mathtt{E}_{\mathtt{m}_1} + \mathtt{E}_{\mathtt{m}_2} \rightarrow \mathtt{E}_{\mathtt{m}_1+\mathtt{m}_2}$; and (ii) homomorphic multiplication property: $\mathtt{E}_{\mathtt{m}_1} * \mathtt{E}_{\mathtt{m}_2} \rightarrow \mathtt{E}_{\mathtt{m}_1*\mathtt{m}_2}$, where $\mathtt{m}_1 + \mathtt{m}_2 < \mathtt{L}$ and $\mathtt{m}_1 * \mathtt{m}_2 < \mathtt{L}$.

Vector encoding. By default, we encode a vector $\mathbf{x} = (x_1, x_2, \cdots, x_d)$ by separately encoding each x_i as

$$\mathtt{E}_{\mathbf{x}} = (\mathtt{E}_{x_1}, \mathtt{E}_{x_2}, \cdots, \mathtt{E}_{x_d}). \tag{5}$$

D. Encoding-based Range Intersection Algorithm

Based on the data comparison algorithm and encoding technique, we first introduce an encoding-based data comparison algorithm, and then leverage it to design an encoding-based multi-dimensional range intersection algorithm.

- **Encoding-based data comparison algorithm.** Given two integers p and q, we can respectively represent them to $2n$ vectors $\{\widehat{\mathbf{p}}_i, \overline{\mathbf{p}}_i\}_{i=1}^n$ as Eq. (2) and n vectors $\{\mathbf{q}_i\}_{i=1}^n$ as Eq. (3). Then, we can encode each $\widehat{\mathbf{p}}_i$, $\overline{\mathbf{p}}_i$ and $\mathbf{q}_i$ into vectors $\mathtt{E}_{\widehat{\mathbf{p}}_i}$, $\mathtt{E}_{\overline{\mathbf{p}}_i}$ and $\mathtt{E}_{\mathbf{q}_i}$ as Eq. (5). Based on the homomorphic properties of the encoding technique, we have $\mathtt{E}_{f(p,q)} = \sum_{i=1}^n ((\prod_{j=1}^{i-1} \mathtt{E}_{\widehat{\mathbf{p}}_j} \mathtt{E}_{\mathbf{q}_j}^T) * \mathtt{E}_{\overline{\mathbf{p}}_i} \mathtt{E}_{\mathbf{q}_i}^T)$. Then, we can deduce that

$$\begin{cases} f(p,q) = 1 \Leftrightarrow \mathtt{E}_{f(p,q)} \mod \mathtt{L} = 1 \\ f(p,q) = 0 \Leftrightarrow \mathtt{E}_{f(p,q)} \mod \mathtt{L} = 0. \end{cases}$$

Based on Eq. (4), we further have

$$\begin{cases} p < q \Leftrightarrow \mathtt{E}_{f(p,q)} \mod \mathtt{L} = 1 \\ p \geq q \Leftrightarrow \mathtt{E}_{f(p,q)} \mod \mathtt{L} = 0. \end{cases}$$

- **Encoding-based multi-dimensional range intersection algorithm.** Let $\mathcal{P} = P_1 \times P_2 \times \cdots \times P_d$ and $\mathcal{Q} = Q_1 \times Q_2 \times \cdots \times Q_d$ be two multi-dimensional ranges, where $P_k = [p_{k,l}, p_{k,r}]$ and $Q_k = [q_{k,l}, q_{k,r})$. The encoding-based multi-dimensional range intersection algorithm is to determine whether $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. First, we have

$$\begin{aligned} &\mathcal{P} \cap \mathcal{Q} \neq \emptyset \Leftrightarrow P_k \cap Q_k \neq \emptyset \text{ for } 1 \leq k \leq d \\ &\Leftrightarrow p_{k,r} \geq q_{k,l} \text{ and } p_{k,l} < q_{k,r} \text{ for } 1 \leq k \leq d \\ &\Leftrightarrow f(p_{k,r}, q_{k,l}) = 0 \text{ and } f(p_{k,l}, q_{k,r}) = 1 \text{ for } 1 \leq k \leq d. \end{aligned}$$

Let $\{s_k, t_k\}_{k=1}^d$ be $2d$ random non-zero integers. Then, with a high probability, we have

$$\begin{aligned} &\mathcal{P} \cap \mathcal{Q} \neq \emptyset \\ &\Leftrightarrow \sum_{k=1}^d (s_k * f(p_{k,r}, q_{k,l}) + t_k * (f(p_{k,l}, q_{k,r}) - 1)) = 0 \\ &\Leftrightarrow \sum_{k=1}^d \left(s_k * \sum_{i=1}^n f_i(p_{k,r}, q_{k,l}) + t_k * \sum_{i=1}^n \left(f_i(p_{k,l}, q_{k,r}) - \frac{1}{n} \right) \right) = 0 \\ &\Leftrightarrow \sum_{k=1}^d \sum_{i=1}^n \left(s_k * f_i(p_{k,r}, q_{k,l}) + t_k * \left(f_i(p_{k,l}, q_{k,r}) - \frac{1}{n} \right) \right) = 0. \end{aligned}$$

Based on the encoding-based data comparison algorithm, we have $f_i(p_{k,r}, q_{k,l}) = \mathtt{E}_{f_i(p_{k,r}, q_{k,l})} \mod \mathtt{L}$ and $f_i(p_{k,l}, q_{k,r}) = \mathtt{E}_{f_i(p_{k,l}, q_{k,r})} \mod \mathtt{L}$. We can deduce that

$$\begin{aligned} &\mathcal{P} \cap \mathcal{Q} \neq \emptyset \\ &\Leftrightarrow \sum_{k=1}^d \sum_{i=1}^n (s_k * \mathtt{E}_{f_i(p_{k,r}, q_{k,l})} + t_k * (\mathtt{E}_{f_i(p_{k,l}, q_{k,r})} - \frac{1}{n})) \mod \mathtt{L} = 0. \end{aligned}$$

Without loss of generality, let $\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} = s_k * \mathtt{E}_{f_i(p_{k,r}, q_{k,l})} + t_k * (\mathtt{E}_{f_i(p_{k,l}, q_{k,r})} - \frac{1}{n})$. We have

$$\mathcal{P} \cap \mathcal{Q} \neq \emptyset \Leftrightarrow \sum_{k=1}^d \sum_{i=1}^n \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} \mod \mathtt{L} = 0. \tag{6}$$

Remark. In the encoding-based data comparison algorithm, determining $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ is implemented by determining whether $\sum_{k=1}^d \sum_{i=1}^n \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} \mod \mathtt{L} \stackrel{?}{=} 0$. Based on our derivation, when $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$, we must have $\sum_{k=1}^d \sum_{i=1}^n \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} \mod \mathtt{L} = 0$. However, $\sum_{k=1}^d \sum_{i=1}^n \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}$ is an integer modulo $\mathtt{L}$. Even if $\mathcal{P} \cap \mathcal{Q} \neq \emptyset$, there is still a probability such that $\sum_{k=1}^d \sum_{i=1}^n \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} = 0$. Since $\sum_{k=1}^d \sum_{i=1}^n \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}$ is in the range of $[0, \mathtt{L}-1]$, the probability will be $\frac{1}{\mathtt{L}}$. When $\mathtt{L}$ is large, the probability is negligible.

V. Our Proposed PMRQ Scheme

In this section, we propose our PMRQ scheme. Before introducing the details, we first present a multi-dimensional range intersection predicate encryption (MRIPE) scheme, which serves as the key component of our PMRQ scheme.

A. MRIPE Scheme

The MRIPE scheme is designed to privately determine whether $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. The main idea is to encrypt $\mathcal{P}$ and $\mathcal{Q}$ into a ciphertext $\mathrm{CT}_{\mathcal{P}}$ and a token $\mathrm{TK}_{\mathcal{Q}}$ such that we can determine $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ through $\mathrm{CT}_{\mathcal{P}}$ and $\mathrm{TK}_{\mathcal{Q}}$. Based on Eq. (6), we have $\mathcal{P} \cap \mathcal{Q} \neq \emptyset \Leftrightarrow \sum_{k=1}^d \sum_{i=1}^n \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} \mod \mathtt{L} = 0$. If we choose some random numbers $\{\{z_{k,i}\}_{i=1}^n\}_{k=1}^d$ satisfying $\sum_{k=1}^d \sum_{i=1}^n z_{k,i} = 0$, we have $\sum_{k=1}^d \sum_{i=1}^n (\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} +$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3158321, IEEE Internet of Things Journal

$z_{k,i}) = \sum_{k=1}^{d}\sum_{i=1}^{n} \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}$. In our scheme, we decompose the problem of computing $\sum_{k=1}^{d}\sum_{i=1}^{n} \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}$ into that of computing each $\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} + z_{k,i}$ for $1 \le k \le d$ and $1 \le i \le n$. It is worth noting that random numbers $\{\{z_{k,i}\}_{i=1}^{n}\}_{k=1}^{d}$ contribute to preserving the single-dimensional privacy because they can be canceled only all of them are summed together. Meanwhile, since we have represented $\{\mathcal{P},\mathcal{Q}\}$ to vectors, we employ the matrix encryption to preserve the privacy of the MRIPE scheme. Formally, the MRIPE scheme Π_{MRIPE} = (MRIPE.KeyGen, MRIPE.RangeEnc, MRIPE.TokenGen, MRIPE.RangeEval) is defined as follows.

• MRIPE.KeyGen(d, n, w) : Let d be the number of dimensions in $\{\mathcal{P},\mathcal{Q}\}$, n be the bit length of values in $\{\mathcal{P},\mathcal{Q}\}$, and w be an encoding parameter. The key generation algorithm generates a set of invertible matrices $\{\{\{\mathbf{M}_{k,i,j}, \mathbf{W}_{k,i,j}\}_{j=1}^{i}\}_{i=1}^{n}\}_{k=1}^{d}$ as the secret key sk, where $\mathbf{M}_{k,i,j} \in \mathbb{R}^{5\times5}$ and $\mathbf{W}_{k,i,j} \in \mathbb{R}^{7\times7}$. Then, it generates an encoding key as $\mathtt{L} \leftarrow \mathsf{HE.Setup}(w)$.

• MRIPE.RangeEnc$(sk, \mathtt{L}, \mathcal{P} = P_1 \times P_2 \times \cdots \times P_d)$: The range encryption algorithm encrypts a range $\mathcal{P}$ as follows.

Step 1: For each range $P_k = [p_{k,l}, p_{k,r}]$, we represent $p_{k,l}$ to $2n$ vectors $\{\widehat{\mathbf{p}}_{k,l,i}, \overline{\mathbf{p}}_{k,l,i}\}_{i=1}^{n}$ and $p_{k,r}$ to $2n$ vectors $\{\widehat{\mathbf{p}}_{k,r,i}, \overline{\mathbf{p}}_{k,r,i}\}_{i=1}^{n}$ as Eq. (2). For each $\widehat{\mathbf{p}}_{k,l,i}$, $\overline{\mathbf{p}}_{k,l,i}$, $\widehat{\mathbf{p}}_{k,r,i}$ and $\overline{\mathbf{p}}_{k,r,i}$, we use $\mathtt{L}$ to encode them into vectors $\mathtt{E}_{\widehat{\mathbf{p}}_{k,l,i}}$, $\mathtt{E}_{\overline{\mathbf{p}}_{k,l,i}}$, $\mathtt{E}_{\widehat{\mathbf{p}}_{k,r,i}}$, and $\mathtt{E}_{\overline{\mathbf{p}}_{k,r,i}}$ as Eq. (5).

Step 2: We use the encoded vectors $\{\{\mathtt{E}_{\widehat{\mathbf{p}}_{k,l,i}}, \mathtt{E}_{\overline{\mathbf{p}}_{k,l,i}}, \mathtt{E}_{\widehat{\mathbf{p}}_{k,r,i}}, \mathtt{E}_{\overline{\mathbf{p}}_{k,r,i}}\}_{i=1}^{n}\}_{k=1}^{d}$ to construct some vectors and matrices.

(1) We choose $4d$ non-zero random integers $\{s_k^{\mathcal{P}}, t_k^{\mathcal{P}}, s_k^{\mathcal{P}'}, t_k^{\mathcal{P}'}\}_{k=1}^{d}$ and $d*n$ random real numbers $\{\{z_{k,i}^{\mathcal{P}}, z_{k,i}^{\mathcal{P}'}\}_{i=1}^{n}\}_{k=1}^{d}$, where $\sum_{k=1}^{d}\sum_{i=1}^{n}(z_{k,i}^{\mathcal{P}} + z_{k,i}^{\mathcal{P}'}) = 0$.

(2) For each pair (k,i), we construct a vector $\boldsymbol{\gamma}_{k,i}$ and i matrices $\{\mathbf{P}_{k,i,j}\}_{j=1}^{i}$ as

$$\boldsymbol{\gamma}_{k,i} = \begin{bmatrix} s_{k,i,0}^{\mathcal{P}} & u_{k,i,0}^{\mathcal{P}} & v_{k,i,0}^{\mathcal{P}} & z_{k,i,0}^{\mathcal{P}} & 1 \end{bmatrix};$$

$$\mathbf{P}_{k,i,j} = \begin{bmatrix} s_{k,i,j}^{\mathcal{P}} * \mathtt{E}_{\widehat{\mathbf{p}}_{k,r,j}} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & u_{k,i,j}^{\mathcal{P}} * \mathtt{E}_{\widehat{\mathbf{p}}_{k,l,j}} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & v_{k,i,j}^{\mathcal{P}} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & z_{k,i,j}^{\mathcal{P}} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} & 1 \end{bmatrix}$$

for $1 \le j \le i-1$;

$$\mathbf{P}_{k,i,i} = \begin{bmatrix} s_{k,i,i}^{\mathcal{P}} * \mathtt{E}_{\overline{\mathbf{p}}_{k,r,i}} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & u_{k,i,i}^{\mathcal{P}} * \mathtt{E}_{\overline{\mathbf{p}}_{k,l,i}} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & v_{k,i,i}^{\mathcal{P}} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & z_{k,i,i}^{\mathcal{P}} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} & 1 \end{bmatrix},$$

where $\{s_{k,i,j}^{\mathcal{P}}, u_{k,i,j}^{\mathcal{P}}, v_{k,i,j}^{\mathcal{P}}, z_{k,i,j}^{\mathcal{P}}\}_{j=0}^{i}$ are random numbers and satisfy

$$\prod_{j=0}^{i} s_{k,i,j}^{\mathcal{P}} = s_k^{\mathcal{P}};\ \prod_{j=0}^{i} u_{k,i,j}^{\mathcal{P}} = \prod_{j=0}^{i} v_{k,i,j}^{\mathcal{P}} = t_k^{\mathcal{P}};\ \prod_{j=0}^{i} z_{k,i,j}^{\mathcal{P}} = z_{k,i}^{\mathcal{P}}.$$

(3) For each pair (k,i), we construct a vector $\boldsymbol{\gamma}'_{k,i}$ and i matrices $\{\mathbf{P}'_{k,i,j}\}_{j=1}^{i}$. The constructing approach is the same as that of $\boldsymbol{\gamma}_{k,i}$ and $\{\mathbf{P}_{k,i,j}\}_{j=1}^{i}$. Differently, the chosen random numbers are $\{s_{k,i,j}^{\mathcal{P}'}, u_{k,i,j}^{\mathcal{P}'}, v_{k,i,j}^{\mathcal{P}'}, z_{k,i,j}^{\mathcal{P}'}\}_{j=0}^{i}$ and satisfy

$$\prod_{j=0}^{i} s_{k,i,j}^{\mathcal{P}'} = s_k^{\mathcal{P}'};\ \prod_{j=0}^{i} u_{k,i,j}^{\mathcal{P}'} = \prod_{j=0}^{i} v_{k,i,j}^{\mathcal{P}'} = t_k^{\mathcal{P}'};\ \prod_{j=0}^{i} z_{k,i,j}^{\mathcal{P}'} = z_{k,i}^{\mathcal{P}'}.$$

Step 3: We encrypt $\boldsymbol{\gamma}_{k,i}$, $\boldsymbol{\gamma}'_{k,i}$, $\mathbf{P}_{k,i,j}$ and $\mathbf{P}'_{k,i,j}$ as

$$\begin{cases} \mathrm{CT}_{\boldsymbol{\gamma}_{k,i}} = \boldsymbol{\gamma}_{k,i}\mathbf{M}_{k,i,1};\ \mathrm{CT}_{\mathbf{P}_{k,i,j}} = \mathbf{M}_{k,i,j}^{-1}\mathbf{P}_{k,i,j}\mathbf{W}_{k,i,j}; \\ \mathrm{CT}_{\boldsymbol{\gamma}'_{k,i}} = \boldsymbol{\gamma}'_{k,i}\mathbf{M}_{k,i,1};\ \mathrm{CT}_{\mathbf{P}'_{k,i,j}} = \mathbf{M}_{k,i,j}^{-1}\mathbf{P}'_{k,i,j}\mathbf{W}_{k,i,j}. \end{cases}$$

Finally, the algorithm outputs the ciphertext $\mathrm{CT}_{\mathcal{P}} = \{\{\mathrm{CT}_{\boldsymbol{\gamma}_{k,i}}, \mathrm{CT}_{\boldsymbol{\gamma}'_{k,i}}, \{\mathrm{CT}_{\mathbf{P}_{k,i,j}}, \mathrm{CT}_{\mathbf{P}'_{k,i,j}}\}_{j=1}^{i}\}_{i=1}^{n}\}_{k=1}^{d}$.

• MRIPE.TokenGen$(sk, \mathtt{L}, \mathcal{Q} = Q_1 \times Q_2 \times \cdots \times Q_d)$: The token generation algorithm encrypts a query range $\mathcal{Q}$ into a query token as follows.

Step 1: For each $Q_k = [q_{k,l}, q_{k,r})$, we respectively represent $q_{k,l}$ and $q_{k,r}$ to n vectors $\{\mathbf{q}_{k,l,i}\}_{i=1}^{n}$ and n vectors $\{\mathbf{q}_{k,r,i}\}_{i=1}^{n}$ as Eq. (3). For each $\mathbf{q}_{k,l,i}$ and $\mathbf{q}_{k,r,i}$, we use $\mathtt{L}$ to encode them into new vectors $\mathtt{E}_{\mathbf{q}_{k,l,i}}$ and $\mathtt{E}_{\mathbf{q}_{k,r,i}}$ as Eq. (5).

Step 2: We use the encoded vectors $\{\{\mathtt{E}_{\mathbf{q}_{k,l,i}}, \mathtt{E}_{\mathbf{q}_{k,r,i}}\}_{i=1}^{n}\}_{k=1}^{d}$ to construct some vectors and matrices.

(1) We choose $4d$ non-zero random integers $\{s_k^{\mathcal{Q}}, t_k^{\mathcal{Q}}, s_k^{\mathcal{Q}'}, t_k^{\mathcal{Q}'}\}_{k=1}^{d}$ and $d*n$ random numbers $\{\{z_{k,i}^{\mathcal{Q}}, z_{k,i}^{\mathcal{Q}'}\}_{i=1}^{n}\}_{k=1}^{d}$, where $\sum_{k=1}^{d}\sum_{i=1}^{n}(z_{k,i}^{\mathcal{Q}} + z_{k,i}^{\mathcal{Q}'}) = 0$.

(2) For each pair (k,i), we construct $i-1$ matrices $\{\mathbf{Q}_{k,i,j}\}_{j=1}^{i-1}$ and a vector $\boldsymbol{\beta}_{k,i}$ as

$$\mathbf{Q}_{k,i,j} = \begin{bmatrix} s_{k,i,j}^{\mathcal{Q}} * \mathtt{E}_{\mathbf{q}_{k,l,j}}^{T} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & u_{k,i,j}^{\mathcal{Q}} * \mathtt{E}_{\mathbf{q}_{k,r,j}}^{T} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & v_{k,i,j}^{\mathcal{Q}} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & 1 & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} & z_{k,i,j}^{\mathcal{Q}} \end{bmatrix}$$

for $1 \le j \le i-1$;

$$\boldsymbol{\beta}_{k,i} = \begin{bmatrix} s_{k,i,i}^{\mathcal{Q}} * \mathtt{E}_{\mathbf{q}_{k,l,i}}^{T} & u_{k,i,i}^{\mathcal{Q}} * \mathtt{E}_{\mathbf{q}_{k,r,i}}^{T} & -v_{k,i,i}^{\mathcal{Q}} * \frac{1}{n} & 1 & z_{k,i,i}^{\mathcal{Q}} \end{bmatrix}^{T},$$

where $\{s_{k,i,j}^{\mathcal{Q}}, u_{k,i,j}^{\mathcal{Q}}, v_{k,i,j}^{\mathcal{Q}}, z_{k,i,j}^{\mathcal{Q}}\}_{j=1}^{i}$ are random numbers and satisfy

$$\prod_{j=1}^{i} s_{k,i,j}^{\mathcal{Q}} = s_k^{\mathcal{Q}};\ \prod_{j=1}^{i} u_{k,i,j}^{\mathcal{Q}} = \prod_{j=1}^{i} v_{k,i,j}^{\mathcal{Q}} = t_k^{\mathcal{Q}};\ \prod_{j=1}^{i} z_{k,i,j}^{\mathcal{Q}} = z_{k,i}^{\mathcal{Q}}.$$

(3) For each pair (k,i), we further construct $i-1$ matrices $\{\mathbf{Q}'_{k,i,j}\}_{j=1}^{i-1}$ and a vector $\boldsymbol{\beta}'_{k,i}$. The constructing approach is the same as that of $\{\mathbf{Q}_{k,i,j}\}_{j=1}^{i-1}$ and $\boldsymbol{\beta}_{k,i}$. Differently, the chosen random numbers are $\{s_{k,i,j}^{\mathcal{Q}'}, u_{k,i,j}^{\mathcal{Q}'}, v_{k,i,j}^{\mathcal{Q}'}, z_{k,i,j}^{\mathcal{Q}'}\}_{j=1}^{i}$ and satisfy

$$\prod_{j=1}^{i} s_{k,i,j}^{\mathcal{Q}'} = s_k^{\mathcal{Q}'};\ \prod_{j=1}^{i} u_{k,i,j}^{\mathcal{Q}'} = \prod_{j=1}^{i} v_{k,i,j}^{\mathcal{Q}'} = t_k^{\mathcal{Q}'};\ \prod_{j=1}^{i} z_{k,i,j}^{\mathcal{Q}'} = z_{k,i}^{\mathcal{Q}'}.$$

Step 3: We encrypt $\mathbf{Q}_{k,i,j}$, $\mathbf{Q}'_{k,i,j}$, $\boldsymbol{\beta}_{k,i}$ and $\boldsymbol{\beta}'_{k,i}$ as

$$\begin{cases} \mathrm{TK}_{\mathbf{Q}_{k,i,j}} = \mathbf{W}_{k,i,j}^{-1}\mathbf{Q}_{k,i,j}\mathbf{M}_{k,i,j+1};\ \mathrm{TK}_{\boldsymbol{\beta}_{k,i}} = \mathbf{W}_{k,i,i}^{-1}\boldsymbol{\beta}_{k,i}; \\ \mathrm{TK}_{\mathbf{Q}'_{k,i,j}} = \mathbf{W}_{k,i,j}^{-1}\mathbf{Q}'_{k,i,j}\mathbf{M}_{k,i,j+1};\ \mathrm{TK}_{\boldsymbol{\beta}'_{k,i}} = \mathbf{W}_{k,i,i}^{-1}\boldsymbol{\beta}'_{k,i}. \end{cases}$$

Finally, the algorithm outputs the query token $\mathrm{TK}_{\mathcal{Q}} = \{\{\{\mathrm{TK}_{\mathbf{Q}_{k,i,j}}, \mathrm{TK}_{\mathbf{Q}'_{k,i,j}}\}_{j=1}^{i-1}, \mathrm{TK}_{\boldsymbol{\beta}_{k,i}}, \mathrm{TK}_{\boldsymbol{\beta}'_{k,i}}\}_{i=1}^{n}\}_{k=1}^{d}$.

• MRIPE.RangeEval$(\mathtt{L}, \mathrm{CT}_{\mathcal{P}}, \mathrm{TK}_{\mathcal{Q}})$: In the range intersection evaluation algorithm, on input the ciphertext $\mathrm{CT}_{\mathcal{P}}$ and the token $\mathrm{TK}_{\mathcal{Q}}$, we first compute

$$\begin{aligned} \mathrm{CT}_{\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}} = {} & \mathrm{CT}_{\boldsymbol{\gamma}_{k,i}}(\prod_{j=1}^{i-1} \mathrm{CT}_{\mathbf{P}_{k,i,j}}\mathrm{TK}_{\mathbf{Q}_{k,i,j}})\mathrm{CT}_{\mathbf{P}_{k,i,i}}\mathrm{TK}_{\boldsymbol{\beta}_{k,i}} \\ & + \mathrm{CT}_{\boldsymbol{\gamma}'_{k,i}}(\prod_{j=1}^{i-1} \mathrm{CT}_{\mathbf{P}'_{k,i,j}}\mathrm{TK}_{\mathbf{Q}'_{k,i,j}})\mathrm{CT}_{\mathbf{P}'_{k,i,i}}\mathrm{TK}_{\boldsymbol{\beta}'_{k,i}}. \end{aligned}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3158321, IEEE Internet of Things Journal

Then, we compute $\mathrm{CT}_{\mathtt{E}_{f(\mathcal{P},\mathcal{Q})}} = \sum_{k=1}^{d}\sum_{i=1}^{n}\mathrm{CT}_{\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}}$. If $\mathrm{CT}_{\mathtt{E}_{f(\mathcal{P},\mathcal{Q})}} \bmod \mathtt{L} = 0$, the algorithm returns 1 to denote "$\mathcal{P}\cap\mathcal{Q}\neq\emptyset$". Otherwise, it returns 0 to denote "$\mathcal{P}\cap\mathcal{Q}=\emptyset$".

Correctness. We prove the correctness of the MRIPE scheme as follows. First, it is easy to deduce that

$$\begin{aligned}&\mathrm{CT}_{\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}}\\ &= (s_k^{\mathcal{P}} * s_k^{\mathcal{Q}} + s_k^{\mathcal{P}'} * s_k^{\mathcal{Q}'}) * \mathtt{E}_{f_i(p_{k,r},q_{k,l})} + (t_k^{\mathcal{P}} * t_k^{\mathcal{Q}} + t_k^{\mathcal{P}'} * t_k^{\mathcal{Q}'}) *\\ &\quad (\mathtt{E}_{f_i(p_{k,l},q_{k,r})} - \frac{1}{n}) + z_{k,i}^{\mathcal{P}} + z_{k,i}^{\mathcal{Q}} + z_{k,i}^{\mathcal{P}'} + z_{k,i}^{\mathcal{Q}'}.\end{aligned}$$

Let $s_k^{\mathcal{P}} * s_k^{\mathcal{Q}} + s_k^{\mathcal{P}'} * s_k^{\mathcal{Q}'} \triangleq s_k$, $t_k^{\mathcal{P}} * t_k^{\mathcal{Q}} + t_k^{\mathcal{P}'} * t_k^{\mathcal{Q}'} \triangleq t_k$, and $z_{k,i}^{\mathcal{P}} + z_{k,i}^{\mathcal{Q}} + z_{k,i}^{\mathcal{P}'} + z_{k,i}^{\mathcal{Q}'} \triangleq z_{k,i}$. We can deduce that

$$\begin{aligned}\mathrm{CT}_{\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}} &= s_k * \mathtt{E}_{f_i(p_{k,r},q_{k,l})} + t_k * (\mathtt{E}_{f_i(p_{k,l},q_{k,r})} - \frac{1}{n}) + z_{k,i}\\ &= \mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} + z_{k,i}.\end{aligned}$$

Since $\sum_{k=1}^{d}\sum_{i=1}^{n} z_{k,i} = \sum_{k=1}^{d}\sum_{i=1}^{n}(z_{k,i}^{\mathcal{P}} + z_{k,i}^{\mathcal{Q}} + z_{k,i}^{\mathcal{P}'} + z_{k,i}^{\mathcal{Q}'}) = 0$, we further have

$$\sum_{k=1}^{d}\sum_{i=1}^{n}\mathrm{CT}_{\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}} = \sum_{k=1}^{d}\sum_{i=1}^{n}(\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} + z_{k,i}) = \sum_{k=1}^{d}\sum_{i=1}^{n}\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}.$$

Based on Eq. (6), we have

$$\begin{aligned}&\sum_{k=1}^{d}\sum_{i=1}^{n}\mathrm{CT}_{\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}} \bmod \mathtt{L} = 0\\ \Leftrightarrow\ &\sum_{k=1}^{d}\sum_{i=1}^{n}\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})} \bmod \mathtt{L} = 0 \Leftrightarrow \mathcal{P}\cap\mathcal{Q}\neq\emptyset.\end{aligned}$$

As a result, the MRIPE scheme is correct.

Remark. Our MRIPE scheme is a probabilistic encryption scheme since the ciphertexts and query tokens are generated by involving many random numbers.

B. Description of the PMRQ Scheme

Based on the MRIPE scheme, we introduce our PMRQ scheme in detail, which can be defined as follows.

• $\mathsf{PMRQ.KeyGen}(d, n, w)$: In the key generation algorithm, the HC first runs $\mathsf{MRIPE.KeyGen}(d, n, w)$ to generate a secret key and an encoding key $\{sk, \mathtt{L}\} \leftarrow \mathsf{MRIPE.KeyGen}(d, n, w)$. Then, it generates an access key K for the AES algorithm. Finally, the HC publishes $\mathtt{L}$ and sends $\{sk, K\}$ to query doctors as the authorized key.

• $\mathsf{PMRQ.Enc}(sk, \mathtt{L}, K, \mathbb{D} = \{\mathbf{x}_i\}_{i=1}^{N})$: In the encryption algorithm, the HC encrypts its dataset $\mathbb{D} = \{\mathbf{x}_i\}_{i=1}^{N}$ as follows.

Step 1: The HC builds an R-tree T for the dataset $\mathbb{D}$. Then, it encrypts the R-tree T. Specifically, for each internal node with an MBR $\mathcal{P}$, it encrypts $\mathcal{P}$ into a ciphertext $\mathrm{CT}_{\mathcal{P}}$ as

$$\mathrm{CT}_{\mathcal{P}} \leftarrow \mathsf{MRIPE.RangeEnc}(sk, \mathtt{L}, \mathcal{P}).$$

For each leaf node with a record $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,d})$, the HC represents $\mathbf{x}_i$ to a range $\mathcal{P}_{\mathbf{x}_i} = [x_{i,1}, x_{i,1}] \times [x_{i,2}, x_{i,2}] \times \cdots \times [x_{i,d}, x_{i,d}]$. Then, it encrypts $\mathcal{P}_{\mathbf{x}_i}$ into two ciphertexts $\{\mathrm{CT}_{\mathbf{x}_i}, \mathrm{AES}_K(\mathbf{x}_i)\}$, where

$$\mathrm{CT}_{\mathbf{x}_i} \leftarrow \mathsf{MRIPE.RangeEnc}(sk, \mathtt{L}, \mathcal{P}_{\mathbf{x}_i}).$$

Step 2: The HC sends the encrypted R-tree, denoted as $E(\mathrm{T})$, to the cloud server.

• $\mathsf{PMRQ.TokenGen}(sk, \mathtt{L}, \mathcal{Q})$: Given a query range $\mathcal{Q}$, the query doctor first generates a query token $\mathrm{TK}_{\mathcal{Q}}$ as

$$\mathrm{TK}_{\mathcal{Q}} \leftarrow \mathsf{MRIPE.TokenGen}(sk, \mathtt{L}, \mathcal{Q}).$$

Then, it sends the query token $\mathrm{TK}_{\mathcal{Q}}$ to the cloud server.

• $\mathsf{PMRQ.Query}(\mathtt{L}, E(\mathrm{T}), \mathrm{TK}_{\mathcal{Q}})$: In the query algorithm, the cloud server uses the token $\mathrm{TK}_{\mathcal{Q}}$ to search on $E(\mathrm{T})$ for records within $\mathcal{Q}$. The algorithm is similar to that over the plaintext R-tree in Algorithm 1. Differently, the conditions $\mathbf{x} \in \mathcal{Q}$ and $\mathcal{P}\cap\mathcal{Q}\neq\emptyset$ are replaced with $\mathsf{MRIPE.RangeEval}(\mathtt{L}, \mathrm{CT}_{\mathbf{x}}, \mathrm{TK}_{\mathcal{Q}}) = 1$ and $\mathsf{MRIPE.RangeEval}(\mathtt{L}, \mathrm{CT}_{\mathcal{P}}, \mathrm{TK}_{\mathcal{Q}}) = 1$, respectively. Finally, the cloud returns the result $\mathbb{C}_{\mathtt{cipher}} = \{\mathrm{AES}_K(\mathbf{x}_i) | \mathsf{MRIPE.RangeEval}(\mathtt{L}, \mathrm{CT}_{\mathbf{x}_i}, \mathrm{TK}_{\mathcal{Q}}) = 1\}$ to the query doctor. On receiving $\mathbb{C}_{\mathtt{cipher}}$, the query doctor recovers each $\mathbf{x}_i$ with the access key K by decrypting $\mathrm{AES}_K(\mathbf{x}_i) \in \mathbb{C}_{\mathtt{cipher}}$.

VI. Security Analysis

In this section, we show that our PMRQ scheme is privacy-preserving. Since the PMRQ scheme is designed based on the MRIPE scheme, we first prove the MRIPE scheme's security.

A. Security of MRIPE Scheme

Since the MRIPE scheme is a searchable encryption scheme, we prove its security under the real/ideal worlds model [23]. In the real world, the views of the adversary are ciphertexts and tokens generated by the MRIPE scheme. In the ideal world, the views of the adversary are ciphertexts and tokens generated by a simulator with the MRIPE scheme's leakage function. Before formalizing the ideal world, we first define the leakage function of our MRIPE scheme. Given two ranges $\mathcal{P}$ and $\mathcal{Q}$, the leakage function is $\mathcal{L}(\mathcal{P}, \mathcal{Q}) = \mathsf{MRIPE.RangeEval}(\mathtt{L}, \mathrm{CT}_{\mathcal{P}}, \mathrm{TK}_{\mathcal{Q}})$. Next, we formalize the ideal world.

Ideal world. In the ideal world, a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ interacts with a simulator having the leakage $\mathcal{L}$, and the simulator simulates the view of $\mathcal{A}$ in our MRIPE scheme. Based on the interaction of $\mathcal{A}$ and the simulator, the ideal world includes four stages, i.e., key generation, token generation stage 1, challenge stage, and token generation stage 2, where the token generation stage and challenge stage are respectively used for simulating the token generation algorithm and encryption algorithm of the MRIPE scheme. The reason why we have two token generation stages is that query users can continuously generate query tokens in our PMRQ scheme. Specifically, the four stages in the ideal world can be formally defined as follows.

• *Key generation:* $\mathcal{A}$ sends a random range $\mathcal{P}$ to the simulator. When the simulator receives $\mathcal{P}$, it randomly chooses a ciphertext $\mathrm{CT}'_{\mathcal{P}}$ for it.

• *Token generation stage 1:* $\mathcal{A}$ sends δ_1 query ranges $\{\mathcal{Q}_i\}_{i=1}^{\delta_1}$ to the simulator. When the simulator receives $\{\mathcal{Q}_i\}_{i=1}^{\delta_1}$, for each $\mathcal{Q}_i$, it uses the leakage $\mathcal{L}$ to randomly generate a query token $\mathrm{TK}'_{\mathcal{Q}_i}$ such that

$$\begin{cases}\mathrm{CT}_{\mathtt{E}_{f(\mathcal{P},\mathcal{Q}_i)}} \bmod \mathtt{L} = 0 & \text{if } \mathcal{L}(\mathcal{P}, \mathcal{Q}_i) = 1\\ \mathrm{CT}_{\mathtt{E}_{f(\mathcal{P},\mathcal{Q}_i)}} \bmod \mathtt{L} \neq 0 & \text{if } \mathcal{L}(\mathcal{P}, \mathcal{Q}_i) = 0.\end{cases}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3158321, IEEE Internet of Things Journal

Finally, it returns these tokens $\{\mathrm{TK}'_{\mathcal{Q}_i}\}_{i=1}^{\delta_1}$ to $\mathcal{A}$.

• *Challenge stage:* The simulator in the challenge stage sends $\mathrm{CT}'_{\mathcal{P}}$ to $\mathcal{A}$.

• *Token generation stage 2:* $\mathcal{A}$ follows the token generation stage 1 to choose $\delta_2 - \delta_1$ query ranges $\{\mathcal{Q}_i\}_{i=\delta_1+1}^{\delta_2}$ and get their query tokens $\{\mathrm{TK}'_{\mathcal{Q}_i}\}_{i=\delta_1+1}^{\delta_2}$ from the simulator.

In the ideal world, $\mathcal{A}$'s views are $\{\mathrm{CT}'_{\mathcal{P}}, \{\mathrm{TK}'_{\mathcal{Q}_i}\}_{i=1}^{\delta_2}\}$. In the real world, $\mathcal{A}$'s views are $\{\mathrm{CT}_{\mathcal{P}}, \{\mathrm{TK}_{\mathcal{Q}_i}\}_{i=1}^{\delta_2}\}$ that are generated by the MRIPE scheme. Based on these views, we formalize the security of the MRIPE scheme.

Definition 1 (Security of MRIPE scheme.): The MRIPE scheme is selectively secure with the leakage $\mathcal{L}$ *iff* for any PPT adversary issuing a polynomial number of query tokens, there exists a simulator such that the probability that the adversary can distinguish the views of real and ideal worlds is negligible.

Theorem 2: MRIPE scheme is selectively secure with $\mathcal{L}$.

Proof. Based on Definition 1, the MRIPE scheme is selectively secure with $\mathcal{L}$ *iff* the probability that $\mathcal{A}$ can distinguish the views of real and ideal worlds is negligible. Since the ciphertexts and tokens in the ideal world are randomly chosen, distinguishing the views of real and ideal worlds is equivalent to distinguishing the ciphertexts and tokens in the real world, i.e., $\{\mathrm{CT}_{\mathcal{P}}, \{\mathrm{TK}_{\mathcal{Q}_i}\}_{i=1}^{\delta_2}\}$, from random ciphertexts and tokens. Next, we show that $\mathcal{A}$ cannot distinguish them.

First, $\mathcal{A}$ cannot separately distinguish $\mathrm{CT}_{\mathcal{P}}$ from random ciphertexts and distinguish $\{\mathrm{TK}_{\mathcal{Q}_i}\}_{i=1}^{\delta_2}$ from random tokens. This is because $\mathrm{CT}_{\mathcal{P}} = \{\{\mathrm{CT}_{\boldsymbol{\gamma}_{k,i}}, \mathrm{CT}_{\boldsymbol{\gamma}'_{k,i}}, \{\mathrm{CT}_{\mathbf{P}_{k,i,j}}, \mathrm{CT}_{\mathbf{P}'_{k,i,j}}\}_{j=1}^{i}\}_{i=1}^{n}\}_{k=1}^{d}$, and each ciphertext in $\mathrm{CT}_{\mathcal{P}}$ contains many random numbers and secret matrices. Then, the unknownness of random numbers and secret matrices can guarantee that $\mathcal{A}$ cannot distinguish $\mathrm{CT}_{\mathcal{P}}$ from random ciphertexts. Similarly, $\mathcal{A}$ also cannot distinguish $\{\mathrm{TK}_{\mathcal{Q}_i}\}_{i=1}^{\delta_2}$ from random tokens. Second, $\mathcal{A}$ may try to distinguish $\{\mathrm{CT}_{\mathcal{P}}, \{\mathrm{TK}_{\mathcal{Q}_i}\}_{i=1}^{\delta_2}\}$ by combining $\mathrm{CT}_{\mathcal{P}}$ and $\mathrm{TK}_{\mathcal{Q}_i}$. This is because when computing $\mathrm{CT}_{\mathtt{E}_{f(\mathcal{P},\mathcal{Q}_i)}}$, the secret matrices can be canceled. That is, $\mathrm{CT}_{\mathtt{E}_{f(\mathcal{P},\mathcal{Q}_i)}} = \sum_{k=1}^{d}\sum_{i=1}^{n}((s_k^{\mathcal{P}} * s_k^{\mathcal{Q}_i} + s_k^{\mathcal{P}'} * s_k^{\mathcal{Q}'_i}) * f_i(p_{k,r}, q_{k,l}) + (t_k^{\mathcal{P}} * t_k^{\mathcal{Q}_i} + t_k^{\mathcal{P}'} * t_k^{\mathcal{Q}'_i}) * (f_i(p_{k,l}, q_{k,r}) - \frac{1}{n}))$. However, even if secret matrices are canceled, the computed $\mathrm{CT}_{\mathtt{E}_{f(\mathcal{P},\mathcal{Q}_i)}}$ still contains random numbers $\{s_k^{\mathcal{P}} * s_k^{\mathcal{Q}_i} + s_k^{\mathcal{P}'} * s_k^{\mathcal{Q}'_i}, t_k^{\mathcal{P}} * t_k^{\mathcal{Q}_i} + t_k^{\mathcal{P}'} * t_k^{\mathcal{Q}'_i}\}$, which ensure that $\mathcal{A}$ cannot distinguish $\mathrm{CT}_{\mathtt{E}_{f(\mathbf{x},\mathcal{Q}_i)}}$ from a random number. Hence, $\mathcal{A}$ cannot distinguish $\{\mathrm{CT}_{\mathcal{P}}, \{\mathrm{TK}_{\mathcal{Q}_i}\}_{i=1}^{\delta_2}\}$ from random ciphertexts and tokens, and the MRIPE scheme is selectively secure with the leakage $\mathcal{L}$. □

B. Security of PMRQ Scheme

We show that the PMRQ scheme can protect the privacy of HC's dataset and doctors' query requests; and the single-dimensional privacy of range queries.

• *HC's dataset and doctors' query requests are privacy-preserving.* First, since the HC's dataset is encrypted by the MRIPE scheme and AES algorithm, the security of the MRIPE scheme and AES algorithm can guarantee that the cloud server cannot obtain the plaintext of the dataset. Second, since each query range $\mathcal{Q}$ in query requests is encrypted by the MRIPE scheme, the security of the MRIPE scheme can prevent the cloud server from knowing the plaintext of the query requests. Therefore, the privacy of HC's dataset and doctors' query requests can be preserved.

• *Single-dimensional privacy of range queries is protected.* In the PMRQ scheme, the range queries are processed through the MRIPE scheme. If the MRIPE scheme can protect the single-dimensional privacy, the PMRQ scheme can protect the single-dimensional privacy. In the MRIPE scheme, determining $\mathcal{P} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ is to compute $\mathrm{CT}_{\mathtt{E}_{f_{k,i}(\mathcal{P},\mathcal{Q})}} = (s_k^{\mathcal{P}} * s_k^{\mathcal{Q}} + s_k^{\mathcal{P}'} * s_k^{\mathcal{Q}'}) * \mathtt{E}_{f_i(p_{k,r},q_{k,l})} + (t_k^{\mathcal{P}} * t_k^{\mathcal{Q}} + t_k^{\mathcal{P}'} * t_k^{\mathcal{Q}'}) * (\mathtt{E}_{f_i(p_{k,l},q_{k,r})} - \frac{1}{n}) + (z_{k,i}^{\mathcal{P}} + z_{k,i}^{\mathcal{P}'} + z_{k,i}^{\mathcal{Q}} + z_{k,i}^{\mathcal{Q}'})$. Since $\{z_{k,i}^{\mathcal{P}}, z_{k,i}^{\mathcal{P}'}, z_{k,i}^{\mathcal{Q}}, z_{k,i}^{\mathcal{Q}'}\}$ are random numbers and can be canceled only when all of them are summed together for $1 \le k \le d$ and $1 \le i \le n$, i.e., $\sum_{k=1}^{d}\sum_{i=1}^{n}(z_{k,i}^{\mathcal{P}} + z_{k,i}^{\mathcal{P}'} + z_{k,i}^{\mathcal{Q}} + z_{k,i}^{\mathcal{Q}'}) = 0$, these random numbers can preserve the single-dimensional privacy of the multi-dimensional range intersection. Therefore, the single-dimensional privacy of range queries can also be preserved.

VII. Performance Evaluation

We evaluate the computational costs of our PMRQ scheme and compare our scheme with other range query schemes.

Experimental setting. We implemented our PMRQ scheme and the compared schemes in Java and conducted experiments on a machine with Intel(R) Xeon(R) CPU E5-2650 v4, 64GB RAM and Ubuntu 16.04 operating system. The evaluation dataset is a real Cardiovascular Disease dataset [24]. We take 10000 records with 5 attributes from this dataset to perform the evaluation. We set the encoding key to be a prime number $\mathtt{L} = 997$ and the length of the access key to be 256 bits.

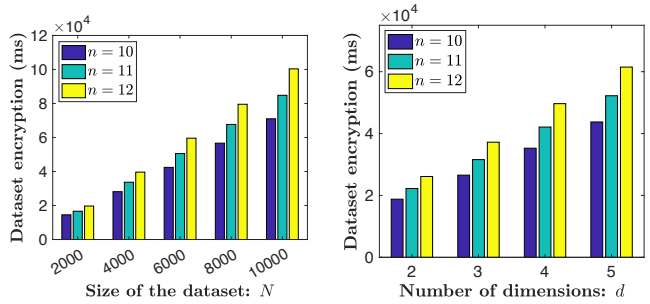
A. Computational Costs of PMRQ Scheme

We theoretically and experimentally analyze the computational costs of our scheme for dataset encryption, query token generation, and query processing.

• *Dataset encryption:* The computational costs of encrypting a dataset are mainly from encrypting the R-tree. Since encrypting a d-dimensional data point $\mathbf{x}$ or a range $\mathcal{P}$ requires $O(d * n^2)$ computational costs and the number of nodes in R-tree is $O(N)$, the computational costs of dataset encryption are $O(N * d * n^2)$, where d is the number of dimensions in the dataset, N is the size of the dataset, and n is the bit length of the dataset's values. In Fig. 2(a) and Fig. 2(b), we depict the experimental results on how the running time of the dataset encryption varies with $\{N, n\}$ and $\{d, n\}$ under the setting of $d = 4$ and $N = 5000$, respectively. These figures demonstrate that the runtime of dataset encryption linearly rises with N and d. Meanwhile, it shows a quadratic increase trend with n. This is because the increase of N will result in a larger R-tree, and the increase of d and n will result in higher computational costs in encrypting the internal nodes and leaf nodes of the R-tree.

• *Token generation:* The computational costs of generating a query token is about $O(d * n^2)$. In Fig. 3(a), we depict the experimental result on how the runtime of token generation varies with d and n. This figure shows that the runtime of

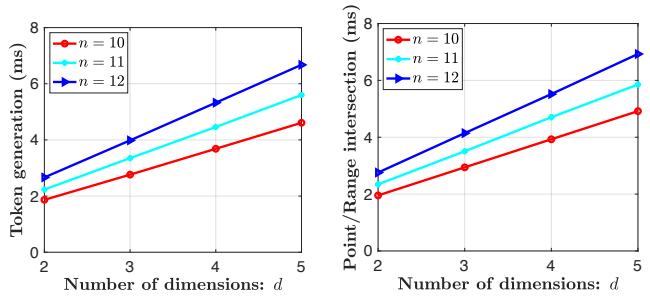
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3158321, IEEE Internet of Things Journal



(a) Dataset encryption with N and n (b) Dataset encryption with d and n

Fig. 2. Runtime of dataset encryption

token generation grows as d and n become larger. This is because the increase of d will result in the growing number of ciphertexts in the query token, and the increase of n will result in the growing size of each ciphertext in the query token.

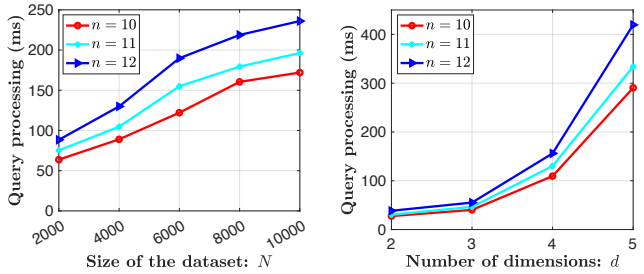


(a) Token generation with d and n (b) Intersection with d and n

Fig. 3. Runtime of token generation and intersection determination

• *Query processing:* The computational costs of query processing are affected by the number of searched nodes in R-tree and the computational costs of searching each node. When the number of records in the query results is $|\mathbb{C}|$, the average number of searched nodes is about $O(|\mathbb{C}| * \log N)$. Meanwhile, the computational costs of searching each node are about $O(d * n^2)$. Hence, the computational costs of range query processing are about $O(|\mathbb{C}| * \log N * d * n^2)$. Next, we present how the runtime of searching each node changes with d and n. Then, we present how the runtime of query processing changes with $\{N, n\}$ and $\{d, n\}$, respectively.

In Fig. 3(b), we plot the runtime of searching each node (i.e., point/range intersection operation) varying with d and n. The figure shows that the runtime has a linear growth trend with d and a quadratic increase trend with n. Meanwhile, the overall runtime is low, e.g., determining an intersection relationship for two 5-dimensional ranges with 10-bit values only takes about 4.91 ms. In Fig. 4(a), we depict the runtime of query processing with N and n under the setting of (i) $d = 4$; and (ii) the size of query results is less than 3. This figure demonstrates the runtime logarithmically rises with N and quadratically grows with n. In Fig. 4(b), we depict the runtime of query processing with d and n under the setting of (i) $N = 5000$; and (ii) the size of query results is less than 8. The figure shows that the runtime rises with d and n. This is because the increase of N will result in a higher R-tree, and the increase of d and n will result in a growing computational costs of evaluating the internal nodes and leaf nodes of the R-tree. In addition, the relationship between the runtime and d is not linear. This is because the increase of d will result in the change of R-tree structure, which makes the runtime of range queries increase.



(a) Query processing with N and n (b) Query processing with d and n

Fig. 4. Runtime of the range query processing

B. Performance Comparison

As shown in TABLE I, we compare our PMRQ scheme with some existing multi-dimensional range query schemes with respect to the search efficiency, query privacy, single-dimensional privacy, and the number of employed cloud servers. This table shows that only our scheme and TRQED+ [13] can achieve multi-dimensional range queries with faster than linear search efficiency, query privacy, and single-dimensional privacy. Meanwhile, Maple [9] is also a representatives among existing schemes because it only leaks the query privacy. Therefore, we compare our scheme with Maple and TRQED+.

TABLE I
COMPARISON AMONG EXISTING SCHEMES

Scheme	Faster than linear search	Query privacy	Single-dimensional privacy	#Cloud servers
Boneh et al's scheme [7]	×	×	√	Single
MRQED [8]	×	√	√	Single
Maple [9]	√	×	√	Single
LSED [10]	√	√	×	Single
Wang et al. scheme [11]	√	√	×	Single
Mei et al. scheme [12]	√	√	×	Single
TRQED$^+$ [13]	√	√	√	Two
Our PMRQ	√	√	√	Single

In our experiment, we implemented Maple and TRQED+ in Java. For the Maple scheme, it was designed based on the bilinear pairing, and we set the security parameter to $\kappa = 512$. For the TRQED+ scheme, except for the single-dimensional privacy, it also employ a flag label to preserve the access pattern privacy, which takes additional computational costs. To be fair, the implemented TRQED+ scheme has removed the flag label. In Fig. 5, we depict the experimental results on how the runtime of range queries processing varies with N under the setting of $d = 4$. Meanwhile, the bit length of values in the dataset is set to 10, i.e., $n = 10$. The average size of query results is less than 3. The figure shows that the computational

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3158321, IEEE Internet of Things Journal

costs of range queries processing in our scheme, Maple, and TRQED+ increase with N. Meanwhile, our scheme and TRQED+ are more efficient than the Maple scheme. Although our scheme is not as efficient as the TRQED+ scheme, it is designed in a single-server model and more practical than the two-server model based TRQED+ scheme.

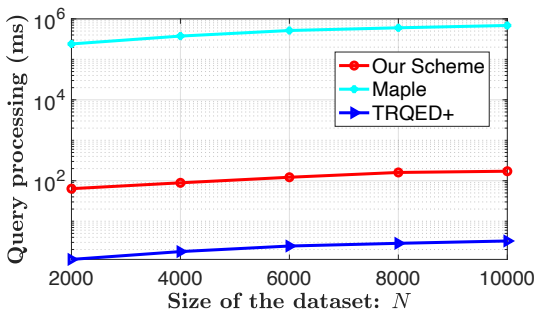


Fig. 5. Comparison of the range query efficiency

VIII. Related Works

Privacy-preserving queries over encrypted data have been widely studied in various areas, e.g., vehicle network [25], [26] and eHealthcare [5]. Since we focus on the multi-dimensional range queries over encrypted data in this work, we will review some existing schemes closely related to our work.

Boneh et al. [7] introduced a bilinear pairing based hidden vector encryption scheme that is applicable to achieve privacy-preserving multi-dimensional range queries. However, it has a linear search efficiency with the dataset's size and cannot preserve the query privacy. To speed up range queries, Shi et al. [8] proposed the MRQED scheme based on an anonymous identity-based encryption (AIBE) scheme [27]. Although the MRQED scheme is more efficient than Boneh et al.'s scheme, it is still inefficient due to the linear search efficiency and computationally expensive public-key based AIBE scheme. To make the query efficiency sublinear to the size of the dataset, Wang et al. [9] proposed the Maple scheme by building an R-tree for the multi-dimensional range queries and applying the hidden vector encryption in [7] to preserve the single-dimensional privacy. Same as Boneh et al.'s scheme, the Maple scheme cannot preserve the query privacy.

Lu et al. [10] presented an LSED scheme by indexing the dataset in each dimension to a B+tree and applying an inner product predicate encryption [28] to preserve the data privacy. Since the proposed scheme separately processes the range queries of each dimension, it inevitably leaks the single-dimensional privacy. Wang et al. [11] indexed the multi-dimensional dataset using an $\widehat{\text{R}}$-tree and preserved the data privacy using an asymmetric scalar-product encryption (ASPE) [29] technique. Similar to LSED [28], the proposed scheme cannot preserve the single-dimensional privacy. Mei et al. [12] designed a multi-dimensional range query scheme based on an interval tree, but this scheme leaks the single-dimensional privacy. Recently, Yang et al. [13] proposed a TRQED+ scheme under the two-server model. With the help of two servers, all dimensions of range queries can be randomly permutated. Thus, the single-dimensional privacy can be preserved. However, TRQED+ was designed based on the two-server model and is secure under the non-collusive assumption between two servers. Since the non-collusive assumption is impractical in some scenarios, especially those with sensitive data, the TRQED+ scheme is impractical. In addition, some schemes [14], [15] were proposed based on the bucketization method, but the query results may contain false positive records. Meanwhile, some order-preserving encryption schemes [30], [31] are applicable to implement multi-dimensional range queries, but they suffer from the ordered chosen plaintext attack [32].

Different from the above works, our PMRQ scheme is designed under a single-server model and can simultaneously preserve the data privacy, query privacy, and single-dimensional privacy.

IX. Conclusion

In this paper, we have proposed an efficient and privacy-preserving multi-dimensional range query scheme under a single-server setting, which can achieve the faster than linear search efficiency, preserve the query privacy, and protect the single-dimensional privacy. First, we designed a data comparison algorithm and a homomorphic encoding technique. Based on them, we designed an encoding-based multi-dimensional range intersection algorithm. Then, we introduced a multi-dimensional intersection predicate encryption (MRIPE) scheme by applying matrix encryption to preserve the privacy of the encoding-based multi-dimensional range intersection algorithm. Finally, based on the MRIPE scheme, we proposed our PMRQ scheme under a single-serve model. In our future works, we will explore other coding and encryption techniques to design more efficient multi-dimensional range query schemes. Meanwhile, we plan to design some efficient and privacy-preserving multi-dimensional range query schemes supporting efficient dynamic updates of the dataset and even protecting the access pattern privacy of the dataset.

Acknowledgements

This research was supported in part by NSERC Discovery Grants (04009), ZJNSF (LZ18F020003), NSFC (61972304), and NSF of Shaanxi Province (2019ZDLGY12-02).

References

[1] K. Hasan, K. Biswas, K. Ahmed, N. S. Nafi, and M. S. Islam, "A comprehensive review of wireless body area network," *J. Netw. Comput. Appl.*, vol. 143, pp. 178–198, 2019.

[2] Y. Zheng, R. Lu, and J. Shao, "Achieving efficient and privacy-preserving k-nn query for outsourced ehealthcare data," *J. Medical Syst.*, vol. 43, no. 5, pp. 123:1–123:13, 2019.

[3] F. Wang, H. Zhu, R. Lu, Y. Zheng, and H. Li, "A privacy-preserving and non-interactive federated learning scheme for regression training with gradient descent," *Inf. Sci.*, vol. 552, pp. 183–200.

[4] Y. Guan, R. Lu, Y. Zheng, J. Shao, and G. Wei, "Toward oblivious location-based k-nearest neighbor query in smart cities," *IEEE Internet Things J.*, pp. 1–1, 2021, doi=10.1109/JIOT.2021.3068859.

[5] S. Zhang, S. Ray, R. Lu, Y. Zheng, Y. Guan, and J. Shao, "Achieving efficient and privacy-preserving dynamic skyline query in online medical diagnosis," *IEEE Internet Things J.*, pp. 1–1, 2021, doi=10.1109/JIOT.2021.3117933.

[6] X. Wang, J. Ma, Y. Miao, R. Yang, and Y. Chang, "EPSMD: an efficient privacy-preserving sensor data monitoring and online diagnosis system," in *2018 IEEE Conference on Computer Communications, INFOCOM 2018, Honolulu, HI, USA, April 16-19, 2018*, 2018, pp. 819–827.

[7] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007*, ser. Lecture Notes in Computer Science, vol. 4392, 2007, pp. 535–554.

[8] E. Shi, J. Bethencourt, T. H. Chan, D. X. Song, and A. Perrig, "Multi-dimensional range query over encrypted data," in *2007 IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA*, 2007, pp. 350–364.

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

[10] Y. Lu, "Privacy-preserving logarithmic-time search on encrypted data in cloud," in *19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, February 5-8, 2012*, 2012.

[11] P. Wang and C. V. Ravishankar, "Secure and efficient range queries on outsourced databases using rp-trees," in *29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013*, 2013, pp. 314–325.

[12] Z. Mei, H. Zhu, Z. Cui, Z. Wu, G. Peng, B. Wu, and C. Zhang, "Executing multi-dimensional range query efficiently and flexibly over outsourced ciphertexts in the cloud," *Inf. Sci.*, vol. 432, pp. 79–96, 2018.

[13] W. Yang, Y. Geng, L. Li, X. Xie, and L. Huang, "Achieving secure and dynamic range queries over encrypted cloud data," *IEEE Trans. Knowl. Data Eng.*, 2020.

[14] Y. Lee, "Secure ordered bucketization," *IEEE Trans. Dependable Secur. Comput.*, vol. 11, no. 3, pp. 292–303, 2014.

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

[16] Y. Zheng, R. Lu, Y. Guan, J. Shao, and H. Zhu, "Towards practical and privacy-preserving multi-dimensional range query over cloud," *IEEE Trans. Dependable Secur. Comput.*, pp. 1–1, 2021, doi=10.1109/TDSC.2021.3101120.

[17] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *IACR Cryptol. ePrint Arch.*, vol. 2012, p. 144, 2012.

[18] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Comput. Theory*, vol. 6, no. 3, pp. 13:1–13:36, 2014.

[19] Y. Zheng, R. Lu, and M. S. I. Mamun, "Privacy-preserving computation offloading for time-series activities classification in ehealthcare," in *2020 IEEE International Conference on Communications, ICC 2020, Dublin, Ireland, June 7-11, 2020*, 2020, pp. 1–6.

[20] Y. Zheng, R. Lu, S. Zhang, Y. Guan, J. Shao, and H. Zhu, "Toward privacy-preserving healthcare monitoring based on time-series activities over cloud," *IEEE Internet Things J.*, pp. 1–1, 2021, doi=10.1109/JIOT.2021.3079106.

[21] A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, USA, June 18-21, 1984*, 1984, pp. 47–57.

[22] D. R. Stinson, *Cryptography - theory and practice*, ser. Discrete mathematics and its applications series. CRC Press, 1995.

[23] Y. Zheng, R. Lu, Y. Guan, J. Shao, and H. Zhu, "Efficient privacy-preserving similarity range query with quadsector tree in ehealthcare," *IEEE Trans. Serv. Comput.*, pp. 1–1, 2021, doi=10.1109/TSC.2021.3081350.

[24] "Cardiovascular Disease Dataset," [Online], Available: https://www.kaggle.com/sulianova/cardiovascular-disease-dataset.

[25] Q. Kong, R. Lu, M. Ma, and H. Bao, "A privacy-preserving and verifiable querying scheme in vehicular fog data dissemination," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1877–1887, 2019.

[26] Q. Kong, L. Su, and M. Ma, "Achieving privacy-preserving and verifiable data sharing in vehicular fog with blockchain," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 4889–4898, 2021.

[27] X. Boyen and B. Waters, "Anonymous hierarchical identity-based encryption (without random oracles)," in *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings*, ser. Lecture Notes in Computer Science, vol. 4117, 2006, pp. 290–307.

[28] E. Shen, E. Shi, and B. Waters, "Predicate privacy in encryption systems," in *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, ser. Lecture Notes in Computer Science, vol. 5444, 2009, pp. 457–473.

[29] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 - July 2, 2009*, 2009, pp. 139–152.

[30] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, ser. Lecture Notes in Computer Science, vol. 5479, 2009, pp. 224–241.

[31] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: Improved security analysis and alternative solutions," in *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, ser. Lecture Notes in Computer Science, vol. 6841, 2011, pp. 578–595.

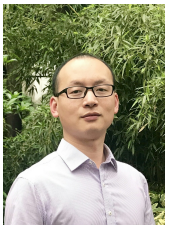
[32] L. Xiao and I. Yen, "A note for the ideal order-preserving encryption object and generalized order-preserving encryption," *IACR Cryptol. ePrint Arch.*, vol. 2012, p. 350, 2012.

Yandong Zheng received her M.S. degree from the Department of Computer Science, Beihang University, China, in 2017 and she is currently pursuing her Ph.D. degree in the Faculty of Computer Science, University of New Brunswick, Canada. Her research interest includes cloud computing security, big data privacy and applied privacy.

Rongxing Lu (S'09-M'11-SM'15-F'21) is a University Research Scholar, an associate professor at the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada. Before that, he worked as an assistant professor at the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore from April 2013 to August 2016. Rongxing Lu worked as a Postdoctoral Fellow at the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious "Governor General's Gold Medal", when he received his PhD degree from the Department of Electrical & Computer Engineering, University of Waterloo, Canada, in 2012; and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. Dr. Lu is an IEEE Fellow. His research interests include applied cryptography, privacy enhancing technologies, and IoT-Big Data security and privacy. He has published extensively in his areas of expertise (with H-index 78 from Google Scholar as of Jan 2022), and was the recipient of 9 best (student) paper awards from some reputable journals and conferences. Currently, Dr. Lu serves as the Chair of IEEE ComSoc CIS-TC (Communications and Information Security Technical Committee), and the founding Co-chair of IEEE TEMS Blockchain and Distributed Ledgers Technologies Technical Committee (BDLT-TC). Dr. Lu is the Winner of 2016-17 Excellence in Teaching Award, FCS, UNB.

Songnian Zhang received his M.S. degree from Xidian University, China, in 2016 and he is currently pursuing his Ph.D. degree in the Faculty of Computer Science, University of New Brunswick, Canada. His research interest includes cloud computing security, big data query and query privacy.

Yunguo Guan is a PhD student of the Faculty of Computer Science, University of New Brunswick, Canada. His research interests include applied cryptography and game theory.



Jun Shao (M'21-SM'22) received his Ph.D. degree from the Department of Computer and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2008.

He was a Post-Doctoral Fellow with the School of Information Sciences and Technology, Pennsylvania State University, Pennsylvania, PA, USA, from 2008 to 2010. He is currently a Professor with the School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, China. His current research interests include network security and applied cryptography.

Fengwei Wang (M'21) received his B.Sc. degree from Xidian University in 2016 and Ph.D. degree from Xidian University in 2021. In 2019, he was with the Faculty of Computer Science, University of New Brunswick as a visiting scholar.

Since 2021, he has been the lecturer with the School of Cyber Engineering, Xidian University, Xi'an, China. His research interests include the areas of applied cryptography, cyber security, and privacy.

Hui Zhu (M'13-SM'19) received the B.Sc. degree from Xidian University, Xian, China, in 2003, the M.Sc. degree from Wuhan University, Wuhan, China, in 2005, and the Ph.D. degree from Xidian University, in 2009.

He was a Research Fellow with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore, in 2013. Since 2016, he has been a Professor with the School of Cyber Engineering, Xidian University. His current research interests include applied cryptography, data security, and privacy.