Achieving Efficient and Privacy-Preserving Dynamic Skyline Query in Online Medical Diagnosis

Songnian Zhang, Suprio Ray, *Member, IEEE*, Rongxing Lu, *Fellow, IEEE*, Yandong Zheng, Yunguo Guan, and Jun Shao, *Member, IEEE*

***Abstract*—Wireless body area network (WBAN) and big data techniques indubitably enable the online medical diagnosis system to be more practical. In the system, to make a more accurate diagnosis, doctors wish to obtain some archived medical data records, which are similar to the sensed patient data, to learn from the prior diagnoses. As a practically useful similarity search, the dynamic skyline query can provide doctors with similar data records having all possible relative weights of attributes. Driven by the powerful cloud, the data owner often outsources encrypted data and the corresponding services, e.g., dynamic skyline query services here, to a third-party cloud. As a result, it is required to perform the dynamic skyline query over encrypted data. However, existing schemes are either insecure or inefficient. To address the issue, in this article, we propose an efficient and privacy-preserving dynamic skyline query scheme and use it in an online medical diagnosis system. Specifically, based on symmetric homomorphic encryption (SHE), we present a set of efficient and secure protocols to achieve various operations, such as less than comparison, equality test, and dominance determination, without leaking any sensitive information to the cloud. With these secure protocols, we carefully design our dynamic skyline query scheme to attain full security and high efficiency at the same time. Detailed security analysis shows that our proposed scheme is indeed privacy-preserving. With extensive experimental evaluations, we show that our proposed scheme outperforms the alternative scheme by two orders of magnitude in the computational cost and at least 8.1× in the communication cost.**

***Index Terms*—Access pattern, dynamic skyline, privacy preservation, skyline query, symmetric homomorphic encryption (SHE).**

I. Introduction

As a practical application area of Internet of Things (IoT), the wireless body area network (WBAN) has been widely studied in the remote/online medical diagnosis system [1]–[3], in which patients' physiological data are collected by biosensors, and doctors can remotely make a medical diagnosis according to the sensed data. Meanwhile, the patient's physiological data and the corresponding diagnosis results can be archived, and other doctors can make full use of the archived data to obtain valuable experiences by searching physiological data records that are similar to the sensed physiological data. To have effective and all-sided reference information, it is desirable for doctors to retrieve similar physiological data records considering all possible relative weights of the attributes, e.g., considering one attribute or an arbitrary combination of attributes [4], which can be achieved by performing dynamic skyline queries [5]. That is because, with a query point (sensed physiological data of a patient), the dynamic skyline query can find a set of similar physiological data records (already had diagnosis results), which are close to the given query point. With these retrieved data records and diagnosis results, doctors can make a more comprehensive medical analysis for the patient and thus, have a more accurate diagnosis. Consequently, the online medical diagnosis system needs to respond to dynamic skyline queries of doctors such that doctors can search Pareto-similar physiological data records from the archived data set. See the formal definition and a detailed example of the dynamic skyline query in Section III-A.

In a real-world scenario, a single hospital generally does not have sufficient medical data to provide valuable references for doctors. Therefore, to mine more valuable information, it is beneficial for hospitals to share their medical data in a third-party cloud and enable the cloud to provide the query services [6]. However, it suffers from privacy issues, i.e., patients' physiological data are sensitive, while the cloud is not fully trusted. A promising solution is to encrypt these outsourced data and perform queries over encrypted data [7]–[10]. Although, various schemes [2], [6], [11], [12] were proposed to deal with skyline queries over encrypted data, they did not consider the dynamic skyline query, which is the target query in our paper, and cannot be directly used to tackle such a query. The reason is that before computing skyline, the dynamic skyline query needs to calculate the absolute values between the data records and the given query point, which results in a gap between the above schemes and the secure dynamic skyline query. Recently, the works in [4] and [13] designed schemes to achieve the secure dynamic skyline query over cloud. Unfortunately, the work in [13] employed the order revealing encryption (ORE) to encrypt outsourced data. It leaks the order relations on each dimension and cannot hide access patterns, which may incur inference attacks [14], [15]. Although Liu *et al.* [4] designed a fully secure dynamic skyline query

Manuscript received July 1, 2021; revised September 9, 2021; accepted October 2, 2021. Date of publication October 5, 2021; date of current version June 7, 2022. This work was supported in part by NSERC Discovery Grants under Grant 04009; in part by ZJNSF under Grant LZ18F020003; and in part by NSFC under Grant U1709217. *(Corresponding author: Rongxing Lu.)*

Songnian Zhang, Suprio Ray, Rongxing Lu, Yandong Zheng, and Yunguo Guan are with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: szhang17@unb.ca; sray@unb.ca; rlu1@unb.ca; yzheng8@unb.ca; yguan4@unb.ca).

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

Digital Object Identifier 10.1109/JIOT.2021.3117933

scheme, i.e., preserving the privacy of outsourced data, query requests, query results, and access patterns, it has performance issues due to the inefficient basic protocols. Therefore, it is still challenging to ensure both the security and efficiency of the dynamic skyline query scheme at the same time.

In this article, we aim to propose a privacy-preserving dynamic skyline scheme that can be used in the online medical diagnosis system. Our proposed scheme has the same security level as that in [4] while ensuring efficiency. To achieve it, we first adopt an efficient symmetric homomorphic encryption (SHE) [10] as our primitive cryptosystem. Based on the characteristics of SHE, we propose several secure and efficient protocols to attain necessary operations in the dynamic skyline query. First, we design two atomic protocols to securely execute basic operations: 1) comparison and 2) equality test, respectively. Then, we devise two secure protocols to accomplish more complex operations, i.e., finding minimum value among n data records and determining dynamic dominance relation of two data records. All of these protocols are achieved in a two-server model and are carefully designed to be as efficient as possible without compromising security. Nevertheless, we also note that since SHE is a leveled fully homomorphic encryption, i.e., it supports limited homomorphic multiplication operations, it has to use a bootstrapping protocol [10] to refresh ciphertexts such that more homomorphic multiplication operations can be executed. Unfortunately, the bootstrapping protocol inevitably increases computational and communication costs. Accordingly, we would like to enable our scheme to not use bootstrapping. However, since the complex scheme usually requires a deep homomorphic multiplication chain, it is challenging to devise SHE-based complex schemes without bootstrapping. To tackle it, we elaborately design our scheme so that it uses as little multiplication as possible. Specifically, the main contributions of this article are threefold as follows.

1) We propose an efficient and privacy-preserving dynamic skyline query scheme that can be used to achieve a secure online medical diagnosis system. On the one hand, from the perspective of security, our proposed scheme can guarantee the privacy of outsourced data, query data, query results, and access patterns. On the other hand, from the perspective of performance, we ensure the number of homomorphic multiplication operations in our scheme without increasing with the scheme execution, making it possible for the maximum multiplication depth of SHE to cover the number of homomorphic multiplication operations. Thus, our proposed scheme can avoid using bootstrapping for the static data set.
2) We carefully devise a set of secure protocols based on SHE to serve as building blocks, which enriches the secure computing protocols under a two-server model. In particular, we first design two secure atomic protocols by using a flip-coin mechanism, which prevents any protocols' inputs and outputs related information from leaking. Then, based on these secure atomic protocols, we present an efficient and secure dynamic dominance (SDD) protocol to determine the dynamic dominance relations. In addition, we devise a secure minimum of n data records (SMINn) protocol, in which a privacy-preserving XNOR gate and a private minimum tree (PM-tree) are proposed to ensure both security and efficiency.
3) Finally, we analyze the security of our proposed scheme and conduct extensive experiments to evaluate its performance. The results show that our proposed scheme is much more efficient than the alternative scheme that has the same security level as ours. Specifically, compared with the alternative scheme, our proposed scheme is two orders of magnitude better in the computation cost and also at least 8.1× better in the communication cost.

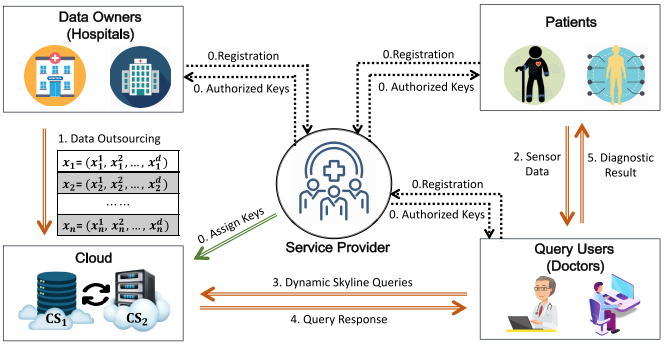


Fig. 1. System model under consideration.

The remainder of this article is organized as follows. In Section II, we introduce our system model, security model, and design goal. Then, we review our preliminaries in Section III. After that, we present our proposed scheme in Section IV, followed by security analysis and performance evaluation in Sections V and VI, respectively. Finally, we discuss some related works in Section VII and draw our conclusion in Section VIII.

II. Models and Design Goal

In this section, we formalize our system model, security model, and identify our design goal.

A. System Model

In our system model, we consider a typical cloud-based online medical diagnosis system with dynamic skyline queries, which mainly consists of five types of entities: 1) a service provider $\mathcal{SP}$; 2) a set of data owners (hospitals) $\mathcal{O} = \{o_1, o_2, \ldots\}$; 3) a cloud $\mathcal{C}$ with two servers $\{\mathcal{CS}_1, \mathcal{CS}_2\}$; 4) multiple query users (doctors) $\mathcal{U}$=$\{u_1, u_2, \ldots\}$; and 5) patients $\mathcal{P}$=$\{p_1, p_2, \ldots\}$. Our system model is shown in Fig. 1.

Service Provider $\mathcal{SP}$: In our system model, $\mathcal{SP}$ is the service organizer and provider, e.g., a health center, who is responsible for initializing the entire system. $\mathcal{SP}$ invites data owners $\mathcal{O} = \{o_1, o_2, \ldots\}$ to share their medical data and provides dynamic skyline query services to query users. However, since $\mathcal{SP}$ may not be powerful in storage and computing, he/she tends to outsource the shared medical data and dynamic

skyline query services to a cloud. Meanwhile, to ensure privacy, $\mathcal{SP}$ provides registration servers to other entities and authorizes proper keys to different entities.

Data Owners $\mathcal{O}$: We consider hospitals as data owners, who hold medical data sets but may not have sufficient data to perform data mining, which drives them to merge their data together. Generally, a medical data record here includes two parts: 1) the physiological data collected from patients and 2) the dignosis results from doctors. For ease of description, in this article, we only show patients' physiological data and ignore the diagnostic results. We represent each data record in the medical data set as a d-dimensional vector, i.e., $\mathcal{X} = \{\mathtt{x}_i = (\mathtt{x}_i^1, \mathtt{x}_i^2, \ldots, \mathtt{x}_i^d) \mid 1 \leq i \leq n\}$, where n is the number of data records in the data set, and assume that all dimensions are integers [16]. Since $\mathcal{SP}$ would outsource the medical data to a cloud, it is equivalent to the data owners uploading their data to the cloud, as shown in Fig. 1.

Cloud $\mathcal{C}$: The cloud contains two servers $\{\mathcal{CS}_1, \mathcal{CS}_2\}$, which are considered as powerful in both storage and computing. $\mathcal{CS}_1$ receives the outsourced data from data owners $\mathcal{O}$, while $\mathcal{CS}_2$ is authorized with the private key. They will cooperatively offer reliable dynamic skyline query services to query users.

Query Users $\mathcal{U} = \{u_1, u_2, \ldots\}$: In the system, the query users are doctors. First, query users obtain the physiological data sensed from patients, which actually is the query request and is denoted as $\mathtt{q} = (\mathtt{q}^1, \mathtt{q}^2, \ldots, \mathtt{q}^d)$. Then, they launch the dynamic skyline query with $\mathtt{q}$ and obtain all similar data records in $\mathcal{X}$ without assigning weights for each attribute. After that, query users can have a comprehensive medical analysis based on the retrieved data records and return the final diagnosis results to patients. Before participating in online medical diagnosis, the query users must be authorized by $\mathcal{SP}$ with the authorized keys. That is, only the authorized query users can obtain the sensed data from patients and receive query responses from the cloud $\mathcal{C}$.

Patients $\mathcal{P} = \{p_1, p_2, \ldots\}$: In our system, patients can enjoy the online medical diagnosis services by transferring their physiological data $\mathtt{q}$ to the designated doctors. Each patient is equipped with specific medical sensors (IoT devices) for collecting physiological data. Before enjoying such online medical services, patients are required to register to $\mathcal{SP}$.

B. Security Model

In our security model, since the service provider $\mathcal{SP}$ initializes the whole system, and the registered data owners $\mathcal{O}$ hold the medical data of patients, they are considered to be fully trusted. For patients and query users (doctors), we consider the authorized ones to be honest. The patients $\mathcal{P}$ will honestly send their sensed physiological data to doctors. Meanwhile, the query users will sincerely follow the protocol to issue the dynamic skyline queries and return diagnosis results to patients. However, the cloud servers $\{\mathcal{CS}_1, \mathcal{CS}_2\}$ are considered to be honest-but-curious. That is, they will honestly follow the underlying scheme but may be curious to learn some private information. To ensure privacy, the data owners will encrypt their medical data sets before outsourcing them to the cloud $\mathcal{C}$. Therefore, in our model, the cloud stores the encrypted data sets and provides the dynamic skyline query services over these encrypted data. Since the cloud servers are not fully trusted, they may attempt to obtain private information, including the plaintexts of the encrypted data sets, query requests, and query results, based on the stored data sets and the process of dynamic skyline queries. We assume that there is no collusion between $\mathcal{CS}_1$ and $\mathcal{CS}_2$, as well as no collusion between the cloud and other entities. It is reasonable since the different cloud servers may have conflicts of interest, and they wish to maintain their reputations. Note that there may be other active attacks, e.g., Denial-of-Service (DoS) attacks, to the system. Since we focus on privacy preservation, those attacks are beyond the scope of this article, and will be discussed in our future work.

C. Design Goal

In this article, we aim to design a privacy-preserving and efficient dynamic skyline query scheme under our system model and security model. In particular, the following objectives should be attained.

1) *Privacy Preservation:* The basic requirement of our proposed scheme is privacy preservation. First, the outsourced data sets should be kept secret from the cloud servers. Second, the query requests and query results should be kept secret from the cloud servers. Besides, our scheme should hide access patterns.
2) *Efficiency:* In order to preserve privacy, it is unavoidable to incur additional costs. As a result, we also aim to minimize the computational and communication costs of querying dynamic skyline over encrypted data.

III. Preliminaries

In this section, we first define the problem of the dynamic skyline query. Then, we introduce the symmetric homomorphic encryption (SHE), which will be used as the cryptographic primitive in our proposed scheme. After that, we introduce two SHE-based atomic protocols that serve as the building blocks of our proposed scheme.

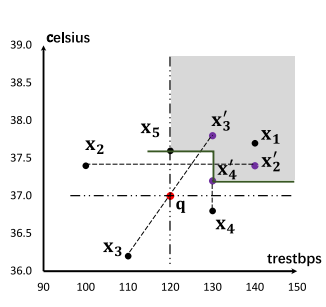
A. Dynamic Skyline Query

The dynamic skyline query was first introduced in [5] and is used to find the skyline points with regard to a query point. Here, we formally define the dynamic skyline query as follows.

Definition 1 (Dynamic Dominance): Given two d-dimensional points $\mathtt{p}, \mathtt{r} \in \mathcal{X}$ and a query point $\mathtt{q}$ in the workplace, $\mathtt{p}$ is said to be dynamically dominating $\mathtt{r}$, denoted as $\mathtt{p} \prec \mathtt{r}$, if $\exists j \in [1, d]$ such that $|\mathtt{p}^j - \mathtt{q}^j| < |\mathtt{r}^j - \mathtt{q}^j|$ and $\forall i \in [1, d], i \neq j$ that $|\mathtt{p}^i - \mathtt{q}^i| \leq |\mathtt{r}^i - \mathtt{q}^i|$.

Definition 2 (Dynamic Skyline Query): Given a data set $\mathcal{X}$ and a query point $\mathtt{q}$ in the workplace, the dynamic skyline query returns a data set $SK \subseteq \mathcal{X}$, in which the points are not dynamically dominated by any other point. That is, $SK = \{\mathtt{x}_i \in \mathcal{X} \mid \nexists \mathtt{r} \in \mathcal{X}$ such that $\mathtt{r} \prec \mathtt{x}_i\}$.

The dynamic skyline query is suitable to find the points that are Pareto-similar to the user-interested point. To illustrate the dynamic skyline query, we take a 2-D data set as an example, which contains the common physiological attributes:



Record	trestbps	celsius
x_1	140	37.7
x_2	100	37.4
x_3	110	36.2
x_4	130	36.8
x_5	120	37.6

$q = (120, 37.0)$

Record	Absolute values	
x_1	20	0.7
x_2	20	0.4
x_3	10	0.8
x_4	10	0.2
x_5	0	0.6

Fig. 2. Example of the dynamic skyline query.

blood pressure (trestbps) and body temperature (celsius), as shown in Fig. 2. There are five data records in the example: $\mathcal{X} = \{x_1, x_2, \ldots, x_5\}$. If the sensed physiological data are $q = (120, 37.0)$, the dynamic skyline query will return $SK = \{x_4, x_5\}$. We can see that finding dynamic skyline points is equivalent to compute the traditional skyline that mapped all points in a new data space where point q is the origin, and the absolute distances to q are used as new values. For example, $x_4 = (130, 36.8)$ is mapped into $(130, 37.2)$, and $(10, 0.2)$ will be used to compute skyline points. Note that in our proposed scheme, the body temperature values will be scaled into integers before outsourcing.

B. Symmetric Homomorphic Encryption

SHE is an efficient homomorphic encryption that uses the same key in both encryption and decryption. It was first proposed in [17] and proved to be CPA secure in [10]. Typically, SHE includes three algorithms, namely: 1) key generation $\mathsf{KeyGen}()$; 2) encryption $\mathsf{Enc}()$; and 3) decryption $\mathsf{Dec}()$, as follows.

1) $\mathsf{KeyGen}(k_0, k_1, k_2)$*:* On input security parameters $\{k_0, k_1, k_2\}$ satisfying $k_1 \ll k_2 < k_0$, the key generation algorithm first chooses two large prime numbers p and q with $|p| = |q| = k_0$ and generates the secret key $sk = (p, \mathcal{L})$, where $\mathcal{L}$ is a random number with $|\mathcal{L}| = k_2$. Then, the algorithm computes $\mathcal{N} = pq$ and sets the public parameter $pp = (k_0, k_1, k_2, \mathcal{N})$. Finally, the basic message space $\mathcal{M}$ is set $[-2^{k_1-1}, 2^{k_1-1})$.
2) $\mathsf{Enc}(sk, m)$*:* Taking a secret key sk and a message $m \in \mathcal{M}$ as inputs, the encryption algorithm outputs the ciphertext $\mathsf{E}(m) = (r\mathcal{L} + m)(1 + r'p) \mod \mathcal{N}$, where $r \in \{0, 1\}^{k_2}$ and $r' \in \{0, 1\}^{k_0}$ are random numbers.
3) $\mathsf{Dec}(sk, \mathsf{E}(m))$*:* On input a secret key sk and a ciphertext $\mathsf{E}(m)$, the decryption algorithm recovers a message $m' = (\mathsf{E}(m) \mod p) \mod \mathcal{L} = (m + r\mathcal{L}) \mod \mathcal{L}$. If $m' < (\mathcal{L}/2)$, it indicates $m \geq 0$ and $m = m'$. Otherwise, $m < 0$ and $m = m' - \mathcal{L}$.

SHE enjoys the following homomorphic properties.

1) *Homomorphic Addition-I:* $\mathsf{E}(m_1) + \mathsf{E}(m_2) \mod \mathcal{N} \rightarrow \mathsf{E}(m_1 + m_2)$.
2) *Homomorphic Multiplication-I:* $\mathsf{E}(m_1) \cdot \mathsf{E}(m_2) \mod \mathcal{N} \rightarrow \mathsf{E}(m_1 \cdot m_2)$.
3) *Homomorphic Addition-II:* $\mathsf{E}(m_1) + m_2 \mod \mathcal{N} \rightarrow \mathsf{E}(m_1 + m_2)$.
4) *Homomorphic Multiplication-II:* $\mathsf{E}(m_1) \cdot m_2 \mod \mathcal{N} \rightarrow \mathsf{E}(m_1 \cdot m_2)$ when $m_2 > 0$.

Although SHE can support almost an unlimited number of homomorphic operations of Homomorphic addition-I, Homomorphic addition-II, and Homomorphic multiplication-II, it has the limited number of operation of Homomorphic multiplication-I. Therefore, SHE is also a leveled fully homomorphic encryption.

Maximum Multiplicative Depth: After conducting one Homomorphic multiplication-I operation, the newly generated ciphertext has the largest noise term $r_1 r_2 \mathcal{L}^2$. To ensure the decryption correctness, we should guarantee that $r_1 r_2 \mathcal{L}^2 < p$. Recalling the $\mathsf{KeyGen}()$ and $\mathsf{Enc}()$ algorithms, we have $|p| = k_0$, $|\mathcal{L}| = k_2$, and $|r_1| = |r_2| = k_2$. Therefore, if we set the maximum multiplicative depth as σ, we have $2(\sigma + 1)k_2 < k_0$, i.e., $\sigma = \lfloor (k_0/2k_2) - 1 \rfloor$.

To make SHE support an infinite number of homomorphic multiplication-I, Zheng *et al.* [10] proposed a bootstrapping protocol to remove the largest noise term $r_1 r_2 \mathcal{L}^2$. However, this protocol incurs additional computational and communication costs. To avoid using the bootstrapping protocol, we carefully devise our proposed scheme and limit the number of the homomorphic multiplication-I operation to be less than σ.

Encryption With Public Key: SHE can encrypt messages with $\{\mathsf{E}(0)_1, \mathsf{E}(0)_2\}$ by using the homomorphic properties, where $\mathsf{E}(0)_1$ and $\mathsf{E}(0)_2$ denote the ciphertext of 0 with different random numbers. If we treat these ciphertexts as a public key pk, we have the other encryption algorithm of SHE $\mathsf{Enc}_{\mathsf{pk}}(pk, m)$, which can output ciphertext $\mathsf{E}(m)$ as follows:

$$\mathsf{E}(m) = m + r_1 \cdot \mathsf{E}(0)_1 + r_2 \cdot \mathsf{E}(0)_2 \mod \mathcal{N} \tag{1}$$

where r_1 and $r_2 \in \{0, 1\}^{k_2}$ are two random numbers. In some scenarios, one entity may only need to encrypt messages. If we authorize sk to the entity, it may lower the security of the whole system. In such a case, we can transfer $pk = \{\mathsf{E}(0)_1, \mathsf{E}(0)_2\}$, instead of sk, to the entity who can encrypt messages with (1). Note that this approach has been proven to be IND-CPA secure in [18].

C. SHE-Based Atomic Protocols

Based on the nice homomorphic properties of SHE, we design two secure atomic protocols: 1) secure less than (SLESS) and 2) secure equal (SEQ) protocols to determine less than and equality over encrypted data, respectively. We achieve these protocols in a two-server model, where $\mathcal{CS}_1$ has $\{\mathsf{E}(m_1), \mathsf{E}(m_2), pk\}$, and $\mathcal{CS}_2$ holds the secret key sk.

Secure Less Than Protocol: The goal of the protocol is to determine whether $m_1 < m_2$ without leaking any m_1 and m_2 related information to $\mathcal{CS}_1$ or $\mathcal{CS}_2$. If $m_1 < m_2$, the protocol outputs $\mathsf{E}(1)$, otherwise, $\mathsf{E}(0)$.

Step 1: $\mathcal{CS}_1$ flips a coin $s \in \{-1, 1\}$ and chooses two random numbers $r_1, r_2 \in \{0, 1\}^{k_1}$ satisfying $r_1 > r_2 > 0$. Then, $\mathcal{CS}_1$ computes

$$\begin{aligned}\mathsf{E}(\alpha) &= \mathsf{E}(s \cdot r_1) \cdot (\mathsf{E}(m_1) + \mathsf{E}(m_2) \cdot \mathsf{E}(-1)) + \mathsf{E}(s \cdot r_2) \\ &= \mathsf{E}(s \cdot r_1 \cdot (m_1 - m_2) + s \cdot r_2).\end{aligned}$$

Afterward, $\mathcal{CS}_1$ sends $\mathsf{E}(\alpha)$ to $\mathcal{CS}_2$.

Step 2: On receiving $\mathsf{E}(\alpha)$, $\mathcal{CS}_2$ first uses sk to recover α. Then, $\mathcal{CS}_2$ checks whether $\alpha < 0$. If yes, $\mathcal{CS}_2$

makes $\theta = 1$ and encrypts it into $\mathsf{E}(\theta)$. Otherwise, $\mathcal{CS}_2$ generates $\mathsf{E}(\theta) = \mathsf{E}(0)$. Next, $\mathcal{CS}_2$ returns $\mathsf{E}(\theta)$ to $\mathcal{CS}_1$.

Step 3: If $s = 1$, $\mathcal{CS}_1$ lets $\mathsf{E}(\delta) = \mathsf{E}(\theta)$. If $s = -1$, $\mathcal{CS}_1$ computes $\mathsf{E}(\delta) = \mathsf{E}(1) + \mathsf{E}(\theta) \cdot \mathsf{E}(-1) = \mathsf{E}(1-\theta)$. Finally, $\mathsf{E}(\delta)$ is the output of our SLESS protocol.

Correctness: When $s = 1$, we have $\mathsf{E}(\alpha) = \mathsf{E}(r_1 \cdot (m_1 - m_2) + r_2)$. If $m_1 < m_2$, $\alpha < 0$. As a result, $\mathsf{E}(\delta) = \mathsf{E}(\theta) = \mathsf{E}(1)$. If $m_1 \geq m_2$, $\alpha > 0$. In this case, $\mathsf{E}(\delta) = \mathsf{E}(\theta) = \mathsf{E}(0)$. Therefore, when $s = 1$, *iff* $m_1 < m_2$, the output of SLESS protocol is $\mathsf{E}(\delta) = \mathsf{E}(1)$. When $s = -1$, we have $\mathsf{E}(\alpha) = \mathsf{E}(r_1 \cdot (m_2 - m_1) - r_2)$. If $m_1 < m_2$, $\alpha > 0$. As a result, $\mathsf{E}(\delta) = \mathsf{E}(1-\theta) = \mathsf{E}(1-0) = \mathsf{E}(1)$. If $m_1 \geq m_2$, $\alpha < 0$. In this case, $\mathsf{E}(\delta) = \mathsf{E}(1-\theta) = \mathsf{E}(1-1) = \mathsf{E}(0)$. Therefore, when $s = -1$, *iff* $m_1 < m_2$, the output of SLESS protocol is $\mathsf{E}(\delta) = \mathsf{E}(1)$. Thus, our SLESS protocol is correct.

Secure Equal Protocol: This protocol is used to determine whether $m_1 \stackrel{?}{=} m_2$ without leaking any m_1 and m_2 related information to $\mathcal{CS}_1$ or $\mathcal{CS}_2$. If $m_1 = m_2$, the protocol outputs $\mathsf{E}(1)$, otherwise, $\mathsf{E}(0)$.

Step 1: $\mathcal{CS}_1$ flips a coin $s \in \{-1, 1\}$ and chooses two random numbers $r_1, r_2 \in \{0,1\}^{k_1}$ satisfying $r_1 > r_2 > 0$. Then, $\mathcal{CS}_1$ computes $\mathsf{E}(\alpha) = \mathsf{E}(s \cdot r_1 \cdot (m_1 - m_2)^2 - s \cdot r_2)$. Afterward, $\mathcal{CS}_1$ sends $\mathsf{E}(\alpha)$ to $\mathcal{CS}_2$.

Step 2: On receiving $\mathsf{E}(\alpha)$, $\mathcal{CS}_2$ first uses sk to recover α. Then, $\mathcal{CS}_2$ checks whether $\alpha < 0$. If yes, $\mathcal{CS}_2$ makes $\theta = 1$ and encrypts it into $\mathsf{E}(\theta)$. Otherwise, $\mathcal{CS}_2$ generates $\mathsf{E}(\theta) = \mathsf{E}(0)$. Next, $\mathcal{CS}_2$ returns $\mathsf{E}(\theta)$ to $\mathcal{CS}_1$.

Step 3: If $s = 1$, $\mathcal{CS}_1$ lets $\mathsf{E}(\delta) = \mathsf{E}(\theta)$. If $s = -1$, $\mathcal{CS}_1$ computes $\mathsf{E}(\delta) = \mathsf{E}(1) + \mathsf{E}(\theta) \cdot \mathsf{E}(-1) = \mathsf{E}(1-\theta)$. Finally, $\mathsf{E}(\delta)$ is the output of our SEQ protocol.

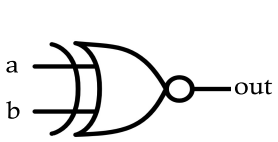
Correctness: Similar to the correctness proof of the SLESS protocol, we can easily obtain that our SEQ protocol is correct.

IV. Our Proposed Scheme

In this section, we first present two novel secure protocols, which will be employed in our proposed scheme. Then, we propose our privacy-preserving dynamic skyline query scheme and online medical diagnosis scheme.

A. Secure Protocols

The main idea of our privacy-preserving dynamic skyline query scheme is to find the data record that has the minimum sum value of all dimensions. This data record must be a skyline point. Then, after removing the data records that are dominated by the skyline point, a new round can be launched to find the next skyline point that has the minimum sum value among the remaining data records. When all data records are removed, it indicates that all skyline points are found. Obviously, there are two important operations in the above idea: 1) finding the data record has the minimum sum value and 2) determining dominance relation of two data records. Although it is easy to execute these two operations over plaintexts, it is challenging to perform them over encrypted data while ensuring both security and efficiency. In this article, we design two secure protocols: 1) secure minimum of n messages (SMINn) protocol and 2) secure dynamic dominance (SDD) protocol, to securely and efficiently achieve the above two operations, respectively.



(a)

Input		Output
a	b	out
0	0	1
0	1	0
1	0	0
1	1	1

(b)

Fig. 3. XNOR gate symbol and truth table. (a) XNOR gate: out = a $\odot$ b. (b) Truth table.

1) SMINn: Before delving into the details of the SMINn protocol, we first introduce a digital logic gate XNOR that takes a logical complement of the exclusive OR (XOR) gate. Typically, XNOR gate has two inputs $\{a, b\}$ and one output (out), and the algebraic notation is out $= a \odot b$, as shown in Fig. 3(a). We present the truth table of XNOR in Fig. 3(b) and observe the following facts: 1) if $a = 0$, out $= 1 - b$ and 2) if $a = 1$, out $= b$. In this article, one of the inputs of XNOR is in plaintext, while the other is in ciphertext. Thus, we have the privacy-preserving version of XNOR as follows:

$$\mathsf{E}(\text{out}) = a \odot \mathsf{E}(b) = \begin{cases} \mathsf{E}(1-b), & \text{if } a = 0 \\ \mathsf{E}(b), & \text{if } a = 1 \end{cases} \tag{2}$$

where $\mathsf{E}(1-b) = \mathsf{E}(1) + \mathsf{E}(-1) \cdot \mathsf{E}(b)$. From (2), we can see that the privacy-preserving XNOR gate guarantees both security and efficiency. First, if the input $\mathsf{E}(b)$ is secret, the output $\mathsf{E}(\text{out})$ would also be secret. Second, it is efficient to compute $\mathsf{E}(\text{out})$ if the operator knows the input a.

Assume $\mathcal{CS}_1$ has a set of encrypted messages $\{\mathsf{E}(m_i) \mid i \in [1, n], m_i \in \mathcal{M}\}$, and $\mathcal{CS}_2$ holds sk. The goal of the SMINn protocol is to find the minimum value $\mathsf{E}(m_{\min})$ among n encrypted values $\{\mathsf{E}(m_i) \mid i \in [1, n]\}$ without leaking underlying plaintexts and access patterns. That is, neither $\mathcal{CS}_1$ nor $\mathcal{CS}_2$ knows the plaintexts of $\{\mathsf{E}(m_{\min}), \mathsf{E}(m_i) \mid i \in [1, n]\}$ or the information about which item in $\{\mathsf{E}(m_i) \mid i \in [1, n]\}$ has the minimum plaintext. Besides, this protocol needs to minimize performance costs.

To achieve the above goals, we propose a private minimum tree, denoted as PM-tree, to find $\mathsf{E}(m_{\min})$ while ensuring both privacy and performance. We depict the process of the tree building as follows.

Step 1: $\mathcal{CS}_1$ first permutes these encrypted values: $\pi(\{\mathsf{E}(m_i) \mid i \in [1, n]\})$, then every 2^k ($k \geq 2$ and $2^k \ll n$) encrypted values are divided into a group, denoted as $\{\mathsf{E}(m'_j) \mid j \in [0, 2^k - 1]\}$. For each group, $\mathcal{CS}_1$ chooses $2^k + 1$ random numbers $\{r_j \mid j \in [0, 2^k]\}$ satisfying $r_{2^k} > r_0, r_1, \ldots, r_{2^k-1}$ and calculates $\mathsf{E}(\widetilde{m}_j) = \mathsf{E}(r_{2^k} \cdot m'_j + r_j)$, where $j \in [0, 2^k - 1]$. Afterward, $\mathcal{CS}_1$ sends $\{\mathsf{E}(\widetilde{m}_j) \mid j \in [0, 2^k - 1]\}$ to $\mathcal{CS}_2$.

Step 2: On receiving $\{\mathsf{E}(\widetilde{m}_j) \mid j \in [0, 2^k - 1]\}$, $\mathcal{CS}_2$ first uses sk to recover $\widetilde{m}_j$. Then, $\mathcal{CS}_2$ compares these plaintexts $\{\widetilde{m}_j \mid j \in [0, 2^k - 1]\}$ and obtains the index

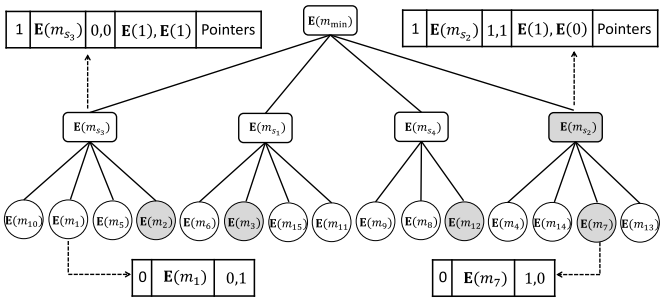


Fig. 4. Example of PM-tree.

TABLE I
FIELDS OF LEAF AND NONLEAF NODES

Field Name	Description
id	Level id, encoding from bottom to top
value	Encrypted message of the node
gIndex	Node index in its group, binary
mIndex	Minimum value's index, encrypted binary
pointers	Point to node's children

ϕ of the minimum one. After that, $\mathcal{CS}_2$ converts ϕ to a bit sequence $b_\phi \in \{0,1\}^k$ and generates $\{\mathsf{E}(b_\phi[i]) \mid i \in [0, k-1]\}$ by encrypting each bit of b_ϕ. Next, $\mathcal{CS}_2$ sends $\{\mathsf{E}(b_\phi[i]) \mid i \in [0, k-1]\}$ to $\mathcal{CS}_1$.

Step 3: According to the order of the members in group $\{\mathsf{E}(\widetilde{m}_j) \mid j \in [0, 2^k - 1]\}$, $\mathcal{CS}_1$ encodes the index j of each member into a bit sequence, denoted as $\{a_j[i] \mid i \in [0, k-1]\}$. Then, with $\{\mathsf{E}(b_\phi[i]) \mid i \in [0, k-1]\}$, $\mathcal{CS}_1$ runs the privacy-preserving XNOR gate to compute flags $\{\mathsf{E}(\delta_j) \mid j \in [0, 2^k - 1]\}$ for each member in the group, i.e., $\mathsf{E}(\delta_j) = \prod_{i=0}^{k-1}(a_j[i] \odot \mathsf{E}(b_\phi[i]))$. Finally, $\mathcal{CS}_1$ computes the encrypted minimum value of the group: $\mathsf{E}(m_s) = \sum_{j=0}^{2^k-1}(\mathsf{E}(\delta_j) \cdot \mathsf{E}(m'_j))$.

After obtaining the encrypted minimum values for each group, $\mathcal{CS}_1$ collects all of them and repeatably performs steps 1–3 until there is only one encrypted minimum value. This encrypted value would contain the minimum value among n encrypted values. When PM-tree is built, the root node is the output of the SMINn protocol.

Correctness: For one group, only if $\phi = j$, the corresponding member $\mathsf{E}(m'_j)$ is selected as the minimum value. $\mathcal{CS}_1$ encodes j into a bit sequence $\{a_j[i]\}$, while $\mathcal{CS}_2$ returns an encrypted bit sequence $\{\mathsf{E}(b_\phi[i])\}$. From our privacy-preserving XNOR operation, we know that $\mathsf{E}(\delta_j) = \mathsf{E}(1)$ *iff* $\phi = j$, otherwise, $\mathsf{E}(\delta_j) = \mathsf{E}(0)$. Therefore, $\mathsf{E}(m_s) = \mathsf{E}(1) \cdot \mathsf{E}(m'_\phi) = \mathsf{E}(m'_\phi)$ is the encrypted minimum value of the group. Since the minimum value is further selected from groups' minimum values, the last group's minimum value is $\mathsf{E}(m_{\min})$.

As shown in Fig. 4, we take an example of PM-tree, in which we assume $n = 15, k = 2$, and $m_{\min} = m_7$. In PM-tree, the leaf node has three fields: (*id, value*, and *gIndex*), while the nonleaf node has five fields: (*id, value, gIndex, mIndex,* and *pointers*). We list the detailed description of these fields in Table I. Note that the *gIndex* of root node is empty, denoted as $\perp$. Taking computing $\mathsf{E}(m_{s_2})$ as an example, $\mathcal{CS}_2$ can obtain the index $\phi = 2$ after receiving $\{\mathsf{E}(m_4), \mathsf{E}(m_{14}), \mathsf{E}(m_7), \mathsf{E}(m_{13})\}$. Then, $\mathcal{CS}_2$ generates $\{\mathsf{E}(b_\phi[0]) = \mathsf{E}(1), \mathsf{E}(b_\phi[1]) = \mathsf{E}(0)\}$ and returns them to $\mathcal{CS}_1$. Since the index of $\{\mathsf{E}(m_4), \mathsf{E}(m_{14}), \mathsf{E}(m_7), \mathsf{E}(m_{13})\}$ is encoded as $\{00, 01, 10, 11\}$, i.e., $\{a_0[i], a_1[i], a_2[i], a_3[i] \mid i \in [0, 1]\}$. By using our privacy-preserving XNOR gate, only $\mathsf{E}(\delta_2) = \mathsf{E}(1)$, while $\mathsf{E}(\delta_0)$, $\mathsf{E}(\delta_1)$, and $\mathsf{E}(\delta_3)$ are $\mathsf{E}(0)$. Thus, $\mathsf{E}(m_{s_2}) = \mathsf{E}(\delta_0) \cdot \mathsf{E}(m_4) + \mathsf{E}(\delta_1) \cdot \mathsf{E}(m_{14}) + \mathsf{E}(\delta_2) \cdot \mathsf{E}(m_7) + \mathsf{E}(\delta_3) \cdot \mathsf{E}(m_{13}) = \mathsf{E}(m_7)$.

Remark: To obtain minimum value $\mathsf{E}(m_{\min})$ among n encrypted values, an intuitive approach [4] is to compute a minimum value of two encrypted values and then use it and the third encrypted value to compute the minimum value among these three encrypted values. Repeating $n-1$, $\mathsf{E}(m_{\min})$ can be computed. However, this approach needs $n-1$ rounds and $2 \cdot (n-1)$ ciphertexts in terms of communication costs. In our PM-tree, it requires $\lceil([n-1]/[2^k-1])\rceil$ rounds and $\lceil([n-1]/[2^k-1])\rceil \cdot (2^k + k)$ ciphertexts communication to complete our SMINn protocol. It is definite that our PM-tree has less communication rounds in all cases. For the number of ciphertexts, when $k = 2$, it is approximate. When $k > 3$, PM-tree transfers less ciphertexts.

2) SDD: Assume $\mathcal{CS}_1$ has two encrypted d-dimensional data records $\mathsf{E}(\mathtt{x}_1), \mathsf{E}(\mathtt{x}_2)$, an encrypted query point $\mathsf{E}(\mathtt{q})$ and pk, where $\mathsf{E}(\mathtt{x}_i) = (\mathsf{E}(\mathtt{x}_i^1), \mathsf{E}(\mathtt{x}_i^2), \ldots, \mathsf{E}(\mathtt{x}_i^d))$ for $i = 1, 2$, and $\mathsf{E}(\mathtt{q}) = (\mathsf{E}(\mathtt{q}^1), \mathsf{E}(\mathtt{q}^2), \ldots, \mathsf{E}(\mathtt{q}^d))$, while $\mathcal{CS}_2$ holds sk. The goal of the SDD protocol is to determine whether $\mathtt{x}_1$ dynamically dominates $\mathtt{x}_2$ with regard to $\mathtt{q}$ (see Definition 1) over encrypted data. If yes, the SDD protocol outputs $\mathsf{E}(1)$, otherwise, $\mathsf{E}(0)$. Before introducing the protocol, we assume $\mathcal{CS}_1$ has obtained $\mathsf{E}(\mathtt{t}_1^i) = \mathsf{E}((\mathtt{x}_1^i - \mathtt{q}^i)^2)$ and $\mathsf{E}(\mathtt{t}_2^i) = \mathsf{E}((\mathtt{x}_2^i - \mathtt{q}^i)^2)$ for $i \in [1, d]$ by using homomorphic properties.

Following Definition 1, a solution [4] is to check whether $\mathtt{t}_1^i \leq \mathtt{t}_2^i$ for each dimension. After obtaining d outputs, denoted as $\{\mathsf{E}(\delta^i) \mid i \in [1, d]\}$, $\mathcal{CS}_1$ can perform the homomorphic multiplication-I operation to obtain $\mathsf{E}(\delta) = \prod_{i=1}^{d}\mathsf{E}(\delta^i)$. After that, $\mathcal{CS}_1$ can have the encrypted dynamic dominance relation by determining whether $\exists i$ such that $\mathtt{t}_1^i < \mathtt{t}_2^i$ with the SLESS protocol. However, this approach is inefficient in the communication cost and needs too many homomorphic multiplication-I operations. To improve the efficiency and lower the number of the homomorphic multiplication-I operation, we present our SDD protocol as follows.

Step 1: For the ith-dimension, $\mathcal{CS}_1$ flips a coin $s_i \in \{-1, 1\}$ and chooses two random numbers $r_1^i, r_2^i \in \{0,1\}^{k_1}$ satisfying $r_1^i > r_2^i > 0$. Afterward, $\mathcal{CS}_1$ computes $\mathsf{E}(\alpha^i) = \mathsf{E}(s_i \cdot r_1^i \cdot (\mathtt{t}_1^i - \mathtt{t}_2^i) - s_i \cdot r_2^i)$. If $s_i = 1$, $\theta_1^i = 2^i$, otherwise, $\theta_1^i = 0$. Next, $\mathcal{CS}_1$ computes $\theta_1 = \sum_{i=1}^{d}\theta_1^i$, generates $\mathsf{E}(\theta_1)$ using (1), and sends $\{\mathsf{E}(\alpha^i) \mid i \in [1, d]\}$ to $\mathcal{CS}_2$.

Step 2: On receiving $\{\mathsf{E}(\alpha^i) \mid i \in [1, d]\}$, $\mathcal{CS}_2$ first uses sk to recover α^i. If $\alpha^i < 0$, $\mathcal{CS}_2$ computes $\theta_2^i = 2^i$. Otherwise, $\theta_2^i = 0$. Then, $\mathcal{CS}_2$ computes $\theta_2 = \sum_{i=1}^{d}\theta_2^i$ and encrypts it into $\mathsf{E}(\theta_2)$. Next, $\mathcal{CS}_2$ sends $\mathsf{E}(\theta_2)$ to $\mathcal{CS}_1$.

Step 3: With $\mathsf{E}(\theta_1)$ (calculated in step 1) and $\mathsf{E}(\theta_2)$, $\mathcal{CS}_1$ runs the SEQ protocol to test whether $\theta_1 \stackrel{?}{=} \theta_2$. If $\theta_1 = \theta_2$, the SEQ protocol returns $\mathsf{E}(\delta_1) = \mathsf{E}(1)$, otherwise, $\mathsf{E}(\delta_1) = \mathsf{E}(0)$.

Step 4: $\mathcal{CS}_2$ adds up the values of all dimensions of $\mathsf{E}(\mathtt{t}_1^i)$ and $\mathsf{E}(\mathtt{t}_2^i)$ ($i \in [1, d]$), i.e., $\mathsf{E}(\mathtt{s}_{\mathtt{t}_1}) = \mathsf{E}(\sum_{i=1}^{d} \mathtt{t}_1^i)$ and $\mathsf{E}(\mathtt{s}_{\mathtt{t}_2}) = \mathsf{E}(\sum_{i=1}^{d} \mathtt{t}_2^i)$. Next, $\mathcal{CS}_1$ runs the SLESS protocol to determine whether $\mathtt{s}_{\mathtt{t}_1} < \mathtt{s}_{\mathtt{t}_2}$. If yes, the SLESS protocol returns $\mathsf{E}(\delta_2) = \mathsf{E}(1)$, otherwise, $\mathsf{E}(\delta_2) = \mathsf{E}(0)$.

Step 5: With $\mathsf{E}(\delta_1)$ and $\mathsf{E}(\delta_2)$, $\mathcal{CS}_1$ computes $\mathsf{E}(\delta) = \mathsf{E}(\delta_1) \cdot \mathsf{E}(\delta_2) = \mathsf{E}(\delta_1 \cdot \delta_2)$ as the output of our SDD protocol.

Correctness: Since checking dynamic dominance relations only needs to compare the absolute values, it is equivalent to comparing the square of the differences. When $s_i = 1$, we have $\theta_1^i = 2^i$. If $\mathtt{t}_1^i - \mathtt{t}_2^i \leq 0$, $\alpha^i = (r_1^i \cdot (\mathtt{t}_1^i - \mathtt{t}_2^i) - r_2^i) < 0$, leading to $\theta_2^i = 2^i$. As a result, $\theta_1^i = \theta_2^i$. If $\mathtt{t}_1^i - \mathtt{t}_2^i > 0$, it means $\alpha^i > 0$, leading to $\theta_2^i = 0$. As a result, $\theta_1^i = 2^i \neq \theta_2^i = 0$. Thus, when $s_i = 1$, *iff* $\forall i \in [1, d]$: $\mathtt{t}_1^i - \mathtt{t}_2^i \leq 0$, we have $\theta_1 = \theta_2$ and $\mathsf{E}(\delta_1) = \mathsf{E}(1)$. Similarly, we can prove that when $s_i = -1$, *iff* $\forall i \in [1, d]$: $\mathtt{t}_1^i - \mathtt{t}_2^i \leq 0$, we have $\mathsf{E}(\delta_1) = \mathsf{E}(1)$. Furthermore, $\exists i$, $\mathtt{t}_1^i < \mathtt{t}_2^i$ (under the condition of $\forall i$, $\mathtt{t}_1^i \leq \mathtt{t}_2^i$) indicates that $\mathtt{s}_{\mathtt{t}_1} < \mathtt{s}_{\mathtt{t}_2}$. In this case, SLESS returns $\mathsf{E}(\delta_2) = \mathsf{E}(1)$. Otherwise, $\mathsf{E}(\delta_2) = \mathsf{E}(0)$. Thus, $\mathsf{E}(\delta) = \mathsf{E}(\delta_1 \cdot \delta_2) = \mathsf{E}(1)$, *iff* $\mathsf{E}(\mathtt{x}_1)$ dynamically dominates $\mathsf{E}(\mathtt{x}_2)$.

Remark: We can see that our proposed SDD protocol only needs two rounds of communication to determine whether $\forall i$, $\mathtt{t}_1^i \leq \mathtt{t}_2^i$, while there are d rounds for the solution in [4]. Besides, the maximum multiplicative depth of our SDD protocol is one, while it is also d for the protocol in [4].

Algorithm 1 Retrieve Dynamic Skyline

Input: An encrypted dataset $\mathsf{E}(\mathcal{X})$ with n records. A query point $\mathsf{E}(\mathtt{q})$. An encrypted maximum value $\mathsf{E}(\mathtt{MAX})$.
Output: A set containing encrypted dynamic skyline points, $\mathcal{S}_{\text{sky}}$.
1: $\mathcal{S}_{\text{sky}} \leftarrow \varnothing$; $\mathcal{T} \leftarrow \varnothing$; $\mathcal{S} \leftarrow \varnothing$;
2: $\mathcal{S}_d \leftarrow \{\mathsf{E}(\rho_i) = \mathsf{E}(0) | i \in [1, n]\}$
3: **for** $i = 1$ to n **do**
4: $\quad \mathsf{E}(\mathtt{t}_i^j) = \mathsf{E}((\mathtt{x}_i^j - \mathtt{q}^j)^2)$, $j \in [1, d]$, and adds it into set $\mathcal{T}$.
5: $\quad \mathsf{E}(\mathtt{s}_i) = \mathsf{E}((n+1) \cdot \sum_{j=1}^{d} \mathtt{t}_i^j + i)$, and adds it into set $\mathcal{S}$.
6: $\mathsf{E}(\mathtt{s}_{\min})$, PM-tree $\leftarrow$ SMINn($\{\mathsf{E}(\mathtt{s}_i) | i \in [1, n]\}$).
7: **if** LESS($\mathsf{E}(\mathtt{s}_{\min})$, $\mathsf{E}(\mathtt{MAX})$)==0 **then**
8: $\quad$ break
9: root $\leftarrow$ PM-tree.root; $\quad$ eflags $\leftarrow \varnothing$
10: mIndexList.addAll(root.*mIndex*)
11: RecursiveSearch(root, mIndexList, eflags)
12: **for** $j = 1$ to d **do**
13: $\quad \mathsf{E}(\mathtt{x}_{\text{sky}}^j) \leftarrow \sum_{i=1}^{n} (\mathsf{E}(\mathtt{x}_i^j) \cdot \text{eflags}[i-1])$
14: $\quad \mathsf{E}(\mathtt{t}_{\text{sky}}^j) \leftarrow \sum_{i=1}^{n} (\mathsf{E}(\mathtt{t}_i^j) \cdot \text{eflags}[i-1])$
15: $\mathsf{E}(\mathtt{x}_{\text{sky}}) = (\mathsf{E}(\mathtt{x}_{\text{sky}}^1), \mathsf{E}(\mathtt{x}_{\text{sky}}^2), \cdots, \mathsf{E}(\mathtt{x}_{\text{sky}}^d))$
16: $\mathsf{E}(\mathtt{t}_{\text{sky}}) = (\mathsf{E}(\mathtt{t}_{\text{sky}}^1), \mathsf{E}(\mathtt{t}_{\text{sky}}^2), \cdots, \mathsf{E}(\mathtt{t}_{\text{sky}}^d))$
17: $\mathcal{S}_{\text{sky}}$.add($\mathsf{E}(\mathtt{x}_{\text{sky}})$)
18: **for** $i = 1$ to n **do**
19: $\quad \mathsf{E}(\delta_i) \leftarrow$ Modified SDD($\mathsf{E}(\mathtt{t}_{\text{sky}})$, $\mathsf{E}(\mathtt{t}_i)$, eflags$[i-1]$)
20: $\quad \mathsf{E}(\rho_i) \leftarrow \mathsf{E}(\rho_i) + \mathsf{E}(\delta_i)$
21: $\quad \mathsf{E}(\tilde{\mathtt{s}}_i) \leftarrow \mathsf{E}(\rho_i) \cdot (\mathsf{E}(\mathtt{MAX}) - \mathsf{E}(\mathtt{s}_i)) + \mathsf{E}(\mathtt{s}_i)$
22: $\mathsf{E}(\mathtt{s}_{\min})$, PM-tree $\leftarrow$ SMINn($\{\mathsf{E}(\tilde{\mathtt{s}}_i) | i \in [1, n]\}$).
23: GOTO line 7
24:
25: **function** RecursiveSearch(node, mIndexList, eflags)
26: $\quad$ **for** each childNode of node **do**
27: $\quad\quad$ gIndexList.addAll(childNode.*gIndex*)
28: $\quad\quad$ **if** childNode is leaf **then**
29: $\quad\quad\quad$ $l \leftarrow$ gIndexList.len()
30: $\quad\quad\quad$ $\mathsf{E}(\delta) \leftarrow \prod_{i=0}^{l-1}$ gIndexList[i] $\odot$ mIndexList[i]
31: $\quad\quad\quad$ eflags.add($\mathsf{E}(\delta)$)
32: $\quad\quad$ **else**
33: $\quad\quad\quad$ mIndexList.addAll(childNode.*mIndex*)
34: $\quad\quad\quad$ RecursiveSearch(childNode, mIndexList, eflags)

B. Privacy-Preserving Dynamic Skyline Query

Given an encrypted data set $\{\mathsf{E}(\mathtt{x}_i) = (\mathsf{E}(\mathtt{x}_i^1), \mathsf{E}(\mathtt{x}_i^2), \ldots, \mathsf{E}(\mathtt{x}_i^d)) \,|\, i \in [1, n]\}$ and an encrypted query point $\mathsf{E}(\mathtt{q}) = (\mathsf{E}(\mathtt{q}^1), \mathsf{E}(\mathtt{q}^2), \ldots, \mathsf{E}(\mathtt{q}^d))$, our privacy-preserving dynamic skyline query scheme can efficiently find dynamic skyline without leaking underlying plaintexts (including data set, query point, and query results), single-dimensional privacy, and access patterns to $\mathcal{CS}_1$ or $\mathcal{CS}_2$. Here, the single-dimensional privacy indicates the order or equality relation of values in each dimension, while the access patterns mean the information about which data records are selected as the final skyline. From the main idea discussed in Section IV-A, there are three phases to securely compute dynamic skyline: 1) preprocessing; 2) finding one skyline point; and 3) updating sum values, which is formally depicted in Algorithm 1.

1) Preprocessing: In the preprocessing phase, $\mathcal{CS}_1$ generates two encrypted sets by using the homomorphic properties

$$\mathcal{T} = \left\{ \mathsf{E}(\mathtt{t}_i) = \left(\mathsf{E}\left(\mathtt{t}_i^1\right), \mathsf{E}\left(\mathtt{t}_i^2\right), \ldots, \mathsf{E}\left(\mathtt{t}_i^d\right) \right) \,|\, 1 \leq i \leq n \right\}$$
$$\mathcal{S} = \left\{ \mathsf{E}(\mathtt{s}_i) = \mathsf{E}\left((n+1) \cdot \sum_{j=1}^{d} \mathtt{t}_i^j + i \right) \,|\, 1 \leq i \leq n \right\} \quad (3)$$

where $\mathsf{E}(\mathtt{t}_i^j) = \mathsf{E}((\mathtt{x}_i^j - \mathtt{q}^j)^2)$, and the items in $\mathcal{S}$ set are the sum of all dimensions of the corresponding items in $\mathcal{T}$ set. To avoid the equal summed values in $\mathcal{S}$ set, we first scale the sum value by multiplying $n + 1$ and then add the corresponding index value into the scaled value. This approach makes the sum values different without changing their order relations. Besides, we initialize a dominance flag set $\mathcal{S}_d$ with n encrypted 0, i.e., $\mathsf{E}(0)$, which will be used in the updating sum values phase. See lines 1–5 for the preprocessing phase.

2) Finding One Skyline Point: In our scheme, the key idea of finding a skyline point is to search the data record that has the minimum sum value. As a result, we can obtain an encrypted skyline point by finding the minimum value $\mathsf{E}(\mathtt{s}_{\min})$ in $\mathcal{S}$ set with our SMINn protocol. However, since we hide access patterns in our SMINn protocol, i.e., the cloud does not know which item in $\mathcal{S}$ has the minimum value, $\mathcal{CS}_1$ cannot directly obtain the skyline point according to $\mathsf{E}(\mathtt{s}_{\min})$. To tackle this issue, we propose a novel approach by using our privacy-preserving XNOR gate and PM-tree. Specifically, $\mathcal{CS}_1$ traverses the PM-tree and collects both the *gIndex* and *mIndex* on the path. When reaching a leaf node, $\mathcal{CS}_1$ uses the privacy-preserving XNOR gate (see details in Section IV-A1) to determine whether the *gIndex* bit sequence and *mIndex* bit sequence (ciphertexts) are the same. If yes, the corresponding leaf node $\mathsf{E}(\mathtt{x}_i)$ has an encrypted flag $\mathsf{E}(f_i) = \mathsf{E}(1)$. Otherwise, $\mathsf{E}(f_i) = \mathsf{E}(0)$. Since only the minimum value in $\mathcal{S}$ has a flag

$\mathsf{E}(f_i) = \mathsf{E}(1)$, $\mathcal{CS}_1$ can compute the skyline point by multiplying the leaf node's value and the corresponding flag and then adding up all the results

$$\mathsf{E}\left(\mathrm{x}_{\mathrm{sky}}^{j}\right) = \sum_{i=1}^{n}\left(\mathsf{E}\left(\mathrm{x}_{i}^{j}\right)\cdot \mathsf{E}(f_i)\right)$$
$$\mathsf{E}\left(\mathrm{t}_{\mathrm{sky}}^{j}\right) = \sum_{i=1}^{n}\left(\mathsf{E}\left(\mathrm{t}_{i}^{j}\right)\cdot \mathsf{E}(f_i)\right) \quad (4)$$

where $j \in [1, d]$. Finally, the skyline point is: $\mathsf{E}(\mathrm{x}_{\mathrm{sky}}) = (\mathsf{E}(\mathrm{x}_{\mathrm{sky}}^1), \mathsf{E}(\mathrm{x}_{\mathrm{sky}}^2), \ldots, \mathsf{E}(\mathrm{x}_{\mathrm{sky}}^d))$. This phase is shown from lines 9 to 17. Note that $\mathsf{E}(\mathrm{t}_{\mathrm{sky}}^j)$ will be used in the next phase.

3) Updating Sum Values: Before finding the next skyline point by the above approach, we need to "remove" the skyline point found in the second phase and the data records that are dominated by this skyline point from the original data set $\{\mathsf{E}(\mathrm{x}_i) \mid i \in [1, n]\}$. Our approach is to update the sum values of the data records that need to be removed and thus, make them impossible to become the new skyline point.

Updating Sum Values of Dominated Data Records: With $\mathsf{E}(\mathrm{t}_{\mathrm{sky}}) = (\mathsf{E}(\mathrm{t}_{\mathrm{sky}}^1), \mathsf{E}(\mathrm{t}_{\mathrm{sky}}^2), \ldots, \mathsf{E}(\mathrm{t}_{\mathrm{sky}}^d))$ and the $\mathcal{T}$ set, $\mathcal{CS}_1$ can adopt the SDD protocol to check whether $\mathsf{E}(\mathrm{x}_{\mathrm{sky}})$ dynamically dominates the points in $\{\mathsf{E}(\mathrm{x}_i) \mid i \in [1, n]\}$. If yes, the SDD protocol outputs $\mathsf{E}(\delta_i) = \mathsf{E}(1)$ for the corresponding data record $\mathsf{E}(\mathrm{x}_i)$. Otherwise, $\mathsf{E}(\delta_i) = \mathsf{E}(0)$. Consequently, the sum value of each data record can be updated by

$$\mathsf{E}(\mathrm{s}_i) = \mathsf{E}(\delta_i)\cdot(\mathsf{E}(\mathtt{MAX}) - \mathsf{E}(\mathrm{s}_i)) + \mathsf{E}(\mathrm{s}_i) \quad (5)$$

where $\mathtt{MAX} > \max(\mathrm{s}_i)$ is a preset value. When $\mathsf{E}(\delta_i) = \mathsf{E}(0)$, the equation will keep the original sum value. When $\mathsf{E}(\delta_i) = \mathsf{E}(1)$, the sum value will be update as MAX. Since $\mathtt{MAX} > \max(\mathrm{s}_i)$, the dominated data points will be impossible to be a skyline point in the next round, which is equivalent to removing them from the original data set. However, since the updated sum values will be used as inputs for the SMINn protocol to find a new skyline point in the next round, the number of homomorphic multiplication-I operation may increase with the number of skyline search rounds. To achieve a fixed multiplicative depth, we still use the idea of (5) to update sum values but with a different approach. First, we initialize a dominance flag set $\mathcal{S}_d = \{\mathsf{E}(\rho_i) = \mathsf{E}(0) \mid i \in [1, n]\}$ in the first phase and update $\mathsf{E}(\rho_i)$ using the homomorphic addition-I operation

$$\mathsf{E}(\rho_i) = \mathsf{E}(\rho_i) + \mathsf{E}(\delta_i) = \mathsf{E}(\rho_i + \delta_i).$$

Then, the sum value can be updated as the following equation:

$$\mathsf{E}(\tilde{\mathrm{s}}_i) = \mathsf{E}(\rho_i)\cdot(\mathsf{E}(\mathtt{MAX}) - \mathsf{E}(\mathrm{s}_i)) + \mathsf{E}(\mathrm{s}_i) \quad (6)$$

in which $\mathsf{E}(\tilde{\mathrm{s}}_i)$ is the updated sum value, and $\mathsf{E}(\mathrm{s}_i)$ is the original sum value of $\mathsf{E}(\mathrm{x}_i)$. If $\mathsf{E}(\mathrm{x}_i)$ is not dominated by other skyline points, $\mathsf{E}(\rho_i) = \mathsf{E}(0)$ and the original sum value will be kept. Otherwise, $\mathsf{E}(\rho_i) \geq \mathsf{E}(1)$, the updated sum value will be no less than MAX. This approach only involves three homomorphic multiplication-I operations no matter how many rounds. One is from step 5 in the SDD protocol, and the others are from (6). See lines 2 and 18–21 for details.

Updating Sum Value of the Skyline Point: However, there is still a problem, i.e., $\{\mathsf{E}(\mathrm{x}_i) \mid i \in [1, n]\}$ exists a data record that is the same as $\mathsf{E}(\mathrm{x}_{\mathrm{sky}})$. We denote the data point as $\mathsf{E}(\mathrm{x}_{\mathrm{s}})$ and the corresponding difference square as $\mathsf{E}(\mathrm{t}_{\mathrm{s}})$. The SDD protocol will output $\mathsf{E}(0)$ for $\mathsf{E}(\mathrm{x}_{\mathrm{s}})$ when checking the dominance relation between $\mathsf{E}(\mathrm{x}_{\mathrm{sky}})$ and $\mathsf{E}(\mathrm{x}_{\mathrm{s}})$. It means the sum value of $\mathsf{E}(\mathrm{t}_{\mathrm{s}})$ will not be updated. Consequently, $\mathsf{E}(\mathrm{x}_{\mathrm{s}})$ is not removed from the original data set. To deal with the problem, we devise a low-cost approach by modifying the SDD protocol without affecting the regular function of the protocol. In our SDD protocol, the key point to output $\mathsf{E}(0)$ for $\mathsf{E}(\mathrm{x}_{\mathrm{s}})$ is step 4 (Section IV-A2). That is because $\mathrm{x}_{\mathrm{sky}}$ and x_{s} are the same in each dimension, leading to $\mathrm{s}_{\mathrm{t}_{\mathrm{sky}}}$ to be equal to $\mathrm{s}_{\mathrm{t}_s}$. As a result, $\mathsf{E}(\delta_2) = \mathsf{E}(0) \Rightarrow \mathsf{E}(\delta) = \mathsf{E}(0)$ is the output of our SDD protocol. If we modify $\mathsf{E}(\mathrm{s}_{\mathrm{t}_2}) = \mathsf{E}(\sum_{i=1}^{d} \mathrm{t}_2^i)$ to $\mathsf{E}(\mathrm{s}_{\mathrm{t}_2}) = \mathsf{E}(\sum_{i=1}^{d} \mathrm{t}_2^i) + \mathsf{E}(f_2)$ in step 4 of Section IV-A2, where $\mathsf{E}(f_2)$ is the corresponding minimum flag of x_2 (see the second phase in this section), the modified SDD protocol will output $\mathsf{E}(1)$ for $\mathsf{E}(\mathrm{x}_{\mathrm{s}})$, because the minimum flag of $\mathsf{E}(\mathrm{x}_{\mathrm{s}})$ is $\mathsf{E}(1)$, i.e., $\mathsf{E}(f_s) = \mathsf{E}(1)$, and $\mathsf{E}(\mathrm{s}_{\mathrm{t}_{\mathrm{sky}}}) < \mathsf{E}(\mathrm{s}_{\mathrm{t}_s})$. Besides, only $\mathsf{E}(f_s)$ is $\mathsf{E}(1)$, and the minimum flags of the other data records are $\mathsf{E}(0)$. Therefore, the modified SDD protocol does not affect the other data points in checking dominance relations.

After updating the sum values of dominated data points and skyline point, $\mathcal{CS}_1$ goes back to the second phase to find a new skyline point. Repeat the second and the third phases until the minimum sum value $\mathrm{s}_{\min} \geq \mathtt{MAX}$, which means all skyline points have been added into the skyline set $\mathcal{S}_{\mathrm{sky}}$. Here, to determine whether $\mathrm{s}_{\min} < \mathtt{MAX}$, we use an LESS protocol by making $\mathcal{CS}_2$ return plaintext θ in the SLESS protocol.

Discussion About Maximum Multiplicative Depth: We carefully design our scheme to achieve a relatively small multiplicative depth. In the preprocessing phase, there are two homomorphic multiplication-I operations, i.e., calculating $\mathsf{E}(\mathrm{t}_i^j)$. In the second phase, the SMINn protocol will incur $\lceil \log n \rceil - 1$ homomorphic multiplication-I operations to compute the corresponding flags. Totally, there will be $\lceil \log n \rceil - 1 + 1 = \lceil \log n \rceil$ homomorphic multiplication-I operations in the phase. As for the third phase, we have known that there are three homomorphic multiplication-I operations. Therefore, the maximum multiplicative depth of our scheme is $\lceil \log n \rceil + 5$.

C. Description of Our Proposed Scheme

In this section, we present our online medical diagnosis system, which is comprised of four phases: 1) system initialization; 2) data outsourcing; 3) dynamic skyline search; and 4) local diagnosis.

1) System Initialization: The service provider $\mathcal{SP}$ is responsible for initializing the whole system. First, given the security parameters (k_0, k_1, k_2), $\mathcal{SP}$ uses $\mathsf{KeyGen}(k_0, k_1, k_2)$ to generate the secret key *sk*. Meanwhile, $\mathcal{SP}$ generates ciphertexts $\{\mathsf{E}(0)_1, \mathsf{E}(0)_2, \mathsf{E}(-1)\}$, in which the set $\{\mathsf{E}(0)_1, \mathsf{E}(0)_2\}$ is the pubic key *pk* of SHE, and $\mathsf{E}(-1)$ is a public parameter. Then, $\mathcal{SP}$ chooses a secure hash function $\mathsf{H}()$, e.g., SHA-256, and a public-key encryption $\mathsf{PKE}()$, e.g., ElGamal. After that, $\mathcal{SP}$ generates three master keys k_o, k_u, and k_p for data owners, query users, and patients, respectively.

1) When a data owner o_i with its identity $\mathtt{ID}_{oi}$ registers to the system, $\mathcal{SP}$ authorizes $k_{oi} = \mathsf{H}(\mathtt{ID}_{oi}, k_o)$ to o_i.
2) When a query user (doctor) u_i with his/her identity $\mathtt{ID}_{ui}$ registers to the system, $\mathcal{SP}$ authorizes $k_{ui} = \mathsf{H}(\mathtt{ID}_{ui}, k_u)$ and k_p to u_i. Besides, $\mathcal{SP}$ calls the key generation of $\mathsf{PKE}()$ to generate a pair (sk_{ui}, pk_{ui}) for u_i and sends sk_{ui} to u_i.
3) When a patient p_i with his/her identity $\mathtt{ID}_{pi}$ registers to the system, $\mathcal{SP}$ authorizes $k_{pi} = \mathsf{H}(\mathtt{ID}_{pi}, k_p)$ to p_i.

Next, $\mathcal{SP}$ publishes $\{pk, \mathsf{H}(), \mathsf{PKE}(), \mathsf{E}(-1)\}$, sends $\{k_o, k_u\}$ to $\mathcal{CS}_1$, and authorizes sk to $\mathcal{CS}_2$.

2) Data Outsourcing: Assume a data owner o_i has a medical data set $\mathcal{X} = \{\mathtt{x}_i \mid 0 \le i \le n\}$, and each data record has d dimensions. With pk, o_i first encrypts these data records into $\{\mathsf{E}(\mathtt{x}_i) \mid i \in [1, n]\}$ by using (1). Here, we ignore the diagnosis results for ease of description, since they are not involved in the dynamic skyline search. Then, o_i chooses the first encrypted data record $\mathsf{E}(\mathtt{x}_1)$ to compute $\mathsf{H}(\mathsf{E}(\mathtt{x}_1), k_{oi})$. Next, o_i transfers $\{\mathsf{E}(\mathtt{x}_i) \mid i \in [1, n]\}||\mathsf{H}(\mathsf{E}(\mathtt{x}_1), k_{oi})||\mathtt{ID}_{oi}$ to $\mathcal{CS}_1$. Upon receiving it, $\mathcal{CS}_1$ first computes $k_{oi} = \mathsf{H}(\mathtt{ID}_{oi}, k_o)$ with the authorized k_o. Then, $\mathcal{CS}_1$ extracts $\mathsf{E}(\mathtt{x}_1)$ from $\{\mathsf{E}(\mathtt{x}_i) \mid i \in [1, n]\}$ and calculates $\mathsf{H}(\mathsf{E}(\mathtt{x}_1), k_{oi})$. If the calculated $\mathsf{H}(\mathsf{E}(\mathtt{x}_1), k_{oi})$ is the same as the received $\mathsf{H}(\mathsf{E}(\mathtt{x}_1), k_{oi})$, $\mathcal{CS}_1$ accepts these encrypted data records, otherwise, rejects.

3) Dynamic Skyline Search: When a patient p_i would like to utilize the online medical diagnosis services, he/she would first choose a doctor (query user) u_i and encrypt the sensed physiological data $\mathtt{q} = (\mathtt{q}^1, \mathtt{q}^2, \ldots, \mathtt{q}^d)$ with the doctor's public key pk_{ui}, which is denoted as $\mathsf{E}_{\mathsf{PKE}}(\mathtt{q})$. Then, p_i generates and sends $\mathsf{E}_{\mathsf{PKE}}(\mathtt{q})||\mathsf{H}(\mathsf{E}_{\mathsf{PKE}}(\mathtt{q}), k_{pi})||\mathtt{ID}_{pi}$ to the doctor u_i. After receiving it, u_i first computes $k_{pi} = \mathsf{H}(\mathtt{ID}_{pi}, k_p)$ with k_p and checks whether $\mathsf{H}(\mathsf{E}_{\mathsf{PKE}}(\mathtt{q}), k_{pi})$ is correct. If yes, u_i accepts the diagnosis request and recovers the sensed data $\mathtt{q}$ from $\mathsf{E}_{\mathsf{PKE}}(\mathtt{q})$ with sk_{ui}, otherwise, rejects.

After that, u_i launches a dynamic skyline query to search the similar data records with regard to $\mathtt{q}$. First, u_i encrypts $\mathtt{q}$ into $\mathsf{E}(\mathtt{q})$ with the public key pk of SHE. Then, u_i generates and sends $\mathsf{E}(\mathtt{q})||\mathsf{H}(\mathsf{E}(\mathtt{q}), k_{ui})||\mathtt{ID}_{ui}$ to $\mathcal{CS}_1$. Using the similar approach introduced in the data outsourcing phase, $\mathcal{CS}_1$ checks whether $\mathsf{H}(\mathsf{E}(\mathtt{q}), k_{ui})$ is correct. If yes, $\mathcal{CS}_1$ performs our privacy-preserving dynamic skyline query scheme (Section IV-B) to obtain the encrypted dynamic skyline points. We denote these skyline points as $\{\mathsf{E}(\tilde{\mathtt{x}}_i) \mid i \in [1, w]\}$, where w is the number of skyline points.

Next, $\mathcal{CS}_1$ generates a random number r and computes $\{r_i^j = \mathsf{H}(r, i||j) \mid i \in [1, w], j \in [1, d], r_i^j \in \{0, 1\}^{k_1}\}$. Furthermore, $\mathcal{CS}_1$ adds these random numbers (noises) to the corresponding skyline point, i.e., $\mathsf{E}(\tilde{\mathtt{x}}_i^j + r_i^j)$. Afterward, $\mathcal{CS}_1$ sends $\{\mathsf{E}(\tilde{\mathtt{x}}_i^j + r_i^j) | i \in [1, w], j \in [1, d]\}$ to $\mathcal{CS}_2$ and r to the query user u_i. For $\mathcal{CS}_2$, it will recover $\{\tilde{\mathtt{x}}_i^j + r_i^j | i \in [1, w], j \in [1, d]\}$ with the secret key sk of SHE and returns them to u_i.

4) Local Diagnosis: After receiving r and $\{\tilde{\mathtt{x}}_i^j + r_i^j | i \in [1, w], j \in [1, d]\}$, u_i first computes $\{r_i^j = \mathsf{H}(r, i||j) \mid i \in [1, w], j \in [1, d]\}$. Then, it is easy for u_i to remove the random noises from $\{\tilde{\mathtt{x}}_i^j + r_i^j | i \in [1, w], j \in [1, d]\}$. Based on the recovered medical data (including diagnostic results), the doctor u_i can have a primary diagnosis for the patient p_i.

V. Security Analysis

In this section, we analyze the security of our dynamic skyline query. Following our design goal, we focus on the privacy preservation of the proposed scheme. Since our proposed scheme is based on SMINn and SDD protocols, in which the SDD protocol adopts our secure atomic protocols as building blocks, we first analyze the privacy preservation of the atomic protocols, SMINn, and SDD protocols.

A. Atomic Protocols Are Privacy Preserving

Our designed SHE-based atomic protocols: SLESS and SEQ, should guarantee that the plaintexts of inputs $\{m_1, m_2\}$, output δ, and the relations of inputs $\{m_1 < m_2, m_1 = m_2\}$ are kept secret from $\mathcal{CS}_1$ and $\mathcal{CS}_2$.

For $\mathcal{CS}_1$: In these protocols, $\mathcal{CS}_1$ holds the inputs and output that are encrypted by SHE. Since SHE has been proven to be semantically secure against IND-CPA in [10], $\mathcal{CS}_1$ will not know the plaintexts of inputs and output without the secret key sk of SHE. In the process of executing these protocols, $\mathcal{CS}_1$ receives $\mathsf{E}(\theta)$ and obtains the relations of inputs by

$$\mathsf{E}(\delta) = \begin{cases} \mathsf{E}(\theta), & \text{if } s = 1 \\ \mathsf{E}(1-\theta), & \text{if } s = -1. \end{cases}$$

Since $\mathcal{CS}_1$ does not have sk, the security of SHE ensures that $\mathcal{CS}_1$ cannot know whether $m_1 < m_2$ in SLESS and $m_1 = m_2$ in SEQ. Therefore, the inputs, output, and the relations of inputs are kept secret from $\mathcal{CS}_1$.

For $\mathcal{CS}_2$: It can recover α using the authorized sk, i.e.,

$$\alpha = s \cdot r_1 \cdot (m_1 - m_2) + s \cdot r_2 \quad \text{in SLESS}$$
$$\alpha = s \cdot r_1 \cdot (m_1 - m_2)^2 - s \cdot r_2 \quad \text{in SEQ.}$$

However, the difference between m_1 and m_2 is protected by two random numbers r_1 and r_2. Therefore, $\mathcal{CS}_2$ cannot infer m_1 and m_2 from α in both protocols. Regarding the relation of inputs, although $\mathcal{CS}_2$ can determine whether $\alpha < 0$ and let $\theta = 1$ (otherwise, $\theta = 0$), we employ the flip-coin mechanism to make $\mathcal{CS}_2$ unable to learn whether $\alpha < 0$ means $m_1 < m_2$ (in SLESS) or $m_1 = m_2$ (in SEQ). As a result, the inputs, output, and the relations of inputs are kept secret from $\mathcal{CS}_2$. Besides, there is no collusion between $\mathcal{CS}_1$ and $\mathcal{CS}_2$. Thus, our designed SHE-based atomic protocols: SLESS and SEQ, are privacy preserving.

B. SMINn and SDD Protocols Are Privacy Preserving

As for the SMINn and SDD protocols, $\mathcal{CS}_1$ has the SHE encrypted inputs and outputs, while $\mathcal{CS}_2$ holds the secret key sk. Since $\mathcal{CS}_1$ does not have the secret key of SHE, and there is no collusion between $\mathcal{CS}_1$ and $\mathcal{CS}_2$, $\mathcal{CS}_1$ cannot know the plaintexts of inputs and outputs from their ciphertexts in static, which is guaranteed by the security of SHE. Therefore, in the following parts, we only analyze the privacy preservation of these protocols when they are running.

1) SMINn Protocol Is Privacy Preserving: Our SMINn protocol should guarantee that neither $\mathcal{CS}_1$ nor $\mathcal{CS}_2$ knows the plaintexts of $\{\mathsf{E}(m_{\min}), \mathsf{E}(m_i) \mid i \in [1, n]\}$ or the information about which item in $\{\mathsf{E}(m_i) \mid i \in [1, n]\}$ is the minimum one.

For $\mathcal{CS}_1$: It obtains the minimum value $\mathsf{E}(m_{\min})$ by recursively computing $\mathsf{E}(m_s) = \sum_{j=0}^{2^k-1}(\mathsf{E}(\delta_j) \cdot \mathsf{E}(m'_j))$ for each group, where $\mathsf{E}(m'_j)$ is generated by dividing $\{\mathsf{E}(m_i) \mid i \in [1, n]\}$ into groups, and $\mathsf{E}(\delta_j)$ is calculated by the privacy-preserving XNOR gate. First, since $\mathcal{CS}_1$ processes $\{\mathsf{E}(m_i) \mid i \in [1, n]\}$ in their ciphertexts, it cannot know m'_j and m_i without sk. Second, recalling Section IV-A1, our XNOR gate can guarantee the operator ($\mathcal{CS}_1$) has no idea on its output $\mathsf{E}(\delta_j)$. That is because the output $\mathsf{E}(\delta_j)$ is homomorphically computed from the encrypted input (see (2)). Consequently, $\mathcal{CS}_1$ cannot know the minimum value m_s in a group, and thus, the final minimum value $m_{\min}$ is kept secret from $\mathcal{CS}_1$. Hence, the plaintexts of $\{\mathsf{E}(m_{\min}), \mathsf{E}(m_i) \mid i \in [1, n]\}$ are kept secret from $\mathcal{CS}_1$. In addition, since $\mathcal{CS}_1$ receives the encrypted index bit sequence, i.e., $\{\mathsf{E}(b_\phi[i]) \mid i \in [0, k-1]\}$, it cannot infer which item is minimum in the group. Thus, $\mathcal{CS}_1$ does not know which item in $\{\mathsf{E}(m_i) \mid i \in [1, n]\}$ is the minimum one.

For $\mathcal{CS}_2$: It can recover $\{\widetilde{m}_j = r_{2^k} \cdot m'_j + r_j \mid j \in [0, 2^k - 1]\}$ using the authorized sk and construct a system of equations

$$\begin{cases} \widetilde{m}_0 = r_{2^k} \cdot m'_0 + r_0 \\ \widetilde{m}_1 = r_{2^k} \cdot m'_1 + r_1 \\ \cdots \qquad \cdots \\ \widetilde{m}_{2^k-1} = r_{2^k} \cdot m'_{2^k-1} + r_{2^k-1}. \end{cases}$$

Since the above system has $2^k + 1$ random numbers, $\mathcal{CS}_2$ cannot solve the system to obtain $\{m'_j \mid j \in [0, 2^k - 1]\}$. Furthermore, in the last group, although $\mathcal{CS}_2$ can obtain $\widetilde{m}_{\min}$, the actual minimum value is perturbed by r_{2^k} and a corresponding random number. As a result, the plaintexts of $\{\mathsf{E}(m_{\min}), \mathsf{E}(m_i) \mid i \in [1, n]\}$ are kept secret from $\mathcal{CS}_2$. Besides, before sending group's members to $\mathcal{CS}_2$, $\mathcal{CS}_1$ adopts the permutation technique to permute them. Therefore, $\mathcal{CS}_2$ learns nothing about which item is selected as the minimum value. Thus, the SMINn protocol is privacy preserving.

2) SDD Protocol Is Privacy Preserving: Our SDD protocol should guarantee that the input plaintexts $\{\mathtt{t}_1^i = (\mathtt{x}_1^i - \mathtt{q}^i)^2, \mathtt{t}_2^i = (\mathtt{x}_2^i - \mathtt{q}^i)^2 \mid i \in [1, d]\}$, output δ, single-dimensional privacy, and the dominance relation of inputs are kept secret from $\mathcal{CS}_1$ and $\mathcal{CS}_2$. Note that in this article, the single-dimensional privacy indicates each dimension's order and equality relations of two given data records.

For $\mathcal{CS}_1$: It always processes inputs over their ciphertexts and exploits the homomorphic properties of SHE to compute the final output $\mathsf{E}(\delta)$. Consequently, $\mathcal{CS}_1$ cannot know the input plaintexts, output plaintext, and the dominance relation. Since $\mathcal{CS}_1$ receives the encrypted $\mathsf{E}(\theta_2)$, and the output of the SEQ protocol $\mathsf{E}(\delta_1)$ is also encrypted, $\mathcal{CS}_1$ neither knows θ_2 nor whether $\theta_1 = \theta_2$ (it is guaranteed by the SEQ protocol). Therefore, $\mathcal{CS}_1$ learns nothing about the single-dimensional privacy of inputs.

For $\mathcal{CS}_2$: It can first recover $\{\alpha^i = s^i \cdot r_i^i \cdot (\mathtt{t}_1^i - \mathtt{t}_2^i) - s_i \cdot r_2^i \mid i \in [1, d]\}$. However, the difference between $\mathtt{t}_1^i$ and $\mathtt{t}_2^i$ is protected by two random numbers. Therefore, $\mathcal{CS}_2$ cannot infer $\{\mathtt{t}_1^i, \mathtt{t}_2^i\}$. Meanwhile, the privacy preservation of SLESS and SEQ guarantees that $\mathcal{CS}_2$ cannot know the actual value of δ_1 and δ_2 due to the flip-coin mechanism. As a result, $\mathcal{CS}_2$ has no knowledge about the output plaintext and the dominance relation. For the single-dimensional privacy, since we integrate the flip-coin mechanism to compute α^i, $\mathcal{CS}_2$ learns nothing about the single-dimensional privacy of inputs. Thus, the SDD protocol is privacy preserving.

C. Proposed Scheme Is Privacy Preserving

Our proposed scheme should guarantee that the outsourced data sets, query requests, and query results are kept secret from $\mathcal{CS}_1$ and $\mathcal{CS}_2$. Meanwhile, our proposed scheme should hide access patterns. In our scheme, the outsourced data sets and query requests are used to retrieve dynamic skyline.

For $\mathcal{CS}_1$: From Algorithm 1, we can see that all of the operations in $\mathcal{CS}_1$ are conducted over SHE ciphertexts. Therefore, $\mathcal{CS}_1$ cannot obtain the plaintexts of outsourced data sets, query requests, and query results without sk. Note that in traversing PM-tree, the privacy-preserving XNOR gate guarantees the minimum value flags are kept secret from $\mathcal{CS}_1$ as we discussed in Section V-B. Besides, since the dynamic skyline points are computed from all encrypted data points $\{\mathsf{E}(\mathtt{x}_i) \mid i \in [1, n]\}$ by applying homomorphic properties, $\mathcal{CS}_1$ does not know which items are selected as skyline points, i.e., the access pattern is protected in $\mathcal{CS}_1$.

For $\mathcal{CS}_2$: It can only touch information through the process of performing SMINn and SDD protocols, and we have shown that both the protocols are privacy preserving. Therefore, $\mathcal{CS}_2$ cannot know the plaintexts of outsourced data sets, query requests, and query results in retrieving dynamic skyline. Besides, the security of the SMINn protocol guarantees that $\mathcal{CS}_2$ learns nothing about which items are selected as the skyline point, i.e., access pattern is hidden. After retrieving skyline points, $\mathcal{CS}_1$ sends $\{\mathsf{E}(\tilde{\mathtt{x}}_i^j + r_j^j) | i \in [1, w], j \in [1, d]\}$ to $\mathcal{CS}_2$, and $\mathcal{CS}_2$ can recover $\{\tilde{\mathtt{x}}_i^j + r_i^j\}$. However, each $\tilde{\mathtt{x}}_i^j$ is protected by a random number r_i^j. As a result, $\mathcal{CS}_2$ cannot infer $\tilde{\mathtt{x}}_i^j$ from $\tilde{\mathtt{x}}_i^j + r_i^j$. Thus, the proposed scheme is privacy preserving.

VI. Performance Evaluation

In this section, we evaluate the performance of our proposed scheme and compare it with the alternative scheme in terms of computational and communication costs. Recalling Section IV-C, the data outsourcing and local diagnosis phases only involve data encryption and noise removal, respectively, which are efficient and intuitive in terms of performance. Therefore, in this section, we mainly focus on the performance of the dynamic skyline search phase.

Experimental Setting: Our scheme was implemented in Java and executed on a machine with 16-GB memory, 3.4-GHz Intel Core i7-3770 processors, and Ubuntu 16.04 OS. In our experiments, we adopt a real-world data set about eye state [19]. In total, there are about 14 000 data records, and each record has 14 attributes. Since the maximum difference of these attributes is less than 500, we set $\mathtt{MAX} = 500^2 \cdot d \cdot (n + 1)$, where d and n are the number of dimensions and data records used in our experiments, respectively. In addition, we set $k = 2$ in the SMINn protocol, i.e., there are four members in a group. For security parameters of SHE, we let $k_0 = 8192, k_1 = 40, k_2 = 160$. The maximum multiplicative depth of our scheme is $\lceil \log n \rceil + 5 = 19$, which is less than $\sigma = \lfloor (k_0/2k_2) - 1 \rfloor = 24$. Therefore, we can avoid applying

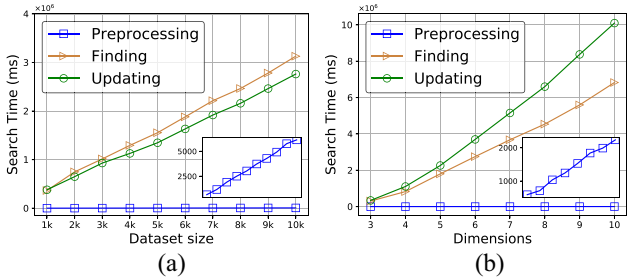


Fig. 5. Computational costs of dynamic skyline search. (a) Varying with the number of data records n, $d = 3$. (b) Varying with the number of dimensions d, $n = 1000$.

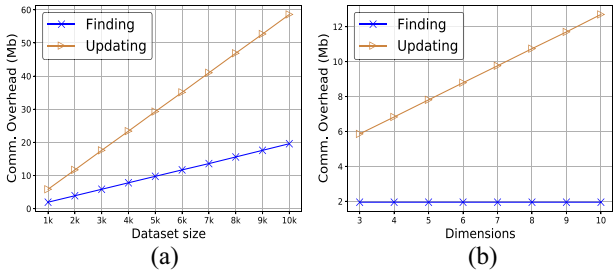


Fig. 6. Communication costs of dynamic skyline search. (a) Varying with the number of data records n, $d = 3$. (b) Varying with the number of dimensions d, $n = 1000$.

the bootstrapping protocol. Note that under such a setting of SHE, our proposed scheme can support a data set containing tens of millions of data records, which is enough for most data sets. For the case that the data set size is increasing, the bootstrapping protocol [10] is recommended if the data set size exceeds the maximum limit. This is because it avoids resetting these security parameters and thus, reencrypting all data records.

A. Performance of Our Proposed Scheme

In this section, we explore the performance of our proposed scheme in searching dynamic skyline, which consists of three phases: 1) preprocessing; 2) finding one skyline point; and 3) updating sum values. From Algorithm 1, we can see that the performance of our scheme is related to the number of data records n and dimensions d. Therefore, in the following, we will show the computational and communication costs varying with n and d in these three phases.

1) Computational Costs of Searching Skyline: Fig. 5 depicts the computational cost of our scheme in searching dynamic skyline, in which we vary the number of data records n from 1000 to 10 000 and the number of dimensions d from 3 to 10. Both Fig. 5(a) and (b) shows that the computational costs of these phases are linearly increasing with the corresponding parameters. However, the computational cost of the preprocessing phase is significantly less than that of finding one skyline point and updating sum values phases, as shown in Fig. 5(a) and (b). It is reasonable since our scheme performs the preprocessing phase only once, while the last two phases need to repeat w (the number of skyline points) times. Fig. 5(a) shows that the finding one skyline point phase takes more time than the updating sum values phase when the growth of n. That is because, in the finding phase, $\mathcal{CS}_1$ needs to traverse the PM-tree, whose height is related to n. In Fig. 5(b), the updating sum values phase is more expensive that the finding one skyline point phase. In fact, to ensure the security of our scheme, we employ the flip-coin mechanism in the SDD protocol, in which $\mathcal{CS}_1$ needs to choose two random numbers for each dimension. Thus, the third phase is more expensive when d is increasing.

2) Communication Costs of Searching Skyline: Fig. 6 plots the communication cost of searching dynamic skyline, in which Fig. 6(a) varies the number of data records n, while Fig. 6(b) varies the number of dimensions d. Since there is no interaction between $\mathcal{CS}_1$ and $\mathcal{CS}_2$ in the preprocessing phase, the communication overhead in this phase is always 0. Thus, we do not show the evaluation of the preprocessing phase in Fig. 6. For the finding one skyline point phase, Fig. 6(a) shows that the communication cost of this phase is linearly increasing with n, while Fig. 6(b) demonstrates that it has nothing to do with d. That is because there are $\lceil([n-1]/[2^k-1])\rceil \cdot (2^k + k)$ SHE ciphertexts in this phase. For the third phase, the communication cost is introduced by the SDD protocol. We know that our SDD protocol needs $(d + 5)$ SHE ciphertexts. Therefore, for each skyline search round, there are $n(d + 5)$ SHE ciphertexts. Thus, both Fig. 6(a) and (b) presents a linear trend. Note that since we set $k = 2$, $n(d + 5) > \lceil([n-1]/[2^k-1])\rceil \cdot (2^k + k) \approx 2(n-1)$. As a result, in both figures, the updating sum values phase is more expensive than the finding one skyline point phase in the communication cost.

B. Comparison With Alternative Scheme

In this section, we compare our scheme with Liu *et al.*'s scheme (denoted as LYXP17) [4], which also designed a secure dynamic skyline query scheme without leaking plaintexts, single-dimensional privacy, and access patterns. To the best of our knowledge, LYXP17 is the only scheme that has the same security level as our proposed scheme in searching dynamic skyline. Since LYXP17 employed the Paillier encryption as the primitive cryptosystem and the protocols introduced in [20] as their basic secure protocols, the scheme would take a lot of time when the data volume is large. To reasonably reduce the evaluation time, we set the security parameter of Paillier as the lowest level, i.e., 512, and the number of data records from 100 to 1000. Note that our scheme can also adopt the data partitioning technique in [4] to further improve performance. To clearly present the improvement of our proposed scheme, we compare our scheme with LYXP17 without using this optimization technique.

1) Comparing Computational Costs in Searching Skyline: Fig. 7 depicts the comparison results of our scheme and LYXP17 in the computational cost. In Fig. 7(a), we vary the number of data records n from 100 to 1000, while it is 3 to 10 for the number of dimensions d in Fig. 7(b). We can see that both the figures show our scheme achieves up to two orders of magnitude better performance than LYXP17. There are two reasons for such a big improvement. First, SHE is much more

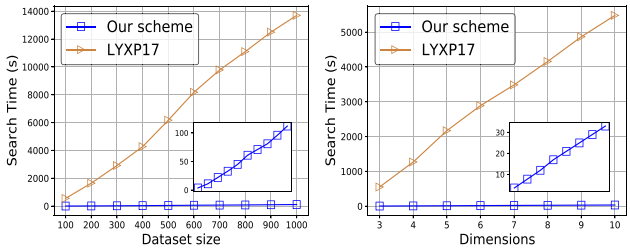


Fig. 7. Comparison of computational costs in searching dynamic skyline. (a) Varying with the number of data records n, $d = 3$. (b) Varying with the number of dimensions d, $n = 100$.

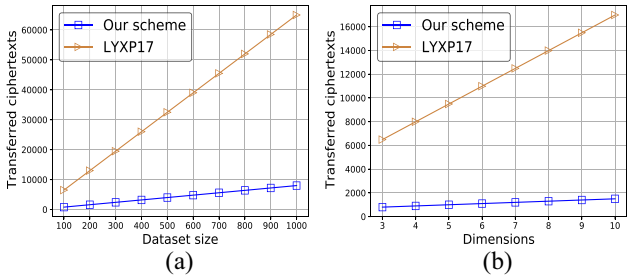


Fig. 8. Comparison of communication costs in searching dynamic skyline. (a) Varying with the number of data records n, $d = 3$. (b) Varying with the number of dimensions d, $n = 100$.

TABLE II
THEORETICAL ANALYSIS OF THE COMMUNICATION COSTS

Phases	**Our scheme**	**LYXP17**
Preprocessing	0	$3nd$
Finding one skyline point	$2(n-1)$	$6nd + 2n + 8(n-1)$
Updating sum values	$n(d+5)$	$5nd + 8n$

efficient that Paillier in encryption and decryption. Second, we design a set of protocols: the secure atomic protocols, and SMINn and SDD protocols, which are more efficient than the secure protocols used in LYXP17.

2) Comparing Communication Costs in Searching Skyline: Since both our scheme and LYXP17 can adjust the security parameters to change the communication overheads, to be fair, we instead compare the number of ciphertexts transferred between $\mathcal{CS}_1$ and $\mathcal{CS}_2$. Fig. 8(a) shows that our scheme is better than LYXP17 in the communication cost around $8.1\times$, while it is at least $8.1\times$ when varying the number of dimensions, as shown in Fig. 8(b). When $d = 10$, our scheme is about $11.3\times$ better than LYXP17. To clearly explain the reasons, we present the theoretical analysis of communication costs in Table II.

From Table II, we can see that our scheme has fewer communication costs in all phases. To ease of evaluation, we assume the secure comparison protocols in LYXP17, such as SLESS and SEQ, have only one communication round and two transferred ciphertexts, which is the same as our comparison protocols. In fact, the comparison protocols used in LYXP17 have more communication rounds and transferred ciphertexts. Therefore, our scheme is at least $8.1\times$ better than LYXP17 in the communication cost.

VII. RELATED WORK

With increasing awareness of privacy protection and the wide application of skyline queries, there has been a lot of research devoted to processing skyline queries over encrypted data [2], [4], [6], [11]–[13]. Bothe *et al.* [11] transformed the skyline computation into a nondescending series and adopted matrix encryption as the primitive cryptosystem. This scheme is not semantically secure, and an adversary can launch a known plaintext attack to determine the secret keys. Zheng *et al.* [6] proposed a noninteractive data comparison protocol to determine dominance relations over encrypted data. However, this work mainly focused on the secure data merging technique and did not consider the single-dimensional privacy and access patterns in computing skyline. Based on the bloom filter technique and a variant of ElGamal encryption, Zhang *et al.* [12] presented a secure probabilistic skyline query scheme for work selection in the mobile crowdsensing scenario. This work is to deal with the probabilistic skyline query rather than the dynamic skyline query that is used in our scheme. Meanwhile, although the proposed scheme [12] can hide the difference between two messages, it cannot preserve the single-dimensional privacy and hide access patterns. In the eHealthcare scenario, Hua *et al.* [2] implemented a privacy-preserving online primary diagnosis system with skyline query. In this scheme, the service provider holds two skyline point sets, in which the elements are encoded, permuted, and perturbed. When a user (patient) would like to enjoy the online primary diagnosis, he/she interacts with the service provider to determine whether the sensed data record dominates the points in these two sets. Different from our scheme that uses the dynamic skyline query, the work in [2] adopted the basic skyline computation to make the online primary diagnosis.

Recently, the works in [4], [13], [21], and [22] separately proposed schemes to achieve secure dynamic skyline queries. Wang *et al.* [13] employed the ORE to encrypt the outsourced data. Obviously, it leaked the order relations for a single dimension and also did not hide access patterns. Wang *et al.* [21] presented a secure skyline query scheme under the secure hardware (SGX) model instead of the semi-honest two-server model. Besides, this approach leaks the access patterns that are protected in our work. Zeighami *et al.* [22] proposed a scheme to use query result materialization for answering dynamic skyline queries on encrypted data. However, it focused on building the result materialization structure in a reasonable time while keeping the storage overhead at practical levels, which cannot address the problems in this work. Liu *et al.* [4] designed a fully secure dynamic skyline scheme that can preserve the privacy of plaintexts, single-dimensional privacy, and access patterns. However, this scheme is not efficient enough. Our proposed scheme can achieve the same security level as [4] while significantly reducing the computational and communication costs.

VIII. CONCLUSION

In this article, we have proposed an efficient and privacy-preserving dynamic skyline query scheme that allows doctors

to make a similar search over encrypted data in the online medical diagnosis system. Specifically, we first defined the dynamic skyline query and introduced the SHE. Then, we presented two secure atomic protocols, i.e., SLESS and SEQ, in which we employed the flip-coin mechanism to ensure security. Based on these atomic protocols, we designed an SDD protocol to securely determine dominance relations. Meanwhile, we proposed a privacy-preserving XNOR gate and a private minimum tree (PM-tree) to construct our SMINn protocol. With all of the above protocols, we carefully designed our dynamic skyline query scheme to preserve privacy while achieving high efficiency. Finally, we analyzed the security of our scheme and conducted extensive experiments to evaluate it. The results show that our scheme is efficient in both computation and communication.

REFERENCES

[1] H. Habibzadeh, K. Dinesh, O. R. Shishvan, A. Boggio-Dandry, G. Sharma, and T. Soyata, "A survey of Healthcare Internet of Things (HIoT): A clinical perspective," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 53–71, Jan. 2020.

[2] J. Hua *et al.*, "CINEMA: Efficient and privacy-preserving online medical primary diagnosis with skyline query," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1450–1461, Apr. 2019.

[3] S. Ivanov, C. Foley, S. Balasubramaniam, and D. Botvich, "Virtual groups for patient WBAN monitoring in medical environments," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 11, pp. 3238–3246, Nov. 2012.

[4] J. Liu, J. Yang, L. Xiong, and J. Pei, "Secure skyline queries on cloud platform," in *Proc. IEEE ICDE*, 2017, pp. 633–644.

[5] D. Papadias, Y. Tao, G. Fu, and B. Seeger, "An optimal and progressive algorithm for skyline queries," in *Proc. SIGMOD*, 2003, pp. 467–478.

[6] Y. Zheng, R. Lu, B. Li, J. Shao, H. Yang, and K.-K. R. Choo, "Efficient privacy-preserving data merging and skyline computation over multi-source encrypted data," *Inf. Sci.*, vol. 498, pp. 91–105, Sep. 2019.

[7] E. Shi, J. Bethencourt, T. H. Chan, D. Song, and A. Perrig, "Multi-dimensional range query over encrypted data," in *Proc. IEEE Symp. Security Privacy*, 2007, pp. 350–364.

[8] R. Li and A. X. Liu, "Adaptively secure conjunctive query processing over encrypted data for cloud computing," in *Proc. IEEE ICDE*, 2017, pp. 697–708.

[9] C. Guo, R. Zhuang, C. Su, C. Z. Liu, and K.-K. R. Choo, "Secure and efficient k nearest neighbor query over encrypted uncertain data in cloud-IoT ecosystem," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9868–9879, Dec. 2019.

[10] Y. Zheng, R. Lu, Y. Guan, J. Shao, and H. Zhu, "Efficient and privacy-preserving similarity range query over encrypted time series data," *IEEE Trans. Depend. Secure Comput.*, early access, Feb. 23, 2021, doi: 10.1109/TDSC.2021.3061611.

[11] S. Bothe, A. Cuzzocrea, P. Karras, and A. Vlachou, "Skyline query processing over encrypted data: An attribute-order-preserving-free approach," in *Proc. PSBD@CIKM*, 2014, pp. 37–43.

[12] X. Zhang, R. Lu, J. Shao, H. Zhu, and A. A. Ghorbani, "Secure and efficient probabilistic skyline computation for worker selection in MCs," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11524–11535, Dec. 2020.

[13] W. Wang *et al.*, "Scale: An efficient framework for secure dynamic skyline query processing in the cloud," in *Proc. Int. Conf. Database Syst. Adv. Appl.*, 2020, pp. 288–305.

[14] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Access pattern disclosure on searchable encryption: ramification, attack and mitigation," in *Proc. NDSS*, 2012, pp. 1–9.

[15] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *Proc. SIGSAC*, 2015, pp. 644–655.

[16] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, "Machine learning classification over encrypted data," in *Proc. NDSS*, 2015, p. 4325.

[17] H. Mahdikhani, R. Lu, Y. Zheng, J. Shao, and A. A. Ghorbani, "Achieving $O(\log^3 n)$ communication-efficient privacy-preserving range query in fog-based IoT," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5220–5232, Jun. 2020.

[18] Y. Guan, R. Lu, Y. Zheng, S. Zhang, J. Shao, and G. Wei, "Toward privacy-preserving cybertwin-based spatio-temporal keyword query for its in 6G era," *IEEE Internet Things J.*, early access, Jul. 12, 2021, doi: 10.1109/JIOT.2021.3096674.

[19] O. Roesler. (2019). *EEG Eye State*. [Online]. Available: https://datahub.io/machine-learning/eeg-eye-state

[20] T. Veugen, F. Blom, S. J. de Hoogh, and Z. Erkin, "Secure comparison protocols in the semi-honest model," *IEEE J. Sel. Topics Signal Process.*, vol. 9, no. 7, pp. 1217–1228, Oct. 2015.

[21] J. Wang, M. Du, and S. S. Chow, "Stargazing in the dark: Secure skyline queries with SGX," in *Proc. Int. Conf. Database Syst. Adv. Appl.*, 2020, pp. 322–338.

[22] S. Zeighami, G. Ghinita, and C. Shahabi, "Secure dynamic skyline queries using result materialization," in *Proc. IEEE 37th Int. Conf. Data Eng. (ICDE)*, 2021, pp. 157–168.

Songnian Zhang received the M.S. degree from Xidian University, Xi'an, China, in 2016. He is currently pursuing the Ph.D. degree with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada.

His research interest includes cloud computing security, big data query, and query privacy.

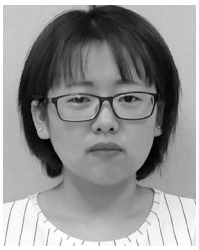
Suprio Ray (Member, IEEE) received the Ph.D. degree from the Department of Computer Science, University of Toronto, Toronto, ON, Canada, in 2015.

He is an Associate Professor with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada. His research interests include big data and database management systems, runtime systems for scalable data science, provenance and privacy issues in big data, and query processing on modern hardware.

Rongxing Lu (Fellow, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012.

He is an Associate Professor and a University Research Scholar with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada. Before that, he worked as an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, from April 2013 to August 2016. He worked as a Postdoctoral Fellow with the University of Waterloo from May 2012 to April 2013.

Dr. Lu was awarded the most prestigious Governor General's Gold Medal in 2012 and the 8th IEEE Communications Society Asia Pacific Outstanding Young Researcher Award in 2013.



Yandong Zheng received the M.S. degree from the Department of Computer Science, Beihang University, Beijing, China, in 2017. She is currently pursuing the Ph.D. degree with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada.

Her research interest includes cloud computing security, big data privacy, and applied privacy.

Yunguo Guan is currently pursuing the Ph.D. degree with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada.

His research interests include applied cryptography and game theory.

Jun Shao (Member, IEEE) received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2008.

He was a Postdoctoral 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.