Hindawi
Security and Communication Networks
Volume 2021, Article ID 2313389, 15 pages
https://doi.org/10.1155/2021/2313389

Research Article

An Efficient and Privacy-Preserving Biometric Identification Scheme Based on the FITing-Tree

Xiaopeng Yang,[1] **Hui Zhu**,[1] **Songnian Zhang**,[2] **Rongxing Lu**,[2] **and Xuesong Gao**[3]

[1]*The State Key Laboratory of Integrated Network Service, Xidian University, Xi'an 710126, China*
[2]*The Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada*
[3]*The State Key Laboratory of Digital Multimedia Technology, Hisense, Qingdao 266061, China*

Correspondence should be addressed to Hui Zhu; zhuhui@xidian.edu.cn

Received 24 June 2021; Revised 12 September 2021; Accepted 28 September 2021; Published 25 October 2021

Academic Editor: Luigi Catuogno

Biometric identification services have been applied to almost all aspects of life. However, how to securely and efficiently identify an individual in a huge biometric dataset is still very challenging. For one thing, biometric data is very sensitive and should be kept secure during the process of biometric identification. On the other hand, searching a biometric template in a large dataset can be very time-consuming, especially when some privacy-preserving measures are adopted. To address this problem, we propose an efficient and privacy-preserving biometric identification scheme based on the FITing-tree, iDistance, and a symmetric homomorphic encryption (SHE) scheme with two cloud servers. With our proposed scheme, the privacy of the user's identification request and service provider's dataset is guaranteed, while the computational costs of the cloud servers in searching the biometric dataset can be kept at an acceptable level. Detailed security analysis shows that the privacy of both the biometric dataset and biometric identification request is well protected during the identification service. In addition, we implement our proposed scheme and compare it to a previously reported M-Tree based privacy-preserving identification scheme in terms of computational and communication costs. Experimental results demonstrate that our proposed scheme is indeed efficient in terms of computational and communication costs while identifying a biometric template in a large dataset.

1. Introduction

With the booming development of the Internet of Things (IoT), the number of smart devices, such as smart cameras and smartwatches, has grown dramatically in recent years. According to the reports, there have been 12 billion IoT devices in 2020, and this amount will grow to more than 30 billion in 2025 [1]. The proliferation of IoT devices and the development of image processing technologies make biometric-based services increasingly easy to deploy and reliable. As a result, biometric-based services have been applied to a variety of scenarios including airport service, criminal investigation, and counterterrorism [2–5].

As a major type of biometric-based services, biometric identification aims to find whether a given biometric template exists in a precollected biometric dataset. Specifically, a biometric template is usually denoted by a vector, e.g., an l-dimension vector $T = \{t_1, t_2, \ldots, t_l\}$. For a given biometric dataset $\mathcal{T}$, T exists in $\mathcal{T}$, which means that there exists a biometric template $T' = \{t_1', t_2', \ldots, t_l'\} \in \mathcal{T}$, which makes the Euclidean distance $\mathrm{dis}(T, T')$ less than or equal to a given threshold δ, i.e., $\mathrm{dis}(T, T') = \sqrt{\sum_{i=1}^{l} (t_i - t_i')^2} \leq \delta$. Since the biometric template dataset owner may be limited in computing power and storage capacity, it is necessary to outsource the biometric dataset to the cloud for data management and biometric identification request processing when the dataset becomes large. However, as biometric data is crucially sensitive, and the cloud server is not always trusted, some measures should be adopted to prevent the cloud from extracting private information from the outsourced data. It is well accepted that encryption is the most intuitive solution to this issue. But we should pay attention to the fact that, in addition to privacy, data utility should also be guaranteed, which means that the similarity between two

templates should be able to be derived from their encrypted data.

To solve this problem, researchers have proposed many schemes [6–13] to achieve privacy-preserving biometric recognition. Unfortunately, most of these schemes [6–10] work in a basic way, which means that they just traverse the whole dataset to get the identification result, and no optimization tactics are taken to accelerate the searching process. Consequently, a huge computing burden is brought to the cloud when it handles too many identification requests simultaneously, which makes these schemes inefficient and unpractical. The situation turns worse when the dataset size grows. As a result, it is urgent to design a privacy-preserving biometric identification scheme by which both the security of the biometric template and the efficiency of the identification process can be guaranteed. Some schemes [11] employ some data structures to expedite the searching process. However, the data owner has to be on standby during the identification service, which results in the loss of the native advantages of cloud computing.

In this paper, we propose an efficient and privacy-preserving biometric identification scheme based on two data structures, namely, the FITing-tree [14] and iDistance [15], and a symmetric homomorphic encryption (SHE) algorithm [16,17]. With our proposed scheme, the privacy of the biometric dataset and the identification request is preserved. In addition, the computational costs of the cloud, which are used to process the identification requests, are kept at an acceptable level. Specifically, the main contributions of this paper are threefold.

(i) First, we propose a privacy-preserving biometric identification scheme with two noncollusive cloud servers. With our proposed scheme, the privacy of the biometric dataset and the identification request is protected during the online biometric identification service process. Specifically, the cloud servers cannot obtain the specific value of the identification request and the plaintext of the biometric template in the dataset.

(ii) Second, the efficiency of the proposed scheme is improved with the FITing-tree and iDistance. By introducing the FITing-tree, the computational costs of the biometric identification process are significantly lowered by reducing the number of similarity comparison operations. Besides, the size of the index is also optimized.

(iii) Third, to evaluate the performance of our proposed scheme, we implement our proposed scheme and conduct extensive experiments on a synthetic dataset. Both the theoretical and experimental results show that the proposed scheme is more efficient than other similar schemes in both computational and communication costs. In addition, we also test the accuracy of our proposed scheme on a real-world face dataset.

The remainder of the paper is organized as follows. In Section 2, we review some related work at first. Then, we formalize our system model and security model and identify our design goal in Section 3 and review some preliminaries, including a SHE scheme, FaceNet algorithm, iDistance data structure, and FITing-tree data structure in Section 4. After that, we present our proposed scheme in Section 5, followed by security analysis and performance evaluation in Section 6 and Section 7, respectively. Finally, we draw our conclusion in Section 8.

2. Related Work

In this section, we will briefly review some related work on privacy-preserving biometric identification.

Early privacy-preserving biometric identification schemes only focus on the privacy-preserving issue. In these schemes, the biometric identification scheme is considered to be a two-party system, where the data owner takes charge of biometric dataset management and template matching. Most of these schemes are designed based on the secure computation protocol [18–20] and homomorphic encryption [9,21,22] techniques. Although the privacy-preserving is achieved in these schemes, the data owner is required to be equipped with powerful computing ability and remarkable storage capacity in these schemes, which can hardly be satisfied in most application scenarios and thus makes these schemes unpractical.

The emergence of cloud computing presents a new and promising paradigm to handle these challenges. Some researchers leverage cloud computing techniques to release the data owner from this burden. In their schemes, the data owner outsources the encrypted biometric dataset to the cloud server, and the matching process is completed on the cloud. Yuan et al. [6] proposed the first cloud-based privacy-preserving biometric identification scheme using a matrix encryption scheme, where the biometric dataset and identification query are both encrypted and sent to the cloud server by the data owner. However, Wang et al. [7] and Zhu et al. [23] pointed out that [6] is not secure under the known-plaintext attack model [24]. In addition, [7] presented a privacy-preserving biometric identification scheme based on the similarity matrix under the same system model in [6] and the security analysis showed that [7] had a higher security level than [6]. Zhang et al. [8] proposed an efficient privacy-preserving biometric identification scheme based on the matrix and perturbed terms with lower time cost and bandwidth consumption than [6,7]. Wang et al. [10] proposed an inference-based framework for privacy-preserving similarity search in Hamming space and achieved privacy-preserving biometric identification based on it. Hu et al. [25] proposed a privacy-preserving biometric identification scheme in an outsourcing environment with two non-colluded servers based on homomorphic encryption and batched protocols. With the help of the cloud, the computing cost of the data owner during the biometric matching is significantly reduced in the above schemes. However, in [6–8], the data owner has to keep on online to encrypt the user's query data and decrypt the identification result, which whittles some advantages of cloud computing away and leads heavy load to the data owner if it serves too many users

at the same time. What is more, in all the cloud-based schemes above, the searching process is not optimized, which means that the searching cost of the cloud server is linear with the size of the dataset. Despite the fact that the cloud server is equipped with strong computing power, it may still run into a bottleneck while simultaneously severing too many users.

To address this issue, some researchers begin to focus on how to achieve sublinear searching efficiency in the biometric identification process, which will significantly ease the pressure of the cloud server. Zhu et al. [11] proposed a cloud-assisted privacy-preserving biometric identification scheme. With the help of an asymmetric scalar-product preserving encryption scheme and R-tree, sublinear search efficiency is achieved in [11]. Nevertheless, the data owner also needs to keep online in [11]. And since R-tree is not constructed among the metric relation between the data objects, the cloud server needs to search the tree twice to find the closest biometric template in the dataset, which reduces the efficiency of the searching process. Yang et al. [26] proposed a privacy-preserving biometric identification scheme based on the M-tree to achieve a sublinear search efficiency.

In this paper, to protect the privacy of the biometric data and reduce the time consumption in the biometric searching process, we introduce SHE and FITing-tree to construct a privacy-preserving biometric identification scheme based on two noncolluded cloud servers. Compared with previous works, the service provider in our proposed scheme does not need to keep online in the identification scheme, and higher efficiency in both computation and communication is achieved.

3. Models and Design Goal

In this section, we formally describe our system model and security model and identify our design goals.

3.1. System Model. In our system model, we consider a cloud-based biometric identification system as shown in Figure 1, which mainly consists of three parts: the service provider, a cloud with two servers, and the client.

(i) Service provider: the service provider (SP) has collected a biometric templates dataset $\mathcal{T}$ with n biometric templates, i.e., $\mathcal{T} = \{T_1, T_2, \ldots, T_n\}$, where each template is an l-dimension vector $T_i = \{t_{i1}, t_{i2}, \ldots, t_{il}\}$. For simplicity and clear description, we assume that the value of each t_{ij} $(1 \le j \le l)$ is a positive integer since a nonintegral biometric template can be transformed into a positive integer vector easily. To make the best use of the dataset, the data owner is willing to offer an online biometric identification service to some users. Since the SP usually has limited computing power and storage capability, to relieve the burden of data management and handling a large number of biometric identification requests, it tends to outsource the biometric dataset to the cloud. In consideration of the sensibility of the biometric data, and the fact that the cloud is not always trusted, the biometric data should be encrypted before being outsourced to the cloud.

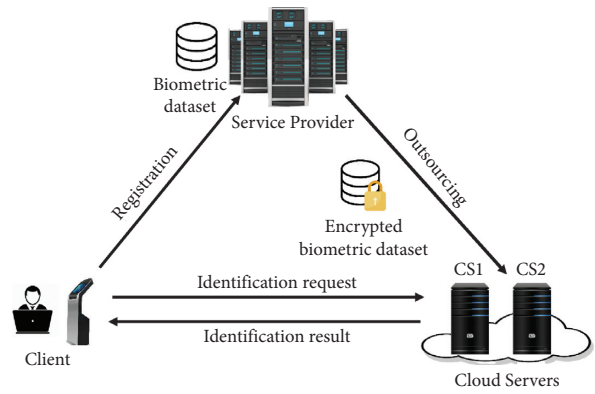

Figure 1: System model under consideration.

(ii) Cloud servers: in our system, the cloud employs two cloud servers, namely, cloud server 1 (CS1) and cloud server 2 (CS2), from two different cloud service providers, to collaboratively process the biometric identification requests. Specifically, CS1 stores the encrypted dataset and indexes and accepts identification requests. CS2 holds the secret key and helps CS1 get the identification result by decrypting some intermediate results. Note that these two cloud servers are powerful in computing and have sufficient storage space.

(iii) Client: the client in our system model can be an IoT device, which is equipped with sensors (e.g., camera, microphone, or fingerprint collector) and has moderate computation ability (e.g., to extract biometric features and encrypt some data). An application employs the client to access the biometric identification service. To get the identification result, the client generates an identification request, submits it to the cloud servers, and processes the response from the cloud servers.

3.2. Security Model. In our system model, we consider that the SP and the client are fully trusted and will honestly follow the prearranged protocol. As for the two cloud servers, CS1 and CS2 are considered to be honest-but-curious, which means they will faithfully follow the protocols by outputting the correct identification result but will be curious about the client's or SP's data once certain conditions are satisfied. Meanwhile, we assume that the two cloud servers do not collude with each other. This is reasonable since the cloud servers should maintain their reputation and interests. Note that since we only focus on how to achieve efficient and privacy-preserving biometric identification in this paper, active attacks on data integrity and source authentication from external adversaries are beyond the scope of our work and will be discussed in our future work.

3.3. Design Goals. Our design goal is to propose an efficient and privacy-preserving cloud-based biometric identification scheme to address the challenges mentioned in the above system model and security model. Specifically, the following two objectives should be achieved:

(i) Privacy: since the biometric data is highly sensitive, the proposed biometric identification scheme should be privacy-preserving, which means that the security of the biometric data stored in the biometric template dataset and identification request should be guaranteed.

(ii) Efficiency: since high time delay is intolerable for a biometric identification system, the proposed biometric scheme should be efficient in terms of both computational and communication costs. The factors causing the inefficiency of the biometric identification system mainly lie in two aspects. First, the cloud servers need to search the whole biometric template dataset at the identification stage, which is quite time-consuming when the template dataset becomes large. Second, to satisfy the privacy-preserving requirements, some additional operations will be necessarily introduced, which significantly increases the computational costs of the cloud servers. Besides, since both the dataset and the identification request are needed to be sent to the cloud servers, it will bring a heavy communication burden to the cloud server while serving too many users simultaneously. Therefore, some measures should be adopted to achieve higher efficiency in computation and communication.

4. Preliminaries

In this section, we briefly review the FaceNet algorithm [27], symmetric homomorphic encryption (SHE) scheme [16], the iDistance data structure [15], and FITing-Tree data structure [14], which will serve as the building blocks of our proposed scheme.

4.1. FaceNet Algorithm. FaceNet [27] is a face recognition system that aims at outputting an embedding by mapping a face image into a compact Euclidean space. With the help of a deep convolutional network, FaceNet works in a two-phases model, i.e., the training phase and the matching phase. In the training phase, given a face image x, a mapping from the face image x to a compact Euclidean space $\mathbb{R}^d$ is built at first. Then, based on the mapping, a Euclidean embedding $f(x) \in \mathbb{R}^d$ can be calculated to represent the face image x. In the matching phase, two face images x and y are given, which will be represented as two embeddings: $f(x) = (x_1, x_2, \ldots, x_d)$ and $f(y) = (y_1, y_2, \ldots, y_d)$. To evaluate the similarity of x and y, the Euclidean distance of the two embeddings $f(x), f(y)$ can be computed as $\mathrm{dis}(f(x), f(y)) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$. Then, a threshold δ is used to determine whether these two faces x, y are the same (denoted as R_{same}) or different (denoted as R_{diff}). The decision process is performed as follows:

$$\begin{cases} (x, y) \in R_{\mathrm{same}}, & \text{if } \mathrm{dis}(f(x), f(y)) \le \delta; \\ (x, y) \in R_{\mathrm{diff}}, & \text{if } \mathrm{dis}(f(x), f(y)) > \delta. \end{cases} \tag{1}$$

4.2. Description of SHE. The SHE [16,17] is a symmetric homomorphic encryption scheme, which mainly consists of three algorithms, namely, (i) key generation, (ii) encryption, and (iii) decryption:

(i) Key generation: given three security parameters (k_0, k_1, k_2), which satisfy the constraint $k_1 \ll k_2 < k_0/2$, then the secret key is generated as $\mathrm{sk} = (p, q, \mathscr{L})$, where p, q are two large prime numbers with $|p| = |q| = k_0$, and $\mathscr{L}$ is a random number with the bit length $|\mathscr{L}| = k_2$. Eventually, compute $\mathscr{N} = pq$ and set the public parameters $\mathrm{PP} = (k_0, k_1, k_2, \mathscr{N})$. The message space of the SHE scheme is $\mathscr{M}$ as $\{0,1\}^{k_1}$.

(ii) Encryption: given a message $m \in \mathscr{M}$, it can be encrypted with the secret key $\mathrm{sk} = (p, q, \mathscr{L})$ as

$$c = E(m) = (r\mathscr{L} + m)(1 + r\prime p) \bmod \mathscr{N}, \tag{2}$$

where $r \in \{0,1\}^{k_2}$ and $r\prime \in \{0,1\}^{k_0}$ are two random numbers.

(iii) Decryption: given a ciphertext $c = E(m)$, it can be decrypted with the secret key $\mathrm{SK} = (p, q, \mathscr{L})$ as

$$m = D(c) = (c \bmod p) \bmod \mathscr{L}. \tag{3}$$

The correctness of the decryption can be proven as follows:

$$\begin{aligned} D(c) &= (c \bmod p) \bmod \mathscr{L} \\ &= (((r\mathscr{L} + m)(1 + r\prime p) \bmod \mathscr{N}) \bmod p) \bmod \mathscr{L} \\ &= (r\mathscr{L} + m) \bmod \mathscr{L} \left(\because 2k_2 < k_0\right) \\ &= m \left(\because k_1 \ll k_2\right). \end{aligned} \tag{4}$$

SHE satisfies the following homomorphic properties:

(i) Homomorphic Addition-I: given two ciphertexts $c_1 = E(m_1) = (r_1\mathscr{L} + m_1)(1 + r_1'p) \bmod \mathscr{N}$ and $c_2 = E(m_2) = (r_2\mathscr{L} + m_2)(1 + r_2'p) \bmod \mathscr{N}$, we have $c_1 + c_2 \longrightarrow E(m_1 + m_2)$

(ii) Homomorphic Multiplication-I: given two ciphertexts c_1, c_2, we have $c_1 \cdot c_2 \longrightarrow E(m_1 \cdot m_2)$

(iii) Homomorphic Addition-II: given a ciphertext c_1 and a plaintext m_2, we have $c_1 + m_2 \longrightarrow E(m_1 + m_2)$

(iv) Homomorphic Multiplication-II: given a ciphertext c_1 and a plaintext m_2, we have $c_1 \cdot m_2 \longrightarrow E(m_1 \cdot m_2)$

4.3. iDistance Data Structure. The iDistance data structure is an indexing and query processing technique for the k-nearest neighbor (kNN) queries on point data in multi-dimensional metric spaces [15]. For a given dataset,

iDistance firstly partitions the data based on a space- or data-partitioning strategy and selects a reference point for each partition. Then, a one-dimensional index is calculated for each data point in one partition based on its distance to the reference point of this partition. Finally, a $B+$ tree is built on these indexes, and the kNN search can be performed using a one-dimensional range search. As shown in Figure 2, the detailed description of the index building process is as follows:

(i) Data partition: the dataset is divided into a set of partitions with some clustering algorithms, e.g., the K-means algorithm. Then, a reference point is assigned for each partition. Suppose that there are m partitions $\{P_1, P_2, \ldots, P_m\}$ whose corresponding reference points are represented by $\{O_1, O_2, \ldots, O_m\}$.

(ii) Index calculation: for a data point p: $\{x_1, \ldots, x_l\} \in P_i$, its index y can be generated based on the distance from its corresponding reference point O_i as follows:

$$y = i \times c + \mathrm{dis}(p, O_i), \tag{5}$$

where c is an offset value used to avoid the overlap between the iDistance range of different partitions. Specifically, c plays a role in splitting the one-dimensional space into regions, and all points in each partition are mapped to different regions. For example, for the ith partition p_i, all data points in this region will be mapped to the range $[i \times c, i \times c + \mathrm{dis}_{\max}(p, O_i))$, where $\mathrm{dis}_{\max}(p, O_i)$ is the greatest distance of all points in P_i from the reference point of P_i. c must be set sufficiently large to avoid the overlap between the index regions of different partitions.

(iii) Range query: the given range query (q, r) aims to find all data points in D that satisfy $\{p \in D | \mathrm{dis}(p, q) \le r\}$. For any data point p in the ith partition P_i, it is straightforward to get the inequality

$$\mathrm{dis}(O_i, q) - \mathrm{dis}(p, q) \le \mathrm{dis}(O_i, p) \le \mathrm{dis}(O_i, q) + \mathrm{dis}(p, q), \tag{6}$$

based on the triangle inequality property of Euclidean distance. With the range query requirement, we have

$$\mathrm{dis}(O_i, q) - r \le \mathrm{dis}(O_i, p) \le \mathrm{dis}(O_i, q) + r, \tag{7}$$

which means that we only need to check data points whose iDistance index lies in $[\mathrm{dis}(O_i, q) - r, \min(\mathrm{dis}(O_i, q) + r, dis_{\max}(p, O_i))]$.

4.4. FITing-Tree Data Structure. The FITing-tree [14] is a data-aware index structure that approximates an index using piece-wise linear functions. With FITing-tree, a given key can be mapped to a storage location with a bounded error.

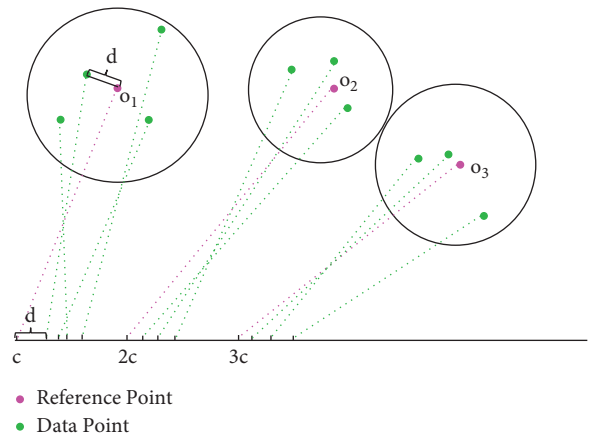

Figure 2: iDistance data structure.

There are two basic data notions in the FITing-tree, namely, the error threshold error and the segment:

(i) Segment: a segment is a line segment that maps a key to its approximate storage position. A segment Seg can be represented by the starting point start and the slope, i.e., Seg = {start, slope}. For a given key x lying in this segment, its predicted position can be calculated by

$$pred_pos = (x - \mathrm{Seg.start}) \times \mathrm{Seg.slope}. \tag{8}$$

(ii) Error: the error threshold error is the maximum distance that the predicted location of any key inside a segment from its actual position.

The operations of the FITing-tree mainly consist of FITing-tree building and query. The detailed description of the FITing-tree building and query process is as follows.

(i) FITing-tree building: the main goal of the FITing-tree building process is to use a series of disjoint linear segments to capture trends that exist in the data with the error threshold satisfied. The dataset is sorted in ascending order at first, and then a greedy streaming algorithm is used to maximize the length of a segment as shown in Algorithm 1. After all the segments are selected, a $B+$ tree is built on these segments.

(ii) Query: in the query process, given a key x, we firstly find which segment this key is located in. For a segment i, x lies in segment i meaning $\mathrm{Seg}[i].\mathrm{start} \le x < \mathrm{Seg}[i+1].\mathrm{start}$. This process can be easily achieved by the $B+$ tree search algorithm. Then, the predicted position pred_pos of x is calculated by equation (8). After interpolating the key's position, the true position of the key is guaranteed to be within the error threshold. Finally, FITing-tree locally searches the region [pred_pos − error, pred_pos + error] using binary search. Figure 3 illustrates this query process.

Input: a key dataset D, error threshold error
Output: a series of disjoint linear segments
(i) Sort dataset D in ascending order.
(ii) Create a new segment Seg = {Seg.Start, Seg.Slope}
(iii) $S_{high} \leftarrow 0$
(iv) $S_{low} \leftarrow \infty$
(v) $i \leftarrow 0$
(vi) Seg.Start = i
(vii) $i = i + 1$
(viii) while $i \leq |D|$ do
(ix) slope = start.$y - D[i]$/start.$x - i$
(x) **if** slope > S_{low} *and* slope < S_{high} **then**
(xi) Add point $(i, D[i])$ to this segment.
(xii) $S_{high} \leftarrow \min(S_{high}, \text{start}.y - D[i] + error/\text{start}.x - i)$
(xiii) $S_{low} \leftarrow \max(S_{low}, \text{start}.y - D[i] - error/\text{start}.x - i)$
(xiv) Seg.*Slope* = $(S_{high} + S_{low}/2)$
(xv) **else**
(xvi) Create a new segment starting with point $(i, D[i])$
(xvii) **end**
(xviii) end

Algorithm 1: The building process of FITing-tree.

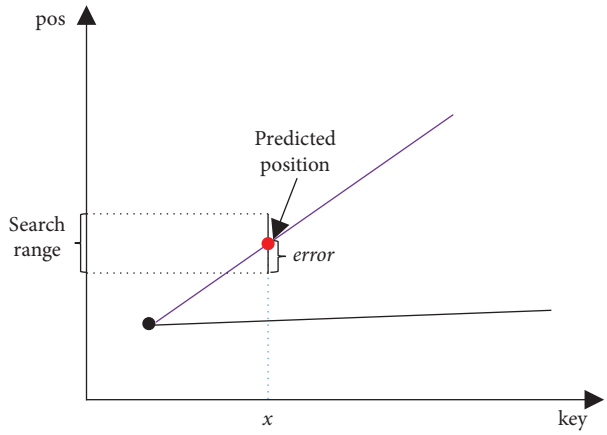

Figure 3: The query process of the FITing-tree.

5. Our Proposed Scheme

n this section, we will present our privacy-preserving cloud-based biometric identification scheme, which consists of four phases, i.e., System Initialization, Index Creation and Encryption, Encrypted Identification Request Generation, and Biometric Identification. Before delving into the details, we give an overview of our proposed scheme. Specifically, in the System Initialization stage, SP firstly generates system parameters (including the security parameters and identification parameters) and distributes them to the client and cloud servers. Then, SP builds an index based on the iDistance and FITing-tree, encrypts the index and the dataset, and outsources them to the cloud in the Index Creation and Encryption stage. The client generates an encrypted identification request based on a given biometric template in the Encrypted Identification Request Generation stage. Eventually, in the Biometric Identification stage, the client sends an encrypted identification request to the cloud, and two cloud servers work together to get the identification result and return it to the client. To describe our proposed scheme clearer, we give the explanation of the main notations used in the following sections in Table 1.

5.1. System Initialization. In the System Initialization phase, SP sets up the system and generates keys for the client and cloud servers. Following the method described in subsection, SP selects the security parameters (k_0, k_1, k_2), calculates the secret key sk = $(p, q, \mathscr{L})$, and generates the public parameters PP = $\{k_0, k_1, k_2, \mathscr{N}\}$, where $\mathscr{N} = p \cdot q$. After that, SP encrypts 0 and 1 with sk, and the corresponding ciphertexts are denoted as $E(0)$ and $E(1)$, respectively. Then, SP sets the identification threshold δ for the identification system. After all parameters are generated, SP publishes PP and δ and sends $E(0), E(1)$ to the client, and sk to CS2, respectively.

Table 1: Definition of main variables in our proposed scheme.

Notation	Definition
sk	The private key of the SHE
E	The encryption process of the SHE scheme
D	The decryption process of the SHE scheme
δ	The threshold of the identification system
$\mathrm{dis}(X, Y)$	The Euclidean distance between X and Y
error	The threshold of the FITing-tree segment
$\{\mathrm{Seg}_1, \mathrm{Seg}_2, \cdots\}$	The segments of the FITing-tree
$\{P_1, P_2, \cdots\}$	The divided partitions while calculating the iDistance

5.2. Index Creation and Encryption. In this phase, SP firstly builds a searching index based on the iDistance data structure and FITing-tree data structure over the biometric dataset $\mathcal{T}$. Then, SP encrypts the index and dataset $\mathcal{T}$ with the SHE scheme. Eventually, SP outsources the encrypted index and dataset to CS1.

5.2.1. Stage 1: Index Building Process. In this stage, the iDistance index for each biometric template in dataset $\mathcal{T}$ is calculated at first. Then, a FITing-tree is built based on these indexes.

(i) Data partition: SP firstly divides the biometric dataset $\mathcal{T}$ into m partitions $\{P_1, P_2, \ldots, P_m\}$ utilizing the K-means algorithm and selects a reference point for each partition, where the reference point for the ith partition P_i is represented by O_i.

(ii) iDistance index calculation: for every template in one partition, the Euclidean distance between this biometric template and the partition's reference point is computed at first. For example, for the ith partition P_i, for any $T = \{t_1, t_2, \ldots, t_l\} \in P_i$, the Euclidean distance between them is computed as $\mathrm{dis}(T, O_i) = \sqrt{\sum_{j=1}^{l} (t_j - o_{ij})^2}$. Besides, the maximum distance of all distance calculated in the ith partition is denoted as $d_{\max-i}$. Then, to avoid the overlap of indexes between different partitions, an offset should be added while calculating the iDistance index. Meanwhile, to keep the gap between indexes of different partitions as small as possible, the offset is selected as $\mathrm{offset}_i = \sum_{j=1}^{i-1} d_{\max-j}$. Eventually, for a biometric template T in the ith partition, its iDistance index is calculated as

$$i\mathrm{Dis}_T = \mathrm{Dis}(T, O_i) + \mathrm{offset}_i. \tag{9}$$

(iii) FITing-tree building: after all the iDistance indexes for all templates are calculated, SP builds a FITing-tree based on these iDistance indexes following Algorithm 1 with a given error threshold. When the building process of the FITing-tree is complete, a series of disjoint linear segments $\{\mathrm{Seg}_1, \mathrm{Seg}_2, \ldots\}$ are generated. Each segment can be represented by its starting point and slope, where $\mathrm{Seg}_i = \{\mathrm{start}_i, \mathrm{slope}_i\}$.

5.2.2. Stage 2: Index Encryption. After the FITing-tree is built, SP encrypts the indexes and the dataset with the SHE scheme. At first, SP encrypts the reference points of each partition with the SHE scheme, and the encrypted reference points are represented as $\{E(O_1), E(O_2), \ldots, E(O_m)\}$. Then, SP encrypts each biometric template in dataset $\mathcal{T}$ using the SHE scheme. For a biometric $T_i = \{t_{i1}, \ldots, t_{il}\} \in \mathcal{T}$, it is encrypted as $E(T) = \{E(t_{i1}), \ldots, E(t_{il})\}$. In the end, SP outsources the encrypted dataset $E(\mathcal{T})$ and the searching index, which includes the encrypted reference points $\{E(O_1), E(O_2), \ldots, E(O_m)\}$, maximum distance list $\{d_{\max-1}, d_{\max-2}, \ldots, d_{\max-m}\}$, and FITing-tree segments $\{\mathrm{Seg}_1, \mathrm{Seg}_2, \ldots\}$ to CS1.

5.3. Encrypted Identification Request Generation. When a client wants to verify whether a biometric template $T_r = \{t_{r1}, t_{r2}, \ldots, t_{rl}\}$ exists in the dataset $\mathcal{T}$, it needs to send an identification request to the cloud servers. Considering the issue of privacy protection, T_r should be encrypted in advance. Since the client receives the ciphertexts $E(0)$ and $E(1)$ in the *System Initialization* phase, the biometric template can be encrypted based on the homomorphic properties of the SHE scheme. The encrypted template is denoted as $E(T_r) = \{E(t_{r1}), E(t_{r2}), \ldots, E(t_{rl})\}$, where

$$E(t_{ri}) = E(1 \cdot t_{ri} + 0 \cdot r_i) = (E(1) \cdot t_{ri} + E(0) \cdot r_i) \bmod \mathcal{N}, \tag{10}$$

and $r_i \in \{0, 1\}^{k_2}$ is a random number.

After the identification request is encrypted, it is sent to CS1.

5.4. Biometric Identification. On receiving a biometric identification request from the client, two cloud servers work together to verify whether it exists in dataset $\mathcal{T}$. Firstly, two cloud servers collaboratively calculate the distance of the identification request corresponding to each reference point. Then, based on the trained FITing-tree and iDistance data structure, a candidate result set is generated. Eventually, two cloud servers traverse the candidate result set to get the identification result.

5.4.1. Stage 1: iDistance Index Calculation. After receiving the biometric request from the client, CS1 firstly calculates the ciphertext of the square of the Euclidean distance between $E(T_r)$ and each encrypted reference point $E(O_i)$, where

$$E\left(\mathrm{dis}(T_r, O_i)^2\right) = \sum_{j=1}^{l} \left(E(o_{ij}) - E(t_{rj})\right)^2. \tag{11}$$

Then, CS1 sends

$$\left\{E\left(\mathrm{dis}(T_r, O_1)^2\right), \ldots, E\left(\mathrm{dis}(T_r, O_m)^2\right)\right\}, \tag{12}$$

to CS2. CS2 decrypts these ciphertexts with sk and returns the corresponding plaintexts to CS1. After getting the plaintexts from CS2, CS1 computes the positive square root

of these plaintexts and gets $\{\mathrm{dis}(T_r, O_1), \ldots, \mathrm{dis}(T_r, O_m)\}$. After that, CS1 checks whether $\mathrm{dis}(T_r, O_i) \le \delta + d_{\max-i}$ holds. If it does, it means that ith partition has an intersection with the query range and will be considered as a candidate partition. Otherwise, it will be ignored. Eventually, the iDistance indexes of the identification request corresponding to each candidate partition's reference point are computed. For example, if ith partition is a candidate partition with reference point O_i, the iDistance index of the identification request with respect to O_i is calculated as $i\mathrm{Dis}_i = \mathrm{dis}(T_r, O_i) + \mathrm{offset}_i$, where $\mathrm{offset}_i = \sum_{j=1}^{i-1} d_{\max-j}$.

5.4.2. Stage 2: Candidate Result Set Generation. When the iDistance indexes of T_r are obtained, a candidate result set is selected for each candidate partitions. According to the range query algorithm of the iDistance, for a given data partition P_i with reference point O_i, we need to search the data points $p \in P_i$ whose iDistance index lies in $[\mathrm{dis}(T_r, O_i) - \delta, \min(\mathrm{dis}(T_r, O_i) + \delta, d_{\max-i})]$. We denote $\mathrm{dis}(T_r, O_i) - \delta$ and $\min(\mathrm{dis}(T_r, O_i) + \delta, d_{\max-i})$ as the lower search bound lb and upper search bound ub, respectively. CS1 finds out the biometric templates, which are stored in the range [pre_pos1 − error, pre_pos2 + error], and adds them to the candidate result set CRS_i, where pre_pos1 is the predicted position of lb calculated by the FITing-tree, and pre_pos2 is the predicted position of ub.

5.4.3. Stage 3: Verification. When the candidate result set is determined, two cloud servers work together to make sure whether there is a biometric template satisfying the identification requirements. Specifically, for each candidate result set CRS_j, CS1 firstly calculates the encrypted distances between T_r and all the templates in this candidate result set. For a template T_i, the encrypted square of the distance from T_r to T_i is calculated as

$$E\left(\mathrm{dis}(T_r, T_i)^2\right) = \sum_{j=1}^{l} \left(E\left(t_{ij}\right) - E\left(t_{rj}\right)\right)^2. \quad (13)$$

Then, the encrypted distances are sent to CS2 to get the plaintexts. While receiving the plaintexts, CS1 finds the template that is closest to T_r and verifies whether it satisfies the identification requirements by checking whether $\mathrm{dis}(T_r, T_i) \le \delta$ holds. If it does, CS1 adds the identifier of T_i to the result set. After all the candidate result sets are checked, CS1 returns the result set to the client.

5.4.4. Correctness. We will show the correctness of the our proposed scheme. If our proposed scheme is not correct, it means that there exists a template $T \in \mathcal{T}$ which satisfies $\mathrm{dis}(T, T_r) \le \delta$, but is not searched by our scheme. To proof the correctness of our scheme, we only need to prove that all the biometric templates $T \in \mathcal{T}$ satisfying $\mathrm{dis}(T, T_r) \le \delta$ are searched by our scheme. All the biometric templates in the candidate result set are verified, and the biometric templates whose position lie in [pre_pos1 − error, pre_pos2 + error] are added to the candidate result set. According to the properties of the FITing-tree, if the predicted position of T lies in [pre_pos1, pre_pos2], it will be added to the candidate result set. Since pre_pos1 is the predicted position of lb calculated by the FITing-tree and pre_pos2 is the predicted position of ub, and the iDistance indexes are in the ascending order in the FITing-Tree, if $\mathrm{dis}(T, T_r) \le \delta$, its iDistance index will lie in $[\mathrm{dis}(O_i, T_r) - \delta, \min(\mathrm{dis}(O_i, T_r) + \delta, \mathrm{dis}_{\max-i})]$. Therefore, if a template $T \in \mathcal{T}$ which satisfies $\mathrm{dis}(T, T_r) < \delta$, it will be searched by our proposed scheme.

6. Security Analysis

In this section, we will analyze the security of our proposed privacy-preserving biometric identification scheme. Since our proposed scheme is designed based on the SHE scheme, which has been proved to be CPA-secure in previous work [17], we mainly focus on the privacy-preserving goal described in section. Specifically, both CS1 and CS2 cannot obtain the plaintext of the biometric dataset and biometric identification request. In the following, we will prove the security of our scheme from the view of the two cloud servers.

Theorem 1. *CS1 cannot obtain the plaintext of the biometric dataset and the biometric identification request during the biometric identification process.*

Proof. We give the view of CS1 during the biometric identification process firstly and analyze why CS1 cannot obtain the plaintext of the biometric dataset and biometric identification request.

(i) View 1: in the Index Creation and Encryption phase, the encrypted biometric dataset $E(\mathcal{T})$ and the encrypted searching indexes including the encrypted reference points $\{E(O_1), E(O_2), \ldots, E(O_m)\}$, maximum distance list $\{d_{\max-1}, d_{\max-2}, \ldots, d_{\max-m}\}$, and FITing-tree segments $\{\mathrm{Seg}_1, \mathrm{Seg}_2, \ldots\}$ are sent to CS1. Since the biometric dataset and reference points are encrypted by the SHE scheme, and CS1 does not have the secret key sk, CS1 cannot get any information about the plaintext of the biometric data from these encrypted data directly. By analyzing the maximum distance list, CS1 can only learn that there exists a biometric template T_i in each data partition that satisfies

$$\mathrm{dis}(T_i, O_i) = d_{\max-i}, \quad (14)$$

where $1 < i < m$. Since the reference points and biometric template are both encrypted by the SHE scheme, CS1 cannot infer their plaintext from these data. The FITing-tree segments are built on the iDistance indexes and do not have corresponding relation to the biometric dataset or identification request; thus, CS1 cannot obtain the plaintext from the segments either.

(ii) View 2: the encrypted biometric identification request. Since the biometric identification request is also encrypted by the SHE scheme and CS1 cannot decrypt it, the plaintext of the biometric identification will not be leaked to CS1.

(iii) View 3: intermediate values during the biometric identification process. In the identification process, the distance between the biometric identification request and each reference point and the distance between the biometric identification request and each candidate template are leaked to CS1. Since the biometric identification request, the reference points, and the biometric template in the candidate result set are all encrypted by the SHE scheme, CS1 cannot get the plaintext of the biometric dataset and biometric identification request from these intermediate values.

Therefore, CS1 cannot obtain the plaintext of the biometric dataset and the biometric identification request during the biometric identification process. □

Theorem 2. *CS2 cannot obtain the plaintext of the biometric dataset and the biometric identification request during the biometric identification process.*

Proof. We give the view of CS2 during the biometric identification process firstly and analyze why CS2 cannot learn the plaintext of the biometric dataset, biometric identification request from these data.

(i) View 1: the distance between the biometric identification request and each reference point. While calculating the iDistance indexes of the biometric identification request, CS2 gets the distance between the biometric identification request and each reference point. However, CS2 cannot get the encrypted reference points and biometric identification request; thus, CS2 cannot obtain the plaintext of the biometric identification request.

(ii) View 2: the distance between the biometric identification request and each candidate template. While verifying the template in the candidate result set, CS2 obtains the distance between the biometric identification request and each candidate template. Since the encrypted biometric dataset and biometric identification request are kept secret from CS2, CS2 cannot get the plaintext of the biometric dataset and the biometric identification request.

Therefore, CS2 cannot learn the plaintext of the biometric dataset and the biometric identification request during the biometric identification process. □

7. Performance Evaluation

In this section, we evaluate the performance of our proposed scheme in terms of computational costs and communication overhead. Specifically, we will compare our proposed scheme with an M-tree based privacy-preserving biometric identification scheme named MASK [26]. The reason why we compare with MASK is twofold: (i) MASK is designed under the same system model with our proposed scheme. (ii) MASK is more efficient than other schemes designed for the biometric identification scenario in terms of computational and communication costs.

7.1. Evaluation Environment. In order to measure the integrated performance, we implement both schemes with Java and conduct some experiments on an Intel Xeon 6226R CPU@2.9 GHz Windows platform with 256 GB RAM. The SHE scheme is used to protect the privacy of the dataset and identification requests in these two schemes. The security parameters are set as $k_0 = 2048$, $k_1 = 20$ and $k_2 = 160$. A real-world dataset and a synthetic dataset are used to test the performance of these two schemes. These two datasets are prepared as follows.

(i) Real-world dataset: we choose the Labeled Faces in the Wild (LFW) dataset [28], which contains 13223 face images collected from 5749 individuals. In this dataset, 1680 of the people pictured have two or more distinct photos. In this paper, we use the FaceNet algorithm to extract face features from these face images at first. Each extracted face feature is a 512-dimensional vector, and all the face features live on the same hypersphere, which means that each of the dimensions of the vector is in the range (-1,1).

(ii) Synthetic dataset: we randomly generate a synthetic dataset that contains 8×10^4 face features. Each face feature is a 512-dimensional vector, and all face features lie in the same range $(-1, 1)$ as the face features extracted by the FaceNet. The templates in the synthetic dataset are distributed in a hypercubes, in which each dimension is lying in (-1, 1).

7.2. Computational Costs. In this section, we will evaluate the computational costs of our proposed scheme while generating the searching index, encrypting identification request, and answering the biometric identification requests, which correspond to the computational costs in Index Creation and Encryption phase, Encrypted Identification Request Generation phase, and Biometric Identification phase, respectively. Since the data in these two schemes are encrypted by the SHE scheme, we denote the computational costs of encrypting and decrypting data by the SHE scheme as C_{enc} and C_{dec}, the computational costs of adding and multiplying two SHE ciphertexts as $C_{\text{add}-I}$ and $C_{\text{mul}-I}$, and the computational costs of multiplying a plaintext and SHE ciphertext as $C_{\text{mul}-II}$.

7.2.1. Index Creation and Encryption. As described in Section 5, there are two stages in the Index Creation and Encryption phase. The computational costs in this phase are related to the dataset size n, data dimension l, and partition numbers k.

(i) Index building process: in this stage, the biometric dataset is firstly divided into m partitions using the K-means algorithm. Then, a reference point is selected for each partition. In Figure 4, we plot the computational costs of partitioning the dataset versus with n when $l = 512$ and $m = 10$. Then, the distance between the biometric templates in each partition and the partition's reference point is calculated. Later, a FITing-tree is built on these iDistance indexes. The computational costs of creating the index are shown in Figure 5.

(ii) Index encryption: when the index building process is complete, the searching index is encrypted. While encrypting the searching index, m reference points are encrypted. Therefore, the computational costs of encrypting the searching index are $m \cdot l \cdot C_{\text{enc}}$. In MASK, the index size is related to the node capacity of the M-tree [29]. Given the node capacity C, there are at most $\sum_{w=1}^{w_{\max}} (\lceil n/C^w \rceil)$ nodes in the M-tree, where $C^{w_{\max}} < n < C^{w_{\max}+1}$. Therefore, the computational costs of encrypting the searing index in MASK are less than $\sum_{w=1}^{w_{\max}} (\lceil n/C^w \rceil) \cdot l \cdot C_{\text{enc}}$. Since the computational costs of encrypting the dataset are the same in these two schemes, we mainly focus on comparing the computational costs of encrypting the searching indexes. We test the computational costs of encrypting the searching indexes in both schemes over the synthetic dataset, and the results are shown in Figure 6 ($m = 10$).

We can see that our proposed scheme is more efficient in generating and encrypting the searching indexes.

7.2.2. Encrypted Identification Request Generation. As described in Section 5, the client generates the encrypted identification request by encrypting the biometric template. Then, the encrypted identification request is sent to the cloud servers to get the identification result. 2 operations of multiplying a plaintext and SHE ciphertext are needed to encrypt each dimension of the identification request. Since the length of the biometric template is l, the computational costs of encrypting the identification template are $2lC_{mul-II}$ in our proposed scheme. In MASK, the computational costs of generating the encrypted identification are $2(l+1)C_{mul-II}$. In this phase, the computational costs are constant when the template length and the security parameters are set to fixed values.

7.2.3. Biometric Identification. In the biometric identification phase, there are three stages:

(i) iDistance index calculation: in this stage, two cloud servers work together to find out which partition the identification request belongs to. At first, the encrypted square of Euclidean distance between the identification request and each partition's reference points is calculated over their ciphertexts by CS1. Then, the encrypted data is sent to CS2 to get the plaintext. The computational costs of CS1 in this phase are •fa, and the computational costs of CS2 are $m \cdot l \cdot C_{\text{dec}}$.

(ii) Candidate result set generation: in this phase, CS1 firstly finds out which segments lb and ub lie in by searching in the B+ tree built over the FITing-tree segments. Then, CS1 calculates the predicted position calculation of lb and ub and determines the candidate result set. Since the predicted position is calculated based on the plaintext, the computational costs in this phase contain the searching costs and predicted position calculation costs.

(iii) Verification: in this phase, two cloud servers collaboratively traverse the biometric templates in the candidate result set. The computational costs of CS1 in this phase are $\sum_{i=1}^{m} (|\text{CRS}_i|) \cdot (2lC_{\text{mul}-I} + (2l-1)C_{\text{add}-I})$, and the computational costs of CS2 in this phase are $\sum_{i=1}^{m} (|\text{CRS}_i|) \cdot l \cdot C_{\text{dec}}$, where $|\text{CRS}_i|$ is the size of the ith partition's candidate result set.

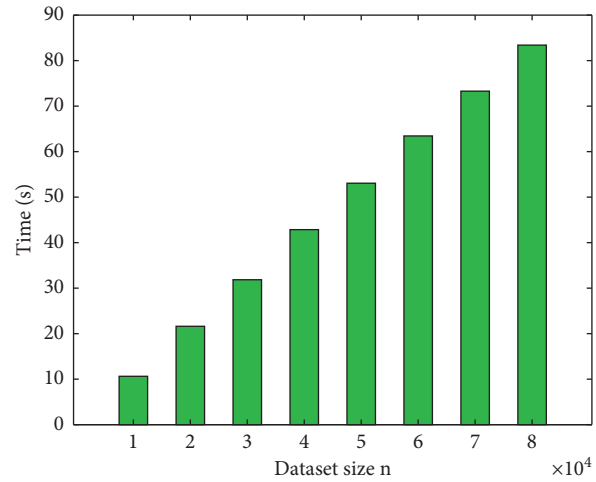

Figure 4: The running time of partitioning the dataset varies with dataset size n.

In MASK, computational costs in this phase consist of the computational costs in searching the M-tree and verifying nodes in leaf nodes. The computational costs of CS1, CS2 in these two schemes are shown in Figures 7(a) and Figure 7(b), respectively. And the integrated running time in the biometric identification process of our proposed scheme and MASK is shown in Figure 7(c). We can see that our proposed scheme takes more time than MASK in identifying a biometric template when the biometric dataset is not very large. But as the size of the biometric dataset grows, our scheme is advantageous in the computational cost of the identification process. Since the searching process in each data partition is independent, it is easy to accelerate the search process of our proposed scheme in a concurrent way. As shown in 7(c), the efficiency of our proposed scheme is greatly improved when it is executed concurrently.

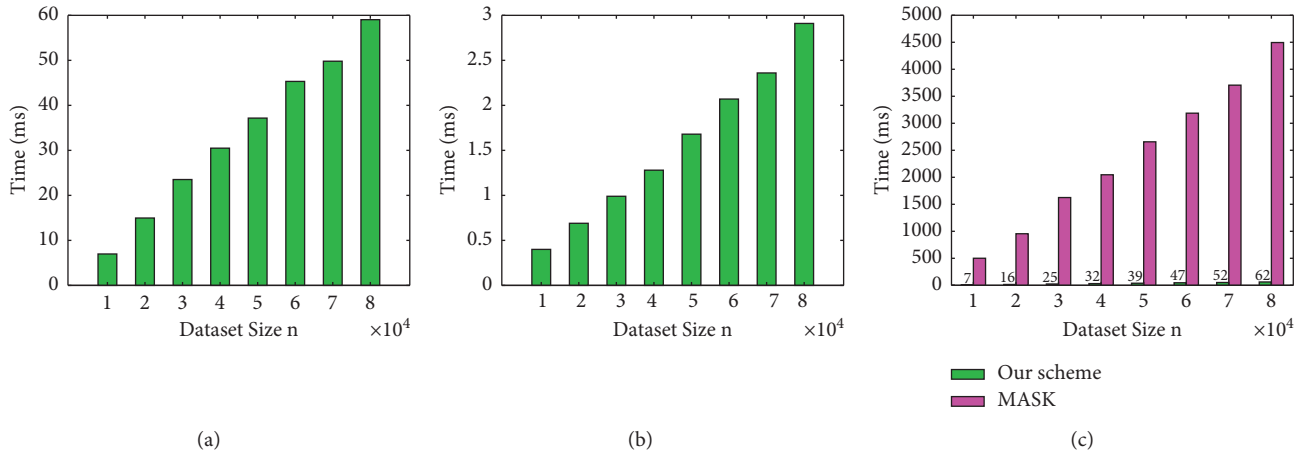

Figure 5: The computational cost of creating the index. (a) The running time of calculating iDistance index with different n. (b) The running time of building FITing-tree with different n. (c) The integrated running time of creating index with different n.

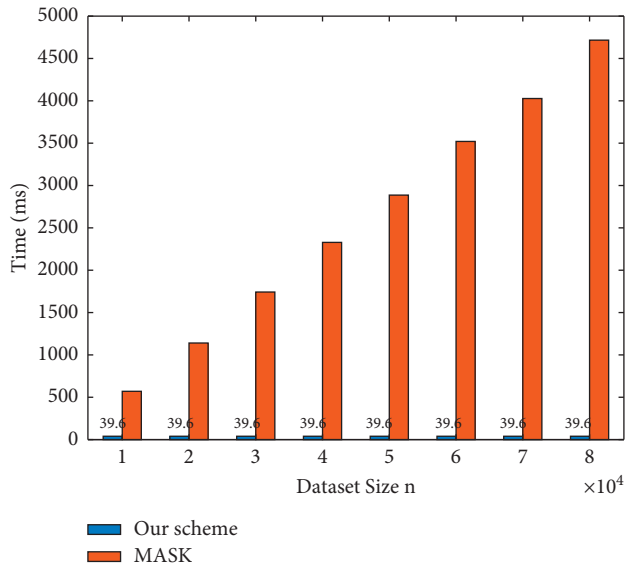

Figure 6: The running time of encrypting indexes varies with dataset size n.

7.3. Communication Overhead. In this section, we will evaluate the communication overhead of our proposed scheme when outsourcing the searching index and the encrypted biometric template dataset, submitting the encrypted identification request and searching the biometric templates, which are corresponding to the communication overhead in Index Creation and Encryption phase, Encrypted Identification Request Generation phase, and Biometric Identification phase, respectively. We analyze the communication overhead in theory at first and test it over the synthetic dataset. For the sake of simplicity, we denote the bit length of an integer and a floating number as L_i and L_f, respectively.

7.3.1. Index Creation and Encryption. In the Index Creation and Encryption phase, the encrypted reference points $\{E(O_1), E(O_2), \ldots, E(O_m)\}$, maximum distance list $\{d_{\max-1}, d_{\max-2}, \ldots, d_{\max-m}\}$, the encrypted $\mathcal{T}$, and FITing-tree segments $\{\mathrm{Seg}_1, \mathrm{Seg}_2, \ldots\}$ are outsourced to CS1. According to the SHE scheme, the size of the ciphertext is k_0 bits. Since there are m reference points and n biometric templates in the dataset $\mathcal{T}$, where both the reference point and biometric template are l-dimensional vectors, the size of the encrypted reference point is $m \cdot l \cdot k_0$ and the size of the encrypted dataset is $n \cdot l \cdot k_0$. As the data in the maximum distance list is stored in floating number, their size is $m \cdot L_f$. A FITing-tree segment consists of a start point and the slope, and the size of each FITing-tree segment is $(L_i + L_f)$. Suppose that there are u segments contained in the FITing-tree, the size of FITing-tree segments is $u(L_i + L_f)$. According to the building process of FTIing-tree, there are at least error + 1 data points in a segment, which means that $u \le n/\text{error} + 1$. In MASK, there are at most $\sum_{w=1}^{w_{\max}} (\lfloor n/C^w \rfloor)$ nodes in the M-tree, where $C^{w_{\max}} < n < C^{w_{\max}+1}$. Hence, the communication overhead of MASK in this stage is less than $(n + \sum(\lfloor n/C^w \rfloor) \cdot l \cdot k_0$.

7.3.2. Encrypted Identification Request Generation. In the Encrypted Identification Request Generation phase, the identification request is encrypted and sent to cloud servers. Since the identification request is an l-dimensional vector, and the ciphertext of each dimension is k_0 bits, the size of the identification request is lk_0. The communication overhead of MASK in this phase is $(l + 1)k_0$.

7.3.3. Biometric Identification. In the Biometric Identification stage, two cloud servers work together to get the identification request. Firstly, two cloud servers select the candidate partition. Then, two cloud servers generate a candidate result set for each partition. Eventually, two cloud servers traverse all the candidate result sets to get the identification result. In this stage, $\sum_{i=1}^{m} (|\mathrm{CRS}_i|) + m$ encrypted data are sent from CS1 to CS2, and $\sum_{i=1}^{m} (|\mathrm{CRS}_i|) + m$ plaintext data are sent from CS2 to CS1.

We test the communication costs of different phases in both two schemes over the synthetic dataset, and the experimental results are shown in Figure 8. Specifically,

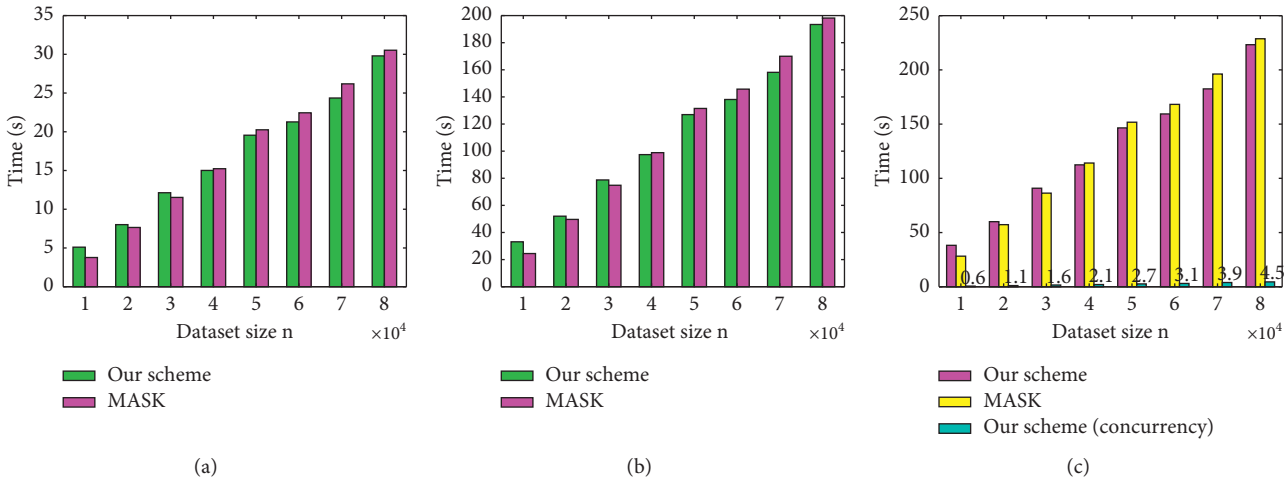

Figure 7: The computational cost in the identification stage. (a) The running time of CS1 in identification stage with different n. (b) The running time of CS2 in identification stage with different n. (c) The integrated running time in identification stage with different n.

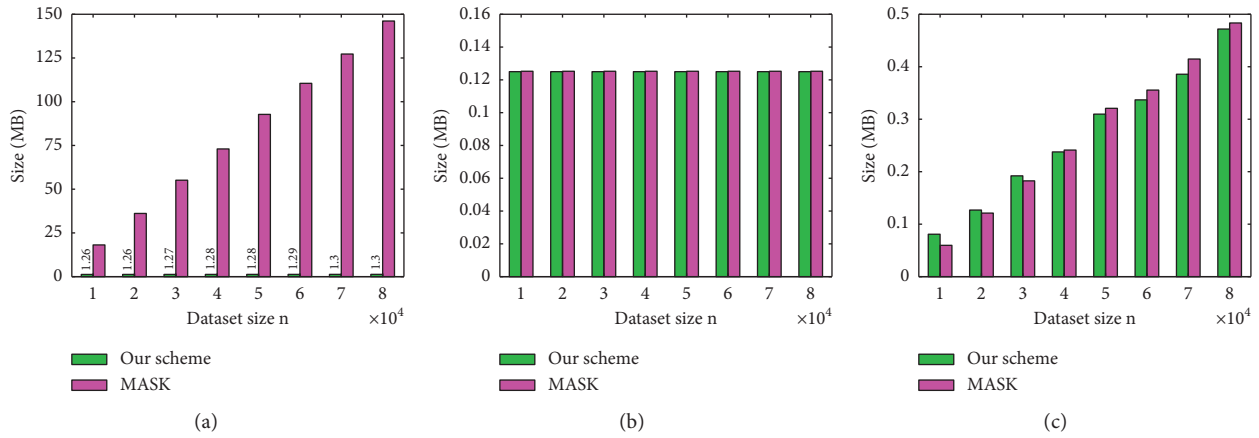

Figure 8: The communication costs of our proposed scheme and MASK. (a) The communication costs of sending the indexes with different n. (b) The communication costs of sending the identifiaction request with different n. (c) The communication costs in the identification phase with different n.

Figure 8(a) shows the communication costs of sending the indexes in these two schemes. The experimental results demonstrate that the communication costs of our proposed scheme in this phase are much lower than those of MASK. Figure 8(a) shows the communication costs of sending the identification request in both two schemes. The communication costs in this phase are almost the same, but our proposed scheme is more efficient. Figure 8(a) shows the communication costs while identifying a template in the dataset. The results show that our proposed scheme sends more data than MASK when the dataset is not very large. But our proposed scheme is more and more efficient when the dataset size grows.

7.4. Storage Cost. In our proposed scheme and MASK, the storage consumption of the cloud servers is mainly used to store the search indexes and the encrypted dataset. Since the size of the encrypted dataset is the same in these two schemes, we mainly compare the storage cost of storing the search indexes. In our proposed scheme, the search indexes consist of the encrypted reference points, the maximum distance list, and the FITing-tree segments. In MASK, the search indexes consist of the M-tree indexes. The storage cost of storing the searching indexes of these two schemes is shown in Table 2.

7.4.1. Accuracy. In our proposed scheme, the combination iDistance and FITing-tree structure can achieve accurate range query. Transforming the biometric template into integers may reduce the accuracy of the identification scheme. This influence is very slight when enough decimal places are kept during the transforming process. We test the accuracy of our proposed scheme in terms of the false acceptance rate (FAR), false rejection rate (FRR), and equal error rate (ERR) over the LFW dataset [28]. We firstly test the accuracy of the original FaceNet algorithm in terms of FAR, FRR, and EER varying with thresholds from 0 to 2, and the result is shown in Figure 9(a). The ERR of the original

Table 2: Storage cost (MB) of our proposed scheme vs. MASK with different dataset size.

Scheme	1×10^4	2×10^4	3×10^4	4×10^4	5×10^4	6×10^4	7×10^4	8×10^4
Our scheme	1.257	1.264	1.271	1.277	1.284	1.292	1.297	1.304
MASK	18.125	36.125	55.125	73.000	92.750	110.500	127.250	146.125

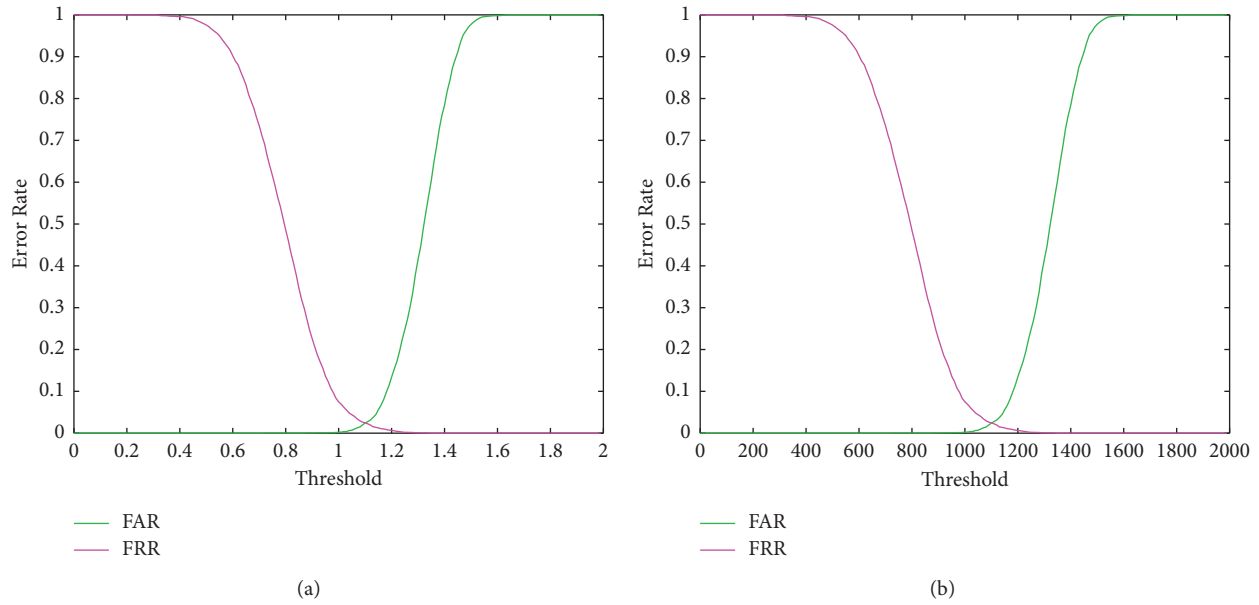

Figure 9: The error rate of the recognition algorithm. (a) The FAR and FRR of the original FaceNet algorithm varying with threshold. (b) The FAR and FRR of the FaceNet algorithm with integer templates varying with threshold.

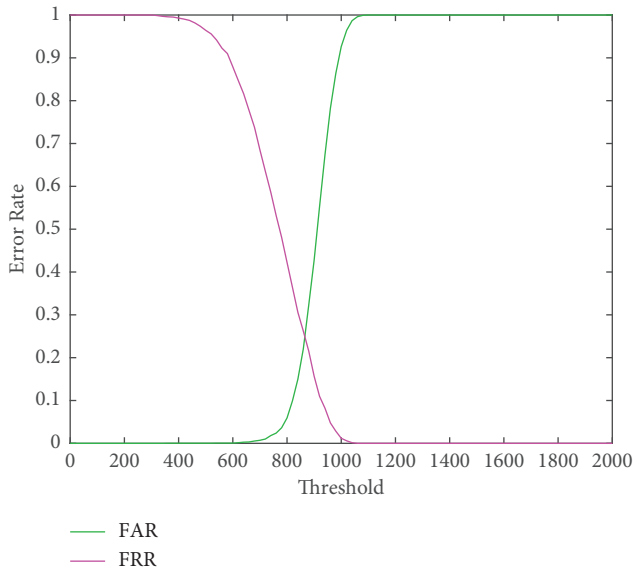

Figure 10: The FAR and FRR of the identification scheme varying with threshold.

FaceNet algorithm is 0.025. Then, we also test the accuracy of the FaceNet algorithm with the biometric templates, which have been converted to integers. In this experiment, the biometric template values are converted to integer values with 3 decimals kept. We test the FAR, FRR varying with thresholds from 0 to 2000, and the result is shown in Figure 9(b). The ERR of FaceNet algorithm with integer templates is 0.026. We can see the accuracy is kept almost the same as the original FaceNet algorithm.

After that, we also evaluate the accuracy of our proposed scheme in terms of FAR, FRR, and ERR in the identification scenario. We test the FAR, FRR varying with thresholds from 0 to 2000, and the result is shown in Figure 10. The ERR of the identification scheme is 0.249.

8. Conclusion

In this paper, we have proposed an efficient and privacy-preserving identification scheme for identifying an individual in huge biometric datasets. Specifically, we introduced the FITing-tree to generate an index for the biometric dataset based on which the efficient identification service can be achieved. Then, we use the SHE technique to ensure the privacy of identification requests and the biometric dataset. The security of our proposed scheme has been analyzed, and the result shows that the privacy of both the biometric dataset and biometric identification can be preserved. To evaluate the computational and communication cost of our proposed scheme, we implement it and test it over a synthetic dataset. Experimental results demonstrate that our proposed scheme is efficient in terms of computational and communication costs when identifying a biometric template in a large dataset.

Data Availability

The data used to support the findings of this study are available at http://vis-www.cs.umass.edu/lfw/index.html.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (61972304 and 61932015), Natural Science Foundation of Shaanxi Province (2019ZDLGY12-02), and Technical Research Program of the Ministry of Public Security (2019JSYJA01).

References

[1] Knud Lasse Lueth, "State of the iot 2020: 12 billion iot connections, surpassing non-iot for the first time," [EB/OL]. https://iot-analytics.com/state-of-the-iot-2020-12-billion-iot-connections-surpassing-non-iot-for-the-first-time/.

[2] K.. Leo, "Gatwick airport commits to facial recognition tech at boarding," [EB/OL]. https://www.bbc.com/news/technology-49728301.

[3] F. Serrano and A. Kazda, "The future of airports post COVID-19," *Journal of Air Transport Management*, vol. 89, no. 1–1, Article ID 101900, 2020.

[4] L. Garg, R. Bilas Pachori, X. Zhang, S. K. Pani, and S. K. Singh, *A Biometric Technology-Based Framework for Tackling and Preventing Crimes*, Wiley Online Library, Hoboken, New Jersey, 2021.

[5] P. Elena and A. Mitrokotsa, "Privacy-preserving biometric authentication: challenges and directions," *Security and Communication Networks*, vol. 2017, Article ID 7129505, 9 pages, 2017.

[6] J. Yuan and S. Yu, "Efficient privacy-preserving biometric identification in cloud computing," in *Proceedings of the IEEE INFOCOM 2013*, pp. 2652–2660, IEEE, Turin, Italy, April 2013.

[7] Q. Wang, S. Hu, K. Ren, and M. He, "Cloudbi: practical privacy-preserving outsourcing of biometric identification in the cloud," in *Proceedings of the Computer Security - ESORICS 2015 - 20th European Symposium on Research in Computer Security*, G. Pernul, Y. Peter, A. Ryan, and E. R. Weippl, Eds., pp. 186–205, Springer, Vienna, Austria, September 2015, *Proceedings, Part II*, volume 9327 of *Lecture Notes in Computer Science*.

[8] C. Zhang, L. Zhu, and C. Xu, "PTBI: an efficient privacy-preserving biometric identification based on perturbed term in the cloud," *Information Sciences*, vol. 409-410, pp. 56–67, 2017.

[9] H. Higo, T. Isshiki, K. Mori, and S. Obana, "Privacy-preserving fingerprint authentication resistant to hill-climbing attacks," in *Proceedings of the Selected Areas in Cryptography - SAC 2015 - 22nd International Conference*, O. Dunkelman and L. Keliher, Eds., pp. 44–64, Springer, Sackville, NB, Canada, August 2015, Revised Selected Papers, *volume 9566 of* Lecture Notes in Computer Science.

[10] Y. Wang, J. Wan, J. Guo, Y.-M. Cheung, and P. C. Yuen, "Inference-based similarity search in randomized montgomery domains for privacy-preserving biometric identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 7, pp. 1611–1624, 2018.

[11] Y. Zhu, X. Li, J. Wang, and J. Li, "Cloud-assisted secure biometric identification with sub-linear search efficiency," *Soft Computing*, vol. 24, no. 8, pp. 5885–5896, 2020.

[12] F. Wang, G. Xu, C. Wang, and J. Peng, "A provably secure biometrics-based authentication scheme for multiserver environment," *Security and Communication Networks*, vol. 2019, Article ID 2838615, 15 pages, 2019.

[13] T. Kim, Y. Oh, and H. Kim, "Efficient privacy-preserving fingerprint-based authentication system using fully homomorphic encryption," *Security and Communication Networks*, vol. 2020, Article ID 4195852, 11 pages, 2020.

[14] A. Galakatos, M. Markovitch, C. Binnig, R. Fonseca, and T. Kraska, "Fiting-tree: a data-aware index structure," in *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019*, P. A. Boncz, S. Manegold, A. Ailamaki, A. Deshpande, and T. Kraska, Eds., ACM, Amsterdam, The Netherlands, pp. 1189–1206, July 2019.

[15] H. V. Jagadish, B. C. Ooi, K.-L. Tan, C. Yu, and R. Zhang, "iDistance," *ACM Transactions on Database Systems*, vol. 30, no. 2, pp. 364–397, 2005.

[16] M. Hassan, R. Lu, Y. Zheng, J. Shao, and G. Ali, "Achieving o(log3n) communication-efficient privacy-preserving range query in fog-based iot," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5220–5232, 2020.

[17] Y. Zheng, R. Lu, Y. Guan, J. Shao, and H. Zhu, "Efficient and privacy-preserving similarity range query over encrypted time series data," *IEEE Transactions on Dependable and Secure Computing*, vol. 1–1, p. 1, 2021.

[18] Y. Huang, L. Malka, D. Evans, and J. Katz, "Efficient privacy-preserving biometric identification," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011*, The Internet Society, San Diego, California, USA, February 2011.

[19] M. Blanton and P. Gasti, "Secure and efficient protocols for iris and fingerprint identification," in *Proceedings of the Computer Security - ESORICS 2011 - 16th European Symposium on Research in Computer Security*, V. Atluri and C. Díaz, Eds., pp. 190–209, Springer, Leuven, Belgium, September 2011, Proceedings, volume 6879 of Lecture Notes in Computer Science.

[20] M. Barni, G. Droandi, and R. Lazzeretti, "Privacy protection in biometric-based recognition systems: a marriage between cryptography and signal processing," *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 66–76, 2015.

[21] T. Hirano, M. Hattori, T. Ito, and N. Matsuda, "Cryptographically-secure and efficient remote cancelable biometrics based on public-key homomorphic encryption," in *Proceedings of the Advances in Information and Computer Security - 8th International Workshop on Security, IWSEC 2013*, K. Sakiyama and M. Terada, Eds., pp. 183–200, Springer, Okinawa, Japan, November 2013, Proceedings, volume 8231 of Lecture Notes in Computer Science.

[22] A. Mandal, A. Roy, and M. Yasuda, "Comprehensive and improved secure biometric system using homomorphic encryption," in *Proceedings of the Data Privacy Management, and Security Assurance - 10th International Workshop, DPM 2015, and 4th International Workshop, QASA 2015*, J. García-Alfaro, G. Navarro-Arribas, A. Aldini, F. Martinelli, and N. Suri, Eds., pp. 183–198, Springer, Vienna, Austria, September 2015, Revised Selected Papers, volume 9481 of Lecture Notes in Computer Science.

[23] Y. Zhu, T. Takagi, and R. Hu, "Security analysis of collusion-resistant nearest neighbor query scheme on encrypted cloud data," *IEICE - Transactions on Info and Systems*, vol. E97.D, no. 2, pp. 326–330, 2014.

[24] H. Delfs and H. Knebl, "Introduction to cryptography - principles and applications," *Information Security and Cryptography*, Springer, Berlin, Germany, 3rd edition, 2015.

[25] S. Hu, M. Li, Q. Wang, S. S. M. Chow, M. Du, and M. Du, "Outsourced biometric identification with privacy," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2448–2463, 2018.

[26] X. Yang, H. Zhu, F. Wang, S. Zhang, R. Lu, and H. Li, "MASK: efficient and privacy-preserving m-tree based biometric identification over cloud," *Peer-to-Peer Networking and Applications*, vol. 14, no. 4, pp. 2171–2186, 2021.

[27] F. Schroff, D. Kalenichenko, and P. James, "Facenet: a unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015*, pp. 815–823, IEEE Computer Society, Boston, MA, USA, June 2015.

[28] "Lfw face database," [EB/OL]. http://vis-www.cs.umass.edu/lfw/index.html.

[29] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: an efficient access method for similarity search in metric spaces," in *Proceedings of the VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases*, M. Jarke, M. J. Carey, and K. R. Dittrich, Eds., Morgan Kaufmann, Athens, Greece, pp. 426–435, August 1997.