CloudBI: Practical Privacy-Preserving Outsourcing of Biometric Identification in the Cloud

Qian Wang[1], Shengshan Hu[1], Kui Ren[2], Meiqi He[1], Minxin Du[1], and Zhibo Wang[1](✉)

[1] State Key Lab of Software Engineering, School of CS, Wuhan University, Wuhan, China
{qianwang,zbwang}@whu.edu.cn

[2] Department of CSE, University at Buffalo, Suny, Buffalo, USA
kuiren@buffalo.edu

Abstract. Biometric identification has been incredibly useful in the law enforcement to authenticate an individual's identity and/or to figure out who someone is, typically by scanning a database of records for a close enough match. In this work, we investigate the privacy-preserving biometric identification outsourcing problem, where the database owner outsources both the large-scale encrypted database and the computationally intensive identification job to the semi-honest cloud, relieving itself from data storage and computation burden. We present new privacy-preserving biometric identification protocols, which substantially reduce the computation burden on the database owner. Our protocols build on new biometric data encryption, distance-computation and matching algorithms that novelly exploit inherent structures of biometric data and properties of identification operations. A thorough security analysis shows that our solutions are practically-secure, and the ultimate solution offers a higher level of privacy protection than the-state-of-the-art on biometric identification outsourcing. We evaluate our protocols by implementing an efficient privacy-preserving fingerprint-identification system, showing that our protocols meet both the security and efficiency needs well, and they are appropriate for use in various privacy-preserving biometric identification applications.

Keywords: Biometric identification · Data outsourcing · Privacy · Cloud computing

1 Introduction

Biometric data, which include fingerprints, DNA, irises, voice patterns, palmprints, and facial patterns etc., are the measurable biological or behavioral characteristics widely-used for identification of individuals [9]. Matching biometric data or biometric identification has been incredibly useful in the law enforcement to authenticate an individual's identity and/or to figure out who someone

G. Pernul et al. (Eds.): ESORICS 2015, Part II, LNCS 9327, pp. 186–205, 2015.
DOI: 10.1007/978-3-319-24177-7_10

is, typically by scanning a database of records for a *good* match. A typical biometric identification system consists of a server-side database owner and users who submit candidate biometric records to the database owner for profile identification. Formally, the database owner holds a large set of biometric records $\mathcal{D} = \langle \mathbf{b}_i, p_i \rangle_{i=1}^{m}$, where $\mathbf{b}_i$ denotes the biometric data corresponding to its identity profile p_i. A user who has a candidate biometric record $\mathbf{b}_c$ wants to learn the target profile p_{i^*} for which $\mathbf{b}_{i^*}$ matches the query $\mathbf{b}_c$ closely enough according to a certain metric.

Nowadays with the increasing development and popularity of cloud computing, individuals, companies and governments are highly motivated to outsource their data onto remote cloud servers to get rid of expensive local storage and computation costs [14]. As far as the biometric identification system is concerned, the database owner (*e.g.*, the FBI is responsible for managing the national fingerprint collection) may desire to outsource the extremely large size of biometric data records to the cloud, readily enjoying the biometric data matching service from the cloud service provider (*e.g.*, Amazon). However, to protect the privacy of sensitive biometric data, the database owner should *encrypt* the database before outsourcing. Whenever a government agency (*e.g.*, the FBI's partner) wants to authenticate an individual's identity or to figure out who someone is (by a fingerprint left on a murder weapon or a bomb, for example), he will turn to the FBI and issue an identification query. After receiving the query from the user, the FBI also generates the encrypted query, which allows the cloud server to execute it over the encrypted database, *i.e.*, scanning the encrypted database for a close match. Now the challenging problem is *how to enable privacy-preserving biometric identification over the encrypted database while apparently relieving the database owner of its high computation burden and relying on the cloud for providing fast and reliable biometric identification service.*

Privacy-preserving biometric identification has been extensively investigated in the secure two-party computation model, where the database owner and the user interactively execute the identification protocol without revealing the self-biometric data information to each other [1,6,15,17]. These works, however, either have efficiency issues (heavily rely on homomorphic encryption) [6] or fail to support the computation of a global minimum [15], which limits their applications. To enable efficient identification for a large-scale database, recently Huang *et al.* [8] and Blanton *et al.* [3] proposed privacy-preserving biometric identification protocols by combining both homomorphic encryption and garbled circuits [12]. Still, the biometric identification problem is essentially formulated as a secure two-party computation problem, their solutions cannot be directly applied to the identification outsourcing model. This is because the semi-trusted cloud server cannot know any private inputs/data except for the encrypted biometric database in the outsourcing computation model. The direct extensions of the above approaches, if applied to our model, (i) will lead to extremely high communication overhead and (ii) cannot relieve the database owner of a high computation burden, *i.e.*, for each identification query, the database owner has to traverse the database to compute the Euclidean distances without taking advantage of the cloud for undertaking heavy computations.

Recently, the biometric identification problem has been explored in the outsourced environment [2,4]. In [2], its single-server solution is far from practical for a large database while its multi-server solution requires that the database is shared among (at least three) servers in a split form. In [4], the authors developed a new outsourceable approach that secures the database and the candidate biometric. But it is assumed that two non-colluding servers cooperate to run the protocol and one of them knows the secret key. Moreover, its protocol requires frequent interactions between two servers which will lead to too much communication overhead. Another line of work similar to privacy-preserving identification is the kNN search problem over the encrypted database. Most of them, however, such as [16], considered the kNN problem in the two-party computation model. In [19], the authors proposed a new "encryption" scheme that achieves privacy-preserving outsourcing, but the scheme can be cracked when there exist collusions between the cloud server and the user. As a following work, [5] proposed a secure kNN query protocol which achieves a higher security level than [19]. But it also assumes the cloud to be two non-colluding servers and has the same drawbacks (*i.e.*, leakage of key secret and low efficiency) as [4], due to the use of the same techniques to compute Euclidean distance. The most similar work to ours is the biometric identification scheme proposed by Yuan *et al.* [20], which also considered the cloud-based identification outsourcing model. It appears that very high identification efficiency can be obtained as compared to [8]. They claimed their scheme is secure under the known-plaintext attack (KPA) model or even the chosen-plaintext attack (CPA) model. We show that, however, the security arguments given for their work do not hold, and the scheme can be completely broken once we manipulate the ciphertexts to remove the introduced randomness in the presence of collusion.

In this work, we, for the first time, identify the deficiencies and security weaknesses of previous privacy-preserving biometric identification protocols in the computation outsourcing model. We propose new schemes that support privacy-preserving biometric identification outsourcing in cloud computing. Our design is carefully tuned to meet the security and efficiency requirements under a three-party outsourcing computation model, where one or two parties may be semi-honest. We exploit inherent structures of biometric data and properties of biometric identification operations and use similarity transformation, trace computation by eigenvalues, and triangular matrix to design effective biometric data encryption algorithms (on the database owner side) while enabling privacy-preserving and correct distance-computation and matching over encrypted biometric data (on the cloud server side). Our main contributions can be summarized as follows.

- We formulate the problem of privacy-preserving biometric data identification outsourcing, establish a well-defined threat model by carefully characterizing attack-specific capabilities in various scenarios. We examine the state-of-the-art solutions and show their insufficiencies and security weaknesses when

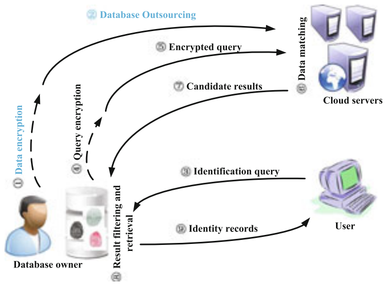


Fig. 1. An overview of cloud-based biometric-matching system.

meeting the practical needs under the biometric identification outsourcing model.

– We present a suite of privacy-preserving biometric identification protocols, which achieve different levels of security strength and substantially reduce the computation burden on the database owner side. Our protocols build on new and secure biometric data encryption and matching algorithms that novelly exploit inherent structures of biometric data and properties of biometric identification. A thorough security analysis shows that our solutions are practically-secure in different attack models.
– We have implemented our protocols to build an efficient privacy-preserving fingerprint-identification system. The system performance is carefully evaluated for each phase of the protocol, in terms of preparation time, communication cost and identification time. Our protocols meet both the security and efficiency needs well, and they are appropriate for use in various privacy-preserving biometric identification applications.

2 Problem Formulation: Outsourcing Computation of Biometric Identification

2.1 System Model and Assumptions

In our work, we consider a cloud-based biometric-matching system involving three parties: the *database owner*, the *user*, the *cloud server* (as illustrated in Fig. 1). In this application scenario, we assume a database owner holding a database $\mathcal{D}$ that contains a collection of biometric data $\langle \mathbf{b}_i \rangle_{i=1}^{m}$ (*e.g.*, fingerprints, voice patterns, palmprints, and facial patterns etc.), which are associated with certain profile information $\langle p_i \rangle_{i=1}^{m}$ (e.g., name, age and criminal record etc.). Before outsouring $\mathcal{D}$ to the remote cloud server, the database owner first preprocesses $\mathcal{D}$ to generate its encrypted form $\mathcal{C}$ and sends it to the cloud for storage. With a candidate biometric image, a user first locally derives its corresponding feature vector and sends the identification query to the database owner. After receiving the query, the database server generates an encrypted query and sends it to the cloud server. Subsequently, the cloud server executes the encrypted identification query over the encrypted database $\mathcal{C}$ and finally returns all the candidate matching results (*i.e.*, hitting encrypted FingerCodes and profiles) to the database owner. Finally, the database owner filters the results based on certain similarity threshold and compute the final output for the user.

More specifically, we assume that both the database owner's biometric data and the user's candidate biometric reading (*e.g.*, fingerprint images) have been processed such that the representations are suitable for biometric matching, *i.e.*, each raw biometric image is pre-processed by some widely-used feature extraction algorithms. Without loss of generality, we follow [8,20] and target fingerprint identification using FingerCodes [10] in our work. In our system, a FingerCode of a fingerprint consists of n elements with size l-bit (typically $n = 640$ and $l = 8$). For two FingerCodes $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$, they are considered a good match if the *Euclidean distance* between them is below a predefined threshold ε, which means that the fingerprints can be considered good candidates from the same person if

$$\|\mathbf{x} - \mathbf{y}\| < \varepsilon. \tag{1}$$

Therefore, the process of identifying a candidate (encrypted) fingerprint and its corresponding profile from a (encrypted) database of fingerprints can be divided into three steps: *secure Euclidian-distance computation*, *top-matching fingerprint determination* and *result filtering and retrieval.* The first and the second steps are executed on the cloud server side, and the third step is executed on the database owner side.

In the cloud-based biometric identification system, the encrypted database and the time-consuming biometric identification tasks are outsourced to the cloud. The system is expected to provide privacy-preserving biometric identification without disclosing any information about the database owner's biometric data to the cloud server and/or the user, and without disclosing any information about the user's biometric data (*i.e.*, query feature vectors) to the cloud (if no collusion exists between the cloud server and the user). We assume the cloud server and the user are semi-trusted, *i.e.*, they will execute the protocol as specified but may try to learn additional information from the encrypted biometric data and all the intermediate results generated during the protocol execution. In particular, under certain circumstances, we assume that the cloud server and the user may collude with each other and try to uncover the encrypted database.

2.2 Threat Model

In our model, we assume that the adversary knows the encryption scheme except the secret key. From a practical point of view, real-life adversaries have different level of background information and capabilities. To this end, we carefully define the threat model and characterize the attack-specific capabilities of adversary in three different scenarios. The *Attack Scenario 1* reflects the very practical case. In the *Attack Scenario 2*, the cloud server knows a set of plaintexts of the database but he does not know the corresponding encrypted values. In the *Attack Scenario 3*, the cloud server may collude with the user. Obviously, attackers in *Attack Scenario 2* and *Attack Scenario 3* are more powerful than that in *Attack Scenario 1*, but there is no higher or lower form of security level between *Scenario 2* and *Scenario 3*.

Attack Scenario 1: The cloud server is semi-trusted and can be seen as the adversary. It observes only the encrypted database $\mathcal{C}$ and all encrypted biometric identification queries. This model is similar to the well-known ciphertext-only attack (COA) model [11] used in the security evaluation of data encryption protocols. In practice, there are applications only accessed by secluded users but others can hardly observe any information other than the encrypted data.

Attack Scenario 2: On the basis of *Attack Scenario 1*, we assume that the adversary has some samples of the database in plaintext but he does not know the corresponding encrypted values. This corresponds to the known-sample attack in database literature [13]. For example, the attacker observes the encrypted database and some of his sources are clients who have been collected fingerprints by the government, it then knows the values of several records in the plaintext database.

Attack Scenario 3: On the basis of *Attack Scenario 1*, we assume that the cloud server and the user may collude with each other. Thus, in addition to the encrypted biometric database the adversary can arbitrarily choose the user's biometric identification query of interest for encryption and execute the encrypted query over $\mathcal{C}$. Considering this attack model is also necessary in some application cases. For example, it is possible for the cloud service provider to act as a user to submit fingerprint information for identification, so it can observe and even control the content of users' candidate FingerCode.

Definition 1. *A biometric identification outsourcing scheme is secure under Attack Scenario α ($\alpha \in \{1, 2, 3\}$) if no adversary can learn any other information from the encrypted biometric database $\mathcal{C}$ and the encrypted identification queries besides what it has already known.*

Remarks. The above security definition takes both collusion and no-collusion cases into account. The cloud server can observe the encrypted biometric database and the encrypted identification queries. It should be noted that if a scheme is secure under both *Attack Scenario 2* and *Attack Scenario 3*, it does not mean that the cloud server can both collude with the user and simultaneously observe some plaintexts of the database. This attack is too strong that as far as we know there exist no effective schemes that can defend against this sophisticated attack. In the following discussion, we show that our scheme achieves a well balance between efficiency and security requirements.

3 Privacy-Preserving Biometric Identification: An Examination of the State-of-the-Art

In this section, we provide an examination of two most closely-related works, which reflect the most recent progress and results in privacy-preserving biometric identification. A careful analysis of these solutions (in terms of both the model and the security strength) motivates us to seek new solutions.

3.1 The Biometric Identification Scheme of Huang et al.

Huang et al. [8] explored the privacy-preserving biometric identification problem in the secure two-party computation model, where the database owner holds the biometric database locally, and the database owner and the user carry the burden of expensive identification jobs. This is completely different from our outsourcing computation model, where we propose to take full advantage of cloud to take away the burden of storing, maintaining and performing computations over the extremely large database (*e.g.*, billions of biometric data and profiles). In particular, the biometric identification outsourcing model assumes the semi-trusted cloud server and users, from which the privacy of biometric database should be protected while enabling effective identification over the encrypted database. This makes the privacy-preserving biometric identification approach in [8] unsuitable for our application model.

Firstly, as reported from the experiment in [8], for a biometric database with one million FingerCodes (a vector of 16 8-bit integers), the corresponding profile size (including photos and other personal information) is around 2 TB (assuming each profile is 2 MB approximately). By using a local machine with an Intel Xeon CPU running at 2.0 GHz and a memory of 4 GB as the database owner, it then requires 1.88 h to run the protocol for each identification query (besides the preparation time of more than 100 h). When the database expands to 1 billion FingerCodes with 2000 TB profiles (the setting is also practical in the real world), the identification time that linearly increases with the database size will be about 78 days. This is apparently unbearable for both the database owner and the user.

Secondly, the entire encrypted biometric database should be transmitted to the user for each identification query, which leads to extremely large communication overhead. For a database of size 5 GB (including the encrypted profiles), according to the experimental results in [8], it will take about 3.5 GB bandwidth to complete the transmission when 100 queries are arriving simultaneously. Finally, we show that even if there is a powerful semi-trusted cloud server the database owner can cooperate with, Huang et al.'s [8] encryption method is still not applicable in the outsourcing computation model due to its ineffective use of cloud. Specifically speaking, in the Euclidean-Distance Protocol of [8], the squared Euclidean distance d_i between $\mathbf{v}_i$ (one of the vectors in the database) and $\mathbf{v}'$ (candidate vector) is computed as follows

$$d_i = ||\mathbf{v}_i - \mathbf{v}'||^2 = \sum_{j=1}^{N}(v_{i,j} - v'_j)^2$$
$$= \sum_{j=1}^{N} v_{i,j}^2 + \sum_{j=1}^{N}(-2v_{i,j}v'_j) + \sum_{j=1}^{N} v'^2_j.$$

Let $S_{i,1} = \sum_{j=1}^{N} v_{i,j}^2$, $S_{i,2} = \sum_{j=1}^{N}(-2v_{i,j}v'_j)$ and $S_{i,3} = \sum_{j=1}^{N} v'^2_j$. Obviously, the encrypted $S_{i,1}$, denoted by $[S_{i,1}]$, can be outsourced to the cloud server for storage space savings and free of data maintenance. By homomorphic encryption, we have

$$[S_{i,2}] = [\sum_{j=1}^{N}(-2v_{i,j}v'_j)] = \prod_{j=1}^{N}[-2v_{i,j}]^{v'_j}. \quad (2)$$

Then, the database owner can outsource $[-2v_{i,j}]$ for $1 \leq j \leq N$ and $1 \leq i \leq M$ to the cloud, where M denotes the database size. These are one-time executions and can be done in the preparation phase. However, when the database owner receives the plaintext identificatrion query v' from the user, he has to download all $[-2v_{i,j}]$ from the cloud server because v' appears in the plaintext form in the computation of $[S_{i,2}]$ (as shown in Eq. (2)) and cannot be encrypted and outsourced to the cloud. Then the database owner has to traverse the database to compute $[S_{i,2}]$ for each $1 \leq i \leq M$ according to Eq. (2). It is obvious that the cloud server cannot free the database owner from the burden of heavy computations. Therefore, we claim that Huang et al. scheme cannot be directly extended to our biometric identification outsourcing model.

3.2 The Biometric Identification Scheme of Yuan et al.

Yuan et al. [20] investigated the biometric identification outsourcing problem under the same system model as ours, and they claimed that their scheme is highly efficient and secure under the chosen message attack model. Roughly, their main idea is to encrypt each extended FingerCode by multiplying randomly-selected matrices and exploit properties embedded in these matrices for Euclidian distance computations. Yuan et al. claimed their scheme is resilient to the *Attack Scenario 3* when the unknowns (associated with the secret keys and data) are less than the system of equations built by the adversary. However, we show that this is not the case at all!

We first describe their scheme in further detail. Let the i-th ($i \in [1, m]$) fingerprint in the database be a n-dimensional feature vector, the database owner generates its FingerCode as $\mathbf{b}_i = [b_{i1}, b_{i2}, \ldots, b_{in}]$, where b_{ik} ($k \in [1, n]$) is an 8-bit integer. To facilitate identification, each feature vector is extended to $\mathbf{B}_i = [b_{i1}, b_{i2}, \ldots, b_{in}, b_{i(n+1)}]$, where $b_{i(n+1)} = -\frac{1}{2}(b_{i1}^2 + b_{i2}^2 + \ldots + b_{in}^2)$. Then, $\mathbf{B}_i$ is used to generate matrix $\mathbf{B}'_i$ as

$$\mathbf{B}'_i = \begin{pmatrix} b_{i1} & 0 & \ldots & 0 \\ 0 & b_{i2} & \ldots & 0 \\ \vdots & \vdots & \ldots & \vdots \\ 0 & \ldots & 0 & b_{i(n+1)} \end{pmatrix}. \quad (3)$$

The database owner randomly generates two $(n+1)$-dimensional vectors $\mathbf{H} = [h_1, h_2, \ldots, h_{n+1}]$ and $\mathbf{R} = [r_1, r_2, \ldots, r_{n+1}]$, and it also randomly generates three $(n+1) \times (n+1)$ invertible matrices $\mathbf{M}_1, \mathbf{M}_2$ and $\mathbf{M}_3$ as secret keys. For each $\mathbf{B}'_i$, a random $(n+1) \times (n+1)$ matrix $\mathbf{A}_i$ is generated to hide $\mathbf{B}'_i$ by

$$\mathbf{D}_i = \mathbf{A}_i^T \mathbf{B}'_i, \tag{4}$$

where $\mathbf{A}_{ik} = [a_{ik1}, a_{ik2}, \ldots, a_{ik(n+1)}]$ is the row vector of $\mathbf{A}_i$ $(k \in [1, n+1])$. It has the property $\mathbf{H}\mathbf{A}_{ik}^T = \sum_{j=1}^{n+1} h_j a_{ikj} = 1$. This implies

$$\mathbf{H}\mathbf{A}_i^T = (1, 1, \ldots, 1). \tag{5}$$

The database owner further encrypts $\mathbf{H}$, $\mathbf{R}$ and $\mathbf{D}_i$ with $\mathbf{M}_1$, $\mathbf{M}_2$ and $\mathbf{M}_3$ as

$$\begin{aligned} \mathbf{C}_i &= \mathbf{M}_1 \mathbf{D}_i \mathbf{M}_2, \\ \mathbf{C}_\mathbf{H} &= \mathbf{H}\mathbf{M}_1^{-1}, \\ \mathbf{C}_\mathbf{R} &= \mathbf{M}_3^{-1}\mathbf{R}^T. \end{aligned} \tag{6}$$

After encryption, a *Index* I_i is built for each FingerCode $\langle \mathbf{b}_i, \mathbf{C}_i \rangle$. Finally, $(I_i, \mathbf{C}_i, \mathbf{C}_\mathbf{H}, \mathbf{C}_\mathbf{R})$ is uploaded to the cloud.

When the user has a candidate fingerprint to be identified, it extends its corresponding FingerCode $\mathbf{b}_c = [b_{c1}, b_{c2}, \ldots, b_{cn}]$ to a $(n+1)$-dimensional vector $\mathbf{B}_c = [b_{c1}, b_{c2}, \ldots, b_{cn}, 1]$. Then $\mathbf{B}_c$ is transferred to $\mathbf{B}'_c$ as in Eq. (3). Finally, the database owner disguises $\mathbf{B}'_c$ by multiplying a $(n+1) \times (n+1)$ random matrix $\mathbf{E}_c$

$$\mathbf{F}_c = \mathbf{B}'_c \mathbf{E}_c,$$

where $\mathbf{E}_{ck} = [e_{ck1}, e_{ck2}, \ldots, e_{ck(n+1)}]$ $(k \in [1, n+1])$. Similar to $\mathbf{A}_{ik}$, it also has the property $\mathbf{E}_{ck}\mathbf{R}^T = \sum_{j=1}^{n+1} r_j e_{ckj} = 1$, which implies that $\mathbf{E}_c\mathbf{R}^T = (1, 1, \ldots, 1)^T$. The database owner further blinds $\mathbf{F}_c$ with secret keys $\mathbf{M}_2$ and $\mathbf{M}_3$ to generate the encrypted identification query $\mathbf{C}_\mathbf{F}$ as

$$\mathbf{C}_\mathbf{F} = \mathbf{M}_2^{-1} \mathbf{F}_c \mathbf{M}_3.$$

Then, $\mathbf{C}_\mathbf{F}$ is submitted to the cloud for identification. Finally, on receiving $\mathbf{C}_\mathbf{F}$, the cloud server compares Euclidean distance between $\mathbf{b}_i$ and $\mathbf{b}_c$ by computing $\mathbf{P}_i = \mathbf{C}_\mathbf{H}\mathbf{C}_i\mathbf{C}_\mathbf{F}\mathbf{C}_\mathbf{R}$ (We eliminate other details since they are irrelevant for the attacks we will describe).

In the following analysis, we show that there exist inherent flaws in the above design. The cloud server can exploit these flaws to figure out $\mathbf{M}_2$, and further to recover the database owner's FingerCodes $\mathbf{b}_i$ or $\mathbf{B}_i$ $(i = 1, \ldots, m)$.

Our attacks rely on the knowledge of a set of plaintext-ciphertext pairs $(\mathbf{B}_i, \mathbf{C}_i)$, *i.e.*, the known-plaintext attack model. Yuan et al. [20] even claimed that a number of users can collude with the cloud server to choose arbitrary plaintexts of candidate queries and obtain the corresponding ciphertexts (*i.e.*, the known-candidate attack defined in [20]). However, we show this is not true.

Attack on the Encrypted Biometric Database by Eliminating Randomness. See Appendix A.

Attack on the Encrypted Biometric Database by Exploiting Euclidian Distance Results. See Appendix B.

4 Our Construction: The New and Improved Solutions

In this work, our ultimate goal is to enable privacy-preserving biometric identification of the encrypted biometric database stored in the cloud, finding the top matching identities for users. We have the following specifical goals. First, correctness, *i.e.*, the correctness of the identification results should be guaranteed. Second, privacy assurance, *i.e.*, the privacy of biometric data should be protected from the adversary. Third, efficiency, *i.e.*, the computation efficiency of the privacy-preserving biometric identification protocol should be practically high.

Keep the above design goals in mind, in this section we first present a cloud-based privacy-preserving biometric matching scheme secure under the *Attack Scenario 2*. We named this basic scheme as CloudBI-I. Then, we propose an enhanced version named CloudBI-II, which achieves security under both the *Attack Scenario 2* and *Attack Scenario 3* and show how it effectively avoids flaws of [20].

4.1 CloudBI-I: The Basic Scheme

Database Encryption Phase. The database owner pre-processes each fingerprint image for which a feature vector FingerCode $\mathbf{b}_i$ is generated. For each n-dimensional FingerCode $\mathbf{b}_i = [b_{i1}, b_{i2}, \ldots, b_{in}]$ ($\mathbf{b}_i \in \langle \mathbf{b}_i \rangle_{i=1}^m$ and typically $n = 640$), it is extended to a $(n+2)$-dimensional vector $\mathbf{B}_i = [b_{i1}, b_{i2}, \ldots, b_{in}, b_{i(n+1)}, 1]$, where $b_{i(n+1)} = -\frac{1}{2}(b_{i1}^2 + b_{i2}^2 + \ldots + b_{in}^2)$. Then $\mathbf{B}_i$ is transferred to a $(n+2) \times (n+2)$ matrix $\mathbf{B}_i'$ with the similar form in Eq. (3). To encrypt the biometric data, the database owner randomly generates two $(n+2) \times (n+2)$ invertible matrices $\mathbf{M}_1$ and $\mathbf{M}_2$. Then for each $\mathbf{B}_i'$, it computes

$$\mathbf{C}_i = \mathbf{M}_1 \mathbf{B}_i' \mathbf{M}_2$$

Given a security parameter k, call $sk \leftarrow \mathsf{SKE.KeyGen}(1^k, r)$, where r is a random number and SKE is a PCPA-secure symmetric encryption scheme. Let $\mathbf{c_p} \leftarrow \mathsf{Enc}(sk, \mathbf{p})$, where $\mathbf{p} = \langle p_i \rangle_{i=1}^m$ is the set of profiles.

After encryption, the tuple $\langle \mathbf{C}_i, \mathbf{c}_{\mathbf{p}_i} \rangle_{i=1}^m$ is uploaded to the cloud.

Biometric Data Matching Phase. The user has a candidate fingerprint (image) to be identified. To this end, it sends the corresponding FingerCode $\mathbf{b}_c = [b_{c1}, b_{c2}, \ldots, b_{cn}]$ to the database owner, who will extend the FingerCode to $\mathbf{B}_c = [b_{c1}, b_{c2}, \ldots, b_{cn}, 1, r_c]$, where r_c is a random value generated by the database owner. Note that r_c is chosen independently for each $\mathbf{b}_c$. Similarly, $\mathbf{B}_c$ is extended to the matrix $\mathbf{B}'_c$ with the form in Eq. (3). To encrypt the identification query, the database server computes

$$\mathbf{C_F} = \mathbf{M}_2^{-1}\mathbf{B}'_c\mathbf{M}_1^{-1}.$$

The encrypted query $\mathbf{C_F}$ is then sent to the cloud server for identification.

To compare the Euclidean distances between each encrypted FingerCode $\mathbf{C}_i$ ($i = 1, \ldots, m$) and $\mathbf{C_F}$, the cloud server first computes

$$\mathbf{P}_i = \mathbf{C}_i\mathbf{C_F} = \mathbf{M}_1\mathbf{B}'_i\mathbf{B}'_c\mathbf{M}_1^{-1}.$$

Then the cloud server computes the eigenvalues of $\mathbf{P}_i$, denoted by λ_j ($j = 1, \ldots, n+2$), by solving the equations $|\lambda I_{(n+2)} - P_i| = 0$, where $I_{(n+2)}$ is the $(n+2) \times (n+2)$ identity matrix. Let T_i denote the *trace* of $\mathbf{P}_i$, we have

$$T_i = \mathsf{tr}(\mathbf{P}_i) = \sum_{j=1}^{n+2} \lambda_j.$$

Finally, the cloud server only needs to rank T_i ($i = 1, \ldots, m$) to find out the minimum k traces or the minimum one (*i.e.*, $k = 1$). For ease of exposition, we consider the $k = 1$ case in the following discussion. Assume $\mathbf{C}_{i^*}$ is the encrypted biometric data that achieves the minimum, and its corresponding profile is denoted by p_{i^*}. Finally, the cloud server sends $(\mathbf{C}_{i^*}, \mathbf{c}_{\mathbf{p}_{i^*}})$ back to the database owner.

Result Filtering and Retrieval Phase. After receiving $(\mathbf{C}_{i^*}, \mathbf{c}_{\mathbf{p}_{i^*}})$, the database owner decrypts $\mathbf{C}_{i^*}$ to have $\mathbf{B}'_{i^*} = \mathbf{M}_1^{-1}\mathbf{C}_{i^*}\mathbf{M}_2^{-1}$. Then it transform $\mathbf{B}'_{i^*}$ to $\mathbf{B}_{i^*}$ and the plaintext FingerCode $\mathbf{b}_{i^*}$. Finally, it computes the actual Euclidean distance between $\mathbf{b}_{i^*}$ and $\mathbf{b}_c$. By checking $\|\mathbf{b}_{i^*} - \mathbf{b}_c\| < \varepsilon$, the database owner decrypts $\mathbf{c}_{\mathbf{p}_{i^*}}$ to have p_{i^*} and sends it to the user if it holds. Otherwise, it outputs $\perp$.

Correctness Analysis. In linear algebra, the transformation $\mathbf{B}'_i\mathbf{B}'_c \mapsto \mathbf{M}_1(\mathbf{B}'_i\mathbf{B}'_c)\mathbf{M}_1^{-1}$ is called a *similarity transformation*. Based on the properties of *similar matrices*, the *trace* is *similarity-invariant*, which means that two similar matrices have the same *trace*, *i.e.*, $\mathsf{tr}(\mathbf{P}_i) = \mathsf{tr}(\mathbf{B}'_i\mathbf{B}'_c)$. We now compute the trace of $\mathbf{B}'_i\mathbf{B}'_c$, denoted by $\mathsf{tr}(\mathbf{B}'_i\mathbf{B}'_c)$. As can be seen, $\mathbf{B}'_i\mathbf{B}'_c$ has the following structure

$$\mathbf{B}'_i\mathbf{B}'_c = \begin{pmatrix} b_{i1}b_{c1} & 0 & 0 & \ldots & 0 & 0 \\ 0 & b_{i2}b_{c2} & 0 & \ldots & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & b_{in}b_{cn} & 0 & 0 \\ 0 & 0 & \ldots & 0 & -0.5\sum_{j=1}^{n} b_{ij}^2 & 0 \\ 0 & 0 & \ldots & 0 & 0 & r_c \end{pmatrix}. \tag{7}$$

By the definition of *trace* and *similarity-invariance* property we have

$$T_i = \mathsf{tr}(\mathbf{B}'_i\mathbf{B}'_c) = \sum_{j=1}^{n} b_{ij}b_{cj} - 0.5\sum_{j=1}^{n} b_{ij}^2 + r_c. \tag{8}$$

Let dist_{ic} and dist_{zc} denote the Euclidean distance between FingerCode $\mathbf{b}_i$ and query $\mathbf{b}_c$, the Euclidean distance between FingerCode $\mathbf{b}_z$ and query $\mathbf{b}_c$, respectively. Then we have

$$\mathsf{dist}_{ic}^2 - \mathsf{dist}_{zc}^2 = \sum_{j=1}^{n}(b_{ij} - b_{cj})^2 - \sum_{j=1}^{n}(b_{zj} - b_{cj})^2.$$

We expand the above expression and re-arrange them to have

$$\begin{aligned}\mathsf{dist}_{ic}^2 - \mathsf{dist}_{zc}^2 &= 2(\sum_{j=1}^{n} b_{zj}b_{cj} - 0.5\sum_{j=1}^{n} b_{zj}^2 + r_c)\\ &\quad - 2(\sum_{j=1}^{n} b_{ij}b_{cj} - 0.5\sum_{j=1}^{n} b_{ij}^2 + r_c) \\ &= 2(\mathsf{tr}(\mathbf{B}'_z\mathbf{B}'_c) - \mathsf{tr}(\mathbf{B}'_i\mathbf{B}'_c))\\ &= 2(T_z - T_i)\end{aligned} \tag{9}$$

Based on Eq. (9), the cloud server can determine $\mathsf{dist}_{ic} \geq \mathsf{dist}_{zc}$ if $\mathsf{T}_z \geq \mathsf{T}_i$; otherwise $\mathsf{dist}_{zc} < \mathsf{dist}_{ic}$. After repeating this checking process for all the encrypted FingerCode $\mathbf{C}_i$'s, the cloud server is able to find out $\mathbf{b}_{i^*}$ (in encrypted form) that has the minimum distance to $\mathbf{b}_c$ (in encrypted form).

Security Analysis

Theorem 1. *Our CloudBI-I scheme is secure under the Attack Scenario 2.*

Proof. Due to the space limitation, please refer to our technical report [18] for the full proof.

4.2 CloudBI-II: The Enhanced Scheme

In the previous section, we have proved that CloudBI-I is secure under the *Attack Scenario 2*. However, it can be broken under the *Attack Scenario 3*. Specifically, in the equation $\mathbf{P}_i = \mathbf{C}_i\mathbf{C}_\mathbf{F} = \mathbf{M}_1\mathbf{B}'_i\mathbf{B}'_c\mathbf{M}_1^{-1}$, as $\mathbf{B}'_i\mathbf{B}'_c$ is a diagonal matrix and the eigenvalues of a diagonal matrix is equal to its main diagonal elements, the cloud server can establish equation $\lambda_j = b_{ij}b_{cj}$. When there exists the collusion between the cloud server and the user, which means b_{cj} can be obtained by the cloud server, it then can work out b_{ij} and thus get all B_i's. Therefore CloudBI-I is not secure under *Attack Scenario 3*. Besides, some approximate information may be leaked to the adversary. For example, an attacker may formulate a system of equations as a simultaneous Diophantine approximate problem [7] and will find some approximate values corresponding to q'_{11} and p'_{1k} ($k = 1, \ldots, n{+}2$). To solve the above problems and achieve higher security strength, we further propose an improved privacy-preserving biometric identification outsourcing scheme. The main idea of the enhanced scheme is to introduce more randomness in the encryptions of the biometric data and the biometric identification query. Consequently, it is impossible to derive any information about $\mathbf{M}_1$ and $\mathbf{M}_2$.

The key difference between CloudBI-II and CloudBI-I is that the database owner will multiply $\mathbf{B}'_i$ and $\mathbf{B}'_c$ each by an additional random triangular matrix during the encryption process. In the *database encryption phase*, for each $\mathbf{B}'_i$ ($i = 1, \ldots, m$), the database owner encrypts it as

$$\mathbf{C}_i = \mathbf{M}_1 \mathbf{Q}_i \mathbf{B}'_i \mathbf{M}_2,$$

where $\mathbf{Q}_i$ is a randomly-generated *lower triangular matrix* with diagonal entries set to 1.

In the *biometric data matching phase*, for each identification query $\mathbf{B}_c$ the database owner randomly generates a *lower triangular matrix* $\mathbf{Q}_c$ with diagonal entries set to 1, and it encrypts the plaintext query as

$$\mathbf{C}_\mathbf{F} = \mathbf{M}_2^{-1} \mathbf{B}'_c \mathbf{Q}_c \mathbf{M}_1^{-1}.$$

The remaining operations for the *result filtering and retrieval phase* are the same as the basic scheme.

Correctness Analysis. We show that the new probabilistic encryption algorithm will not affect the correctness of the final results. After receiving $\mathbf{C}_i$ and $\mathbf{C}_\mathbf{F}$, the cloud server computes

$$\begin{aligned}\mathbf{P}_i = \mathbf{C}_i \mathbf{C}_\mathbf{F} &= \mathbf{M}_1 \mathbf{Q}_i \mathbf{B}'_i \mathbf{M}_2 \mathbf{M}_2^{-1} \mathbf{B}'_c \mathbf{Q}_c \mathbf{M}_1^{-1} \\ &= \mathbf{M}_1 \mathbf{Q}_i \mathbf{B}'_i \mathbf{B}'_c \mathbf{Q}_c \mathbf{M}_1^{-1}.\end{aligned}$$

Due to the property of *similarity transformation*, $T_i = \mathsf{tr}(\mathbf{P}_i) = \mathsf{tr}(\mathbf{Q}_i \mathbf{B}'_i \mathbf{B}'_c \mathbf{Q}_c)$. In linear algebra, the product of a *diagonal matrix* and a *lower triangular matrix* is also *lower triangular*. Thus, we have

$$\mathbf{Q}_i \mathbf{B}'_i = \begin{pmatrix} b_{i1} & 0 & 0 & \cdots & 0 & 0 \\ t_{21} & b_{i2} & 0 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ t_{n1} & \cdots & t_{n(n-1)} & b_{in} & 0 & 0 \\ t_{(n+1)1} & t_{(n+1)2} & \cdots & t_{(n+1)n} & -\frac{1}{2}\sum_{j=1}^{n} b_{ij}^2 & 0 \\ t_{(n+2)1} & t_{(n+2)2} & \cdots & t_{(n+2)n} & t_{(n+2)(n+1)} & 1 \end{pmatrix}, \tag{10}$$

where t_{ij} are random values. It shows that the multiplication of $\mathbf{B}'_i$ by $\mathbf{Q}_i$ does not change its main diagonal entries.

Following the same reason, the multiplication of $\mathbf{B}'_i \mathbf{B}'_c$ by $\mathbf{Q}_i$ and $\mathbf{Q}_c$ respectively will not change the main diagonal entries of $\mathbf{B}'_i \mathbf{B}'_c$ (as shown in Eq. (7)). It indicates that $\mathsf{tr}(\mathbf{Q}_i \mathbf{B}'_i \mathbf{B}'_c \mathbf{Q}_c)$ is also equal to $\sum_{j=1}^{n} b_{ij} b_{cj} - 0.5 \sum_{j=1}^{n} b_{ij}^2 + r_c$. Thus, Eq. (9) still holds. So, the cloud server can return the target $(\mathbf{C}_{i^*}, \mathbf{c}_{\mathbf{p}_{i^*}})$ to the database owner.

Table 1. A summary of complexity costs. Here, m denotes the number of $\langle FingerCode, profile\rangle$ pairs in the database; k denotes the number of closest fingerprints required to be returned, *e.g.*, $k = 1$ if the closest one is returned; $n \ll m$.

		Phases	Yuan et al. scheme [20]	CloudBI-I	CloudBI-II
Comp.	Database owner	**Prep.**	$\mathcal{O}(mn^3)$	$\mathcal{O}(mn^3)$	$\mathcal{O}(mn^3)$
		Iden.	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^3)$
		Retr.	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$
	User	**Iden.**			
	Cloud server	**Iden.**	$\mathcal{O}(mn^2 + m\log m)$	$\mathcal{O}(mn^3 + m\log m)$	$\mathcal{O}(mn^3 + m\log m)$
Comm.	Database owner	**Prep.**	$\mathcal{O}(mn^2)$	$\mathcal{O}(mn^2)$	$\mathcal{O}(mn^2)$
		Iden.	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$
		Retr.	$\mathcal{O}(k)$	$\mathcal{O}(k)$	$\mathcal{O}(k)$
	User	**Iden.**	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
	Cloud server	**Iden.**	/	/	/
		Retr.	$\mathcal{O}(k)$	$\mathcal{O}(k)$	$\mathcal{O}(k)$
Security	*Attack Scenario 2*		Yes	Yes	Yes
	Attack Scenario 3		No	No	Yes

Security Analysis. Apparently, the multiplication of random matrices will not compromise the security of CloudBI-I, thus the enhanced scheme CloudBI-II is also secure under the *Attack Scenario 2*. For the *Attack Scenario 3*, we have the following theorem.

Theorem 2. *Our CloudBI-II scheme is secure under the Attack Scenario 3.*

Proof. Due to the space limitation, please refer to our technical report [18] for the full proof.

Remarks. By the introduction of random matrices $\mathbf{Q}_i$ and $\mathbf{Q}_c$, CloudBI-II makes it impossible for the adversary to apply simultaneous Diophantine approximation attack. Specifically, according to CloudBI-II, all the relevant equations that involves p_{11} can be listed by the adversary are

$$\begin{cases} p_{11}q_{11} + t_{21}p_{12}q_{11}, \ldots + t_{(n+2)1}p_{1(n+2)}q_{11} = \mathbf{C}_{i1(11)} \\ p_{11}q_{12} + t_{21}p_{12}q_{12}, \ldots + t_{(n+2)1}p_{1(n+2)}q_{12} = \mathbf{C}_{i1(12)} \\ \ldots \\ p_{11}q_{1(n+2)} + t_{21}p_{12}q_{1(n+2)}, \ldots + t_{(n+2)1}p_{1(n+2)}q_{1(n+2)} \\ = \mathbf{C}_{i1(1(n+2))}. \end{cases} \tag{11}$$

However, in Eq. (11), there are $n + 2$ equations with $3n + 5$ unknowns in total such that p_{11} cannot be determined or even approximately solved.

5 Implementation and Evaluation

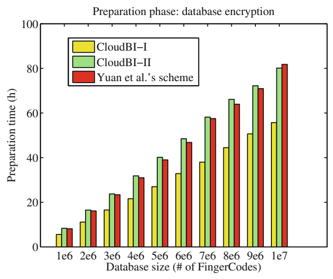
We evaluate the performance of our protocols by implementing a privacy-preserving fingerprint identification system. We set up the cloud with 1000 nodes, each with an Intel Core i5-4440M 2.80 GHz CPU and 16 GB memory. We set up the database owner on a separate machine with the same hardware configuration. We randomly generate 640-entry vectors to construct the FingerCode database following [8,20], and the database size ranges from one million to ten million $\langle FingerCode, profile \rangle$ pairs. We also randomly select a sequence of feature vectors as the query FingerCodes.

5.1 Complexity Analysis

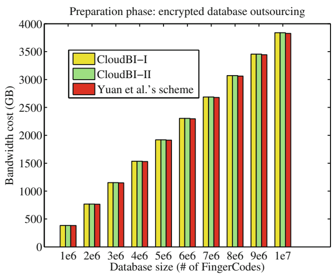
Before delveing into the experimental results, we first provide an overview of the complexity of the privacy-preserving fingerprint identification execution on all three participating parties, in terms of computation and communication overheads. Table 1 summarizes the complexities for our system (including CloudBI-I and CloudBI-II) and for the biometric identification system proposed in [20]. Here, we eliminate the discussion of Huang et al. [8], who essentially considered the two-party secure computation model (see Sect. 3.1 for further details of its extension to the computation outsourcing model). The *preparation phase*, the *identification phase* and the *retrieval phase* are corresponding to the three phases (as described before) during the protocol execution. Note that, we assume each matrix multiplication has time complexity of $\mathcal{O}(n^3)$, where n is the dimension of a FingerCode. $\mathcal{O}(m \log m)$ is the sorting cost of fuzzy Euclidian distances. It is worth noting that although the computation and communication complexities grow linearly with the database size, these are one-time costs that will not influence the real-time performance of the biometric identification process. Our system focuses on outsourcing of the storage and computation workloads to the cloud for utilizing its high storage and computation capacity. In practice, our privacy-preserving fingerprint identification protocol allows the identification process to be performed in parallel on multiple cloud instances, which can ensure the efficiency of the identification process even with a very large-scale fingerprint database.

5.2 Experimental Evaluation

Preparation Phase. Figure 2 shows the time cost and the bandwidth cost in the preparation phase. Note that, both costs are one-time startup costs. Not surprisingly, the database encryption time and the communication cost for outsourcing to the cloud increase linearly with the database size (*i.e.*, the number of FingerCodes contained in the database). The experimental results conform to the theoretical results in Table 1, which shows that CloudBI-I and CloudBI-II have the same computation complexity with [20]. As CloudBI-I has less matrix multiplication operations than [20], it can save about 33 % time cost for biometric database encryption. The bandwidth consumptions of three schemes, as shown



(a)

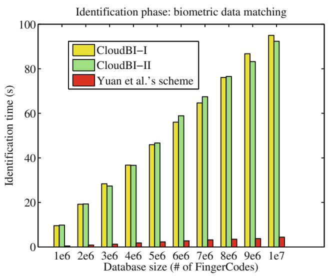


(b)

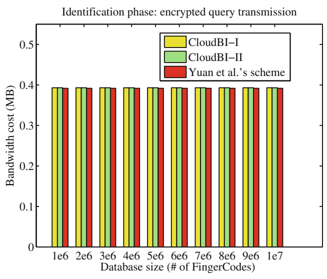
Fig. 2. Preparation phase: (a) Time costs for different sizes of database; (b) Bandwidth costs for different sizes of database.

in Fig. 2(b), are almost the same. As suggested in practical applications, hard disks can be used to drive the outsourced encrypted data transmission services offered by cloud service provider (*e.g.*, Amazon) to save bandwidth consumption.

Identification Phase. Figure 3 shows the time cost and the bandwidth cost in the identification phase. As demonstrated in Fig. 3(a), for a single query, with the increase of database size, the biometric data matching time of our schemes and [20] are linear functions of the database size. In the identification phase, the computation cost of [20] are far less than CloudBI-II (*i.e.*, mn^2 *vs.* mn^3). However, we **emphasize** that Yuan et al. [20] have not noticed that substantial security is sacrificed for achieving such fast computation of $\mathbf{P}_i$ in [20], where matrix multiplications are transformed to vector-matrix multiplications. As discussed in Sect. 3.2, we launch successful attacks on [20] by leveraging this weakness (see Eq. (12)). For bandwidth consumption of a single query, the cost is constant (*e.g.*, 400 KB in our experimental setting) as shown in Fig. 3(b). In our system, a query request can be processed in parallel on the cloud side. A set of computing instances in the cloud can be used to handle biometric data matching on distinct small portions of database in parallel, and each of them can find out a candidate result. Finally, by comparing these candidate results, a cloud instance can figure out the final result. If simultaneous queries come, as shown in Fig. 4(a), the identification time increases linearly with the number of queries without putting additional workload. The above results clearly validate the scalability of our cloud-based privacy-preserving biometric identification system. To demonstrate the computation savings on the database owner, we show the comparison of time costs of biometric identification on the database owner with and without identification outsourcing in Fig. 4(b). As can be seen, our schemes achieves constant time cost on the database owner, while the time cost of performing identification locally (over plaintext biometric database without outsourcing) increases linearly with the database size. The larger the database size (*e.g.*, with 10 million FingerCodes) is, the higher efficiency gain can be achieved.

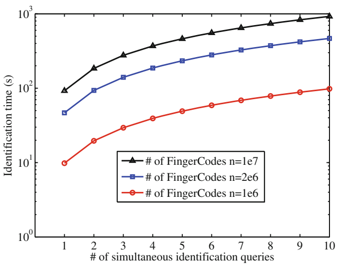


(a)

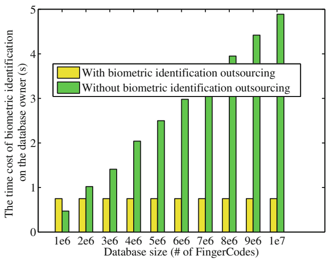


(b)

Fig. 3. Identification phase: (a) Time costs for different sizes of database; (b) Bandwidth costs for different sizes of database.



(a)



(b)

Fig. 4. a) Time costs under different number of simultaneous identification queries; (b) Comparison of time costs of biometric identification on the database owner.

6 Concluding Remarks

In this work, we investigated the privacy-preserving biometric identification outsourcing problem by developing new privacy-preserving biometric identification protocols. Our approaches enable efficient and privacy-preserving biometric matching with minimum database owner's involvement. Our experimental results show that the cloud-based biometric identification system is appropriate for use in various privacy-preserving biometric identification applications.

Acknowledgments. Kui's research is supported in part by US National Science Foundation under grant CNS-1262277. Qian's research is supported in part by National Natural Science Foundation of China (Grant No. 61373167), Natural Science Foundation of Hubei Province (Grant No. 2013CFB297), Wuhan Science and Technology Bureau (Grant No. 2015010101010020), National Basic Research Program of China (973 Program, Grant No. 2014CB340600) and National High Technology Research and Development Program of China (863 Program, Grant No. 2015AA016004). Zhibo's research is supported in part by Fundamental Research Funds for the Central Universities (No. 2042015kf0016).

A Attack on Yuan et al. [20] by Eliminating Randomness

We begin by describing an attack on the scheme as described above to recover $\mathbf{B}_i$ by eliminating the randomness in the encrypted biometric database.

Based on $(\mathbf{C}_i, \mathbf{C}_\mathbf{H})$, the server can eliminate the random matrix $\mathbf{A}_i$ and then derive the secret matrix $\mathbf{M}_2$ by computing

$$\begin{aligned}\mathbf{C}_\mathbf{H}\mathbf{C}_i &= (\mathbf{H}\mathbf{M}_1^{-1})(\mathbf{M}_1\mathbf{D}_i\mathbf{M}_2)\\ &= \mathbf{H}\mathbf{D}_i\mathbf{M}_2 = \mathbf{H}(\mathbf{A}_i^T\mathbf{B}_i')\mathbf{M}_2 = (\mathbf{H}\mathbf{A}_i^T)\mathbf{B}_i'\mathbf{M}_2 \\ &= (1,1,\ldots,1)\cdot\mathbf{B}_i'\mathbf{M}_2 = \mathbf{B}_i\mathbf{M}_2.\end{aligned} \tag{12}$$

Here, note that $(1,1,\ldots,1)\cdot\mathbf{B}_i' = \mathbf{B}_i$. In Eq. (12), since $\mathbf{M}_2$ is a $(n+1)\times(n+1)$ constant matrix, if the cloud server possesses $(n+1)$ *linearly independent* $\mathbf{B}_i$ and constructs $(n+1)$ equations, then $\mathbf{M}_2$ can be recovered. Once $\mathbf{M}_2$ is known, the cloud server can recover all $\mathbf{B}_i$'s $(i = 1,\ldots,m)$ according to Eq. (12). Next we give an illustrating example of this attack.

Assume that $n = 2$ and the database owner has $\mathbf{B}_i$ $(i = 1,\ldots,m)$, $\mathbf{H} = [1,2,3]$ and $\mathbf{R} = [1,1,2]$. The three randomly-generated secret matrices $\mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3$ are

$$\mathbf{M}_1 = \begin{pmatrix}1&2&0\\1&1&0\\0&0&1\end{pmatrix}, \mathbf{M}_2 = \begin{pmatrix}1&0&1\\0&2&1\\0&3&1\end{pmatrix}, \mathbf{M}_3 = \begin{pmatrix}2&0&0\\1&0&1\\0&1&2\end{pmatrix}.$$

Correspondingly, $\mathbf{A}_1, \mathbf{A}_2$ and $\mathbf{A}_3$ are generated to satisfy $\mathbf{H}\mathbf{A}_i^T = (1,1,\ldots,1)$ as

$$\mathbf{A}_1 = \begin{pmatrix}1&0&0\\-1&1&0\\0&-1&1\end{pmatrix}, \mathbf{A}_2 = \begin{pmatrix}0&2&-1\\1&0&0\\-1&1&0\end{pmatrix},$$
$$\mathbf{A}_3 = \begin{pmatrix}-1&-2&2\\1&0&0\\0&-1&1\end{pmatrix}.$$

Then we have

$$\mathbf{C}_\mathbf{H} = \mathbf{H}\mathbf{M}_1^{-1} = [1,0,3], \text{ where } \mathbf{M}_1^{-1} = \begin{pmatrix}-1&2&0\\1&-1&0\\0&0&1\end{pmatrix}$$

Without loss of generality, we assume $\mathbf{B}_1 = [1,3,-5], \mathbf{B}_2 = [0,2,-2]$ and $\mathbf{B}_3 = [2,2,-4]$. According to Eq. (4), we can compute $\mathbf{D}_i$. Based on Eq. (6), we have $\mathbf{C}_i$ $(i = 1,2,3)$

$$\mathbf{C}_1 = \begin{pmatrix}1&36&14\\1&15&6\\0&-15&-5\end{pmatrix}, \mathbf{C}_2 = \begin{pmatrix}0&-2&0\\0&4&2\\0&0&0\end{pmatrix},$$
$$\mathbf{C}_3 = \begin{pmatrix}-10&28&0\\-6&16&0\\4&-12&0\end{pmatrix}.$$

Then $(\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_3, \mathbf{C}_\mathbf{H})$ are sent to the cloud server (*i.e.*, the adversary in our model), who can construct the following equations using Eq. (12)

$$\begin{aligned}\mathbf{B}_1\mathbf{M}_2' &= \mathbf{C}_\mathbf{H}\mathbf{C}_1 = (1,-9,-1),\\ \mathbf{B}_2\mathbf{M}_2' &= \mathbf{C}_\mathbf{H}\mathbf{C}_2 = (0,-2,0),\\ \mathbf{B}_3\mathbf{M}_2' &= \mathbf{C}_\mathbf{H}\mathbf{C}_3 = (2,-8,0).\end{aligned}$$

According to the known-plaintext attack model (or the extension to the chosen-plaintext attack model) defined in [20], if we let the adversary (*i.e.*, the cloud server) observe $\mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3$, it can solve for

$$\mathbf{M}_2' = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 1 \\ 0 & 3 & 1 \end{pmatrix} = \mathbf{M}_2.$$

By plugging $\mathbf{M}_2$ into Eq. (12), then the cloud server can solve for all unknown $\mathbf{B}_i$ $(i = 4, \ldots, m)$!

Remarks. The above attack works due to the elimination of the randomness introduced by $\mathbf{A}_i$. When applying the property Eq. (5) in Eq. (12), the number of secret unknowns will be equal to the number of equations. In fact, we can also show that by knowing a set of $(\mathbf{B}_c, \mathbf{C}_\mathbf{F})$'s, any plaintext query can be recovered by eliminating the random matrix $\mathbf{E}_c$ with the property $\mathbf{E}_c\mathbf{R}^T = (1, 1, \ldots, 1)^T$. This implies that by knowing a sequence of users' queries and their encrypted versions, the cloud can reveal any other plaintext queries (submitted by other users) even if they have been encrypted. So, the biometric identification scheme in [20] is also insecure under the *Attack Scenario 3*. Due to the space limitation, we omit the analysis here.

B Attack on Yuan et al. [20] by Exploiting Euclidian Distance Results

We next describe another attack on the scheme to recover $\mathbf{B}_i$ by exploiting the Euclidian distance results.

In the scheme of Yuan et al. [20], the cloud server compares Euclidean distance between $\mathbf{b}_i$ and $\mathbf{b}_c$ by computing

$$\mathbf{P}_i = \mathbf{C_H}\mathbf{C}_i\mathbf{C_F}\mathbf{C_R} = \sum_{j=1}^{n+1} b_{ij}b_{cj} = \sum_{j=1}^{n} b_{ij}b_{cj} - \frac{1}{2}\sum_{j=1}^{n} b_{ij}^2. \tag{13}$$

We show that it is easy to construct enough equations to recover $\mathbf{B}_i$. Similarly, we give an illustrating example. Assume that $n = 2$ and $\mathbf{B}_i = [0, 2, -2]$ $(i \in [1, m])$. Yuan et al. [20] claimed that their scheme allows the collusion between the cloud server and a number of users. Based on this fact, the adversary can select query plaintexts of his interest for encryption. Therefore, assume that two *linear independent* FingerCodes $\mathbf{B}_{c1} = [7, 8, 1]$ and $\mathbf{B}_{c2} = [1, 2, 1]$ are chosen for query, and their corresponding encrypted queries $\mathbf{C}_{\mathbf{F}1}$ and $\mathbf{C}_{\mathbf{F}2}$ are also known by the cloud server. After performing the Euclidian distance comparison using Eq. (13), the cloud server has $\mathbf{P}_1 = 14$ and $\mathbf{P}_2 = 2$. By using two equations $\mathbf{P}_i = \sum_{j=1}^{2} b_{i'j}b_{cj} - \frac{1}{2}\sum_{j=1}^{2} b_{i'j}^2$ $(i = 1, 2)$, the cloud server can solve $\mathbf{B}_{i'} = [0, 2, -2] = \mathbf{B}_i$. The success of this attack further demonstrates that Yuan's scheme is vulnerable in the *Attack Scenario 3*. This attack works due to the lack of randomness in the Euclidian distance results $\mathbf{P}_i$. It tells us that besides the biometric data, $\mathbf{P}_i$ should also be well-protected while not affecting result correctness.

References

1. Barni, M., Bianchi, T., Catalano, D., Raimondo, M.D., Labati, R.D., Failla, P., Fiore, D., Lazzeretti, R., Piuri, V., Scotti, F. et al.: Privacy-preserving fingercode authentication. In: Proceedings of MMSec 2010, pp. 231–240. ACM (2010)
2. Blanton, M., Aliasgari, M.: Secure outsourced computation of iris matching. J. Comput. Secur. **20**(2), 259–305 (2012)
3. Blanton, M., Gasti, P.: Secure and efficient protocols for iris and fingerprint identification. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 190–209. Springer, Heidelberg (2011)
4. Chun, H., Elmehdwi, Y., Li, F., Bhattacharya, P., Jiang, W.: Outsourceable two-party privacy-preserving biometric authentication. In: Proceedings of ASIACCS 2014, pp. 401–412. ACM (2014)
5. Elmehdwi, Y., Samanthula, B.K., Jiang, W.: Secure k-nearest neighbor query over encrypted data in outsourced environments. In: Proceedings of ICDE 2014, pp. 664–675. IEEE (2014)
6. Erkin, Z., Franz, M., Guajardo, J., Katzenbeisser, S., Lagendijk, I., Toft, T.: Privacy-preserving face recognition. In: Goldberg, I., Atallah, M.J. (eds.) PETS 2009. LNCS, vol. 5672, pp. 235–253. Springer, Heidelberg (2009)
7. Galbraith, S.D.: Mathematics of Public Key Cryptography. Cambridge University Press, New York (2012)
8. Huang, Y., Malka, L., Evans, D., Katz, J.: Efficient privacy-preserving biometric identification. In: Proceedings of NDSS 2011 (2011)
9. Jain, A.K., Hong, L., Pankanti, S.: Biometric identification. Commun. ACM **43**(2), 90–98 (2000)
10. Jain, A.K., Prabhakar, S., Hong, L., Pankanti, S.: Filterbank-based fingerprint matching. IEEE Trans. Image Process. **9**(5), 846–859 (2000)
11. Katz, J., Lindell, Y.: Introduction to Modern Cryptography: Principles and Protocols. CRC Press, Boca Raton (2007)
12. Lindell, Y., Pinkas, B.: A proof of security of yaos protocol for two-party computation. J. Cryptology **22**(2), 161–188 (2009)
13. Liu, K., Giannella, C.M., Kargupta, H.: An attacker's view of distance preserving maps for privacy preserving data mining. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 297–308. Springer, Heidelberg (2006)
14. Mell, P., Grance, T.: Draft nist working definition of cloud computing. Referenced June 3rd **15**, 32 (2009)
15. Osadchy, M., Pinkas, B., Jarrous, A., Moskovich, B.: Scifi-a system for secure face identification. In: Proceedings of the S&P 2010, pp. 239–254. IEEE (2010)
16. Qi, Y., Atallah, M.J.: Efficient privacy-preserving k-nearest neighbor search. In: Proceedings of ICDCS 2008, pp. 311–319. IEEE (2008)
17. Sadeghi, A.-R., Schneider, T., Wehrenberg, I.: Efficient privacy-preserving face recognition. In: Lee, D., Hong, S. (eds.) ICISC 2009. LNCS, vol. 5984, pp. 229–244. Springer, Heidelberg (2010)
18. Wang, Q., Hu, S.S., Ren, K., He, M.Q., Du, M.X., Wang, Z.B.: CloudBI: Practical privacy-preserving outsourcing of biometric identification in the cloud. Technical report (2015). http://web.eecs.utk.edu/~zwang32/publications/CloudBI.pdf
19. Wong, W.K., Cheung, D.W-L., Kao, B., Mamoulis, N.: Secure knn computation on encrypted databases. In: Proceedings of SIGMOD 2009, pp. 139–152. ACM (2009)
20. Yuan, J., Yu, S.: Efficient privacy-preserving biometric identification in cloud computing. In: Proceedings of INFOCOM 2013, pp. 2652–2660. IEEE (2013)