Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 525387, 13 pages
http://dx.doi.org/10.1155/2014/525387

Research Article

Minutiae Matching with Privacy Protection Based on the Combination of Garbled Circuit and Homomorphic Encryption

Mengxing Li,[1] Quan Feng,[2] Jian Zhao,[2] Mei Yang,[2] Lijun Kang,[3] and Lili Wu[3]

[1] *School of Communication and Electronic Engineering, Hunan City University, Yiyang, Hunan 41300, China*
[2] *Engineering College, Gansu Agricultural University, Anning District, Lanzhou 730070, China*
[3] *College of Information Sciences and Technology, Gansu Agricultural University, Anning District, Lanzhou 730070, China*

Correspondence should be addressed to Quan Feng; fquan@sina.com

Received 10 August 2013; Accepted 17 November 2013; Published 24 February 2014

Academic Editors: J. Ma and J. C. R. Tseng

Biometrics plays an important role in authentication applications since they are strongly linked to holders. With an increasing growth of e-commerce and e-government, one can expect that biometric-based authentication systems are possibly deployed over the open networks in the near future. However, due to its openness, the Internet poses a great challenge to the security and privacy of biometric authentication. Biometric data cannot be revoked, so it is of paramount importance that biometric data should be handled in a secure way. In this paper we present a scheme achieving privacy-preserving fingerprint authentication between two parties, in which fingerprint minutiae matching algorithm is completed in the encrypted domain. To improve the efficiency, we exploit homomorphic encryption as well as garbled circuits to design the protocol. Our goal is to provide protection for the security of template in storage and data privacy of two parties in transaction. The experimental results show that the proposed authentication protocol runs efficiently. Therefore, the protocol can run over open networks and help to alleviate the concerns on security and privacy of biometric applications over the open networks.

1. Introduction

Biometric characteristic, such as fingerprint, face, and iris, has been used to a higher level of security in order to cope with an increasing demand for reliable and highly usable information security systems. Currently, most practical biometric systems handle biometric data locally, or in secure local network. When they are migrated directly to an open network which is in partly secure or insecure environments, the inherent risks of privacy and security of traditional biometric technologies will be blown up. Jain et al. [1] analyze the vulnerabilities about biometrics to intrinsic failures and potential attacks by adversaries. One of the most serious risks is compromising the template database, which will exert a disastrous impact on the whole authentication system. Besides, a new challenge of privacy and security arises since a remote server and a user may not trust each other before the authentication. Their respective data should not be directly delivered to each other as in a usual biometric system. These data, as the private data, must not be leaked to each other during the authentication. Therefore, the template and the privacy protection are the key challenges for any biometric system deployed over the open network and must be properly addressed.

Among many solutions to privacy-preserving biometrics, some schemes focus on constructing a transformed template with noninvertible transformation, including fuzzy commitment [2], fuzzy vaults [3], biohashing [4], fuzzy sketch/fuzzy extractor [5], cancelable template [6], and random local region descriptor (RLRD) [7]. In these schemes, the researchers try to prevent biometric information being revealed from the transformed template by one-way transformation. However, some of them, for example, fuzzy vault, biohashing, and cancelable template, are proven to be vulnerable to attacks [8–10]. Though fuzzy sketch scheme is

theoretically complete and secure, it complies with stronger requirements than what suffices in practice, which leads to degradation of accuracy.

Some researchers employ cryptographic technique to achieve privacy protection, which has a solid theoretical foundation. They have designed privacy-preserving protocols by resorting to secure two-party computation, where two nontrusted parties cooperate to carry out a computation without revealing their own inputs. Bringer and Chabanne [11] proposed a biometric authentication protocol which can protect the sensitive relationship between the biometric template and the relevant pseudorandom username, which is based on the homomorphic properties of Goldwasser-Micali and Paillier cryptosystems. Tang et al. [12] developed two concepts of biometric privacy called identity privacy and transaction anonymity, respectively, and presented an authentication protocol employing Private Information Retrieval protocols which was based on the ElGamal cryptosystems. In both works, however, the type of biometrics is limited to those which can be represented as binary strings because their protocols only computed the Hamming distance between biometric templates in the encryption domain. Erkin et al. [13] designed a privacy-preserving face recognition protocol based on additive homomorphic encryption, in which Euclidean distances between feature vectors or finding a minimum are computed by using the homomorphic property of the cipher texts. The protocol of Erkin requires large network traffic and a large memory, which makes those systems less practical. Sadeghi et al. [14] improved the performance by combining homomorphic encryption with garbled circuits. Barni et al. [15] employed fingercode [16] as a feature along with the technique similar to Sadeghi et al. [14] to realize privacy-preserving system for fingerprint-based identification. Huang et al. [17] provided further improvement for the privacy-preserving identification protocol by elaborate optimizations such as packing method, carefully integrating the subtraction and comparison computations, and backtracking technique. Upmanyu et al. [18] designed "blind" authentication protocol in which some classifiers such as linear, support vector machines, and neural networks were realized by using multiplicative homomorphic encryption. However, the experience of biometric recognition proved that, in most cases, direct matching shows a better performance than classifiers because, in practical applications, it is difficult to gather enough biometric samples of one person to train the classifier.

In this paper, we propose a solution to fingerprint authentication with providing protection for both template and privacy. In particular, we focus on the realization of fingerprint authentication in a secure framework by implementing minutiae matching instead of fixed-length features as in the existed schemes [16–18]. We address the problem of minutiae matching in the encrypted domain by combining homomorphic encryption with gabled circuit. In the proposed scheme, the template of a user, stored on the server, is encrypted by the user's private key, even the server learning nothing about the genuine information of the user. Therefore, our scheme shows a better performance in privacy protection than the existed schemes [16, 17].

2. Preliminaries

The primary cryptographic tools we use are homomorphic encryption, oblivious transfer, and garbled circuits. We briefly summarize each of these standard techniques here.

2.1. Additively Homomorphic Encryption. Let $[x]$ denote encryption of x with a public key. Our constructions use a semantically secure public-key encryption scheme that preserves the group homomorphism of addition and allows multiplication by a constant. This property, which is obtained by the additively homomorphic encryption schemes, supports the following operations that can be performed without knowledge of the private key: (1) Given the encryptions $[a]$ and $[b]$, we can efficiently compute $[a+b] = [a][b]$. (2) Given that a constant c belongs to the same group, we can compute $[c \cdot a] = [a]^c$.

There are many public-key cryptosystems satisfying the above properties. In our implementation, we use Paillier's cryptosystem [19] which has plaintext space Z_N and cipher text space $Z^*_{N^2}$, where N is a T-bit RSA modulus and T is the bits length of RSA.

2.2. Oblivious Transfer. 1-out-of-2 oblivious transfer (OT_1^2) allows a sender, holding strings s^0, s^1, to transfer to a receiver, holding a selection bit b, exactly one of the inputs s^b. The receiver learns nothing about s^{1-b}, and the sender has no idea of b. Parallel OT_1^2 of m l-bit strings is denoted as OT_l^m. For $i = 1, \ldots, m$, the sender inputs a pair of l-bit strings $s_i^0, s_i^1 \in \{0,1\}^l$ and the receiver inputs m choice bits $b_i \in \{0,1\}$. At the end of the protocol, the receiver learns about the chosen string $s_i^{b_i}$, but nothing about $s_i^{1-b_i}$ whereas the sender learns nothing about the choice b_i. Oblivious transfer has been studied extensively. In this scheme, we use oblivious transfer extension scheme of Ishai et al. [20] which serves to efficiently reduce the number of computationally expensive public-key operations of OT_l^m to be independent of m.

2.3. Garbled Circuits. Garbled circuit [21, 22] allows two parties holding inputs x and y, respectively, to evaluate an arbitrary function $f(x, y)$ without leaking any information about their inputs. The basic idea is that a server creates an "encrypted" version of the circuit C to compute f, and then a client obliviously computes the output of the circuit. In more detail, for each wire w_i of C, the server randomly chooses two secrets, $\widetilde{w}_i^0$ and $\widetilde{w}_i^1$, where $\widetilde{w}_i^j$ is called garbled value of w_i's value j. Further, for each gate G_i of C, the server creates a garbled table $\widetilde{T}_i$ which records a collection of the garbled values corresponding to the output wires of G_i with those corresponding to the input wires. All $\widetilde{T}_i$s are transferred to a client as well as the garbled values of the server' input. The client gets the garbled values corresponding to his input by OT protocol from the server. Then, the client can evaluate the garbled circuit to obtain the garbled output simply gate by gate, using the garbled tables $\widetilde{T}_i$s.

Some optimizations can be applied to the standard garbled circuit protocol. A powerful technique is "free XOR"

scheme [23, 24] which eliminates the need to garble XOR gates, so XOR gates become "free," incurring no communication or cryptographic operations. Another efficient approach [25] can reduce the size of a garbled table from four to three cipher texts for a 2-input-and-1-output gate, thus saving 25% of network bandwidth.

3. Overview

In the existing schemes [13–15, 17], the authors only considered protecting the privacy of biometric data of a user and a server, without taking the template protection into consideration. Although biometrics is assumed as public data, it should not be easy to obtain the biometric data by compromising a central server. Fingerprint is one of the biometric characteristics with the highest level of reliability. Barni et al. [15] and Huang et al. [17] both take fingercode as the feature, whose length is fixed. This helps to reduce the computational and communicational complexity. In fact, the minutiae set is the most popular feature used in practical systems because minutiae-based matching is more robust to distortions frequently encountered in practical applications, so these systems usually achieve a good accuracy. Minutiae are the endpoints and bifurcations of fingerprint ridges. Each minutia can be represented as (x, y, θ) triplet, where (x, y) is the location of the minutia and θ angle of the associated ridge $(0 \le \theta < 360^\circ)$. A template of minutiae is represented as a set of points in the three-dimensional. Fingerprint matching can be reduced to finding the paired points problem. Let $M^T = \{(x_i, y_i, \theta_i) \mid 1 \le i \le N_T\}$ and $M^Q = \{(x'_j, y'_j, \theta'_j) 1 \le i \le N_Q\}$ denote the template and query, respectively. A minutiae-based fingerprint matching algorithm usually returns the number of matched minutiae on both M_T and M_Q to generate similarity scores. In this paper, the matching score S_M is calculated as follows:

$$S_M = \frac{100 \times N_M}{\max\left(N_T, N_Q\right)}, \tag{1}$$

where N_M is the number of paired minutiae. If S_M is greater than or equal to a predefined threshold T_M; then the query and the template can be considered coming from the same finger.

A minutia $M_i = (x_i, y_i, \theta_i)$ in M^T and a minutia $M'_j = (x'_j, y'_j, \theta'_j)$ in M^Q are considered matching if the following conditions are satisfied:

$$d\left(\left(x_i, y_i\right), \left(x'_j, y'_j\right)\right) < T_D, \tag{2}$$

$$\min\left(\left|\theta_i - \theta'_j\right|, 360 - \left|\theta_i - \theta'_j\right|\right) < T_\theta, \tag{3}$$

where $d(\,)$ is a distance function and T_D and T_θ are the given thresholds. In this paper, we consider two distance metrics: (i) square of Euclidean distance $d_E((x_i, y_i), (x'_j, y'_j)) = (x_i - x'_j)^2 + (y_i - y'_j)^2$ (for simplicity, we still call Euclidean distance) and (ii) city block distance $d_B((x_i, y_i), (x'_j, y'_j)) = |x_i - x'_j| + |y_i - y'_j|$. However, for a given minutia belonging to M^Q, the above approach might find at least one matching result belonging to M^T, but, in fact, at most one is correct. In this paper, for a minutia $M'_j \in M^Q$ and those minutiae belonging to M^T satisfying (2) and (3), we choose the closest to M'_j as the matched minutia.

We design the privacy-preserving protocol based on the aforementioned matching algorithm, which works in the two-party setting in the semihonest attacker model. In this model, the participants do not deviate from their protocol but may use any information they obtain to their own advantage. Suppose that Bob (the server) holds a database containing the template of the users. To protect the user's privacy, however, the template is not the original feature but its encrypted version. Thus, even Bob does not learn the user's biometric information. Alice (the user) can update her template by choosing a new private key. When Bob receives Alice's request for authentication, he first retrieves the encrypted template from his database. During the interactive authentication protocol, Alice provides the fresh minutiae M^Q as the inputs. She trusts Bob to correctly perform the matching algorithm but is unwilling to expose her information of fingerprint to Bob. The protocol consists of two phases: the first one is related to the blind distance computation, which is carried out by homomorphic encryption (Section 4); the second one is related to minutiae matching, which is implemented by garbled circuit (Section 5). At the end of protocol, Alice obtains the number of the matched minutiae and returns it to Bob. However, the number is represented as the garbled value, so Alice knows nothing about the genuine value. Bob decrypts it and computes the matching score. For the sake of simplicity, we just describe the content concerning distance computation and minutiae matching.

4. Blind Distance Computation

In this section, we present two protocols which compute the two kinds of blind distances of Euclidean and city block, respectively, which is the first phase in our authentication process.

4.1. Euclidean-Distance Protocol

Basic. Alice computes the distance between each minutia in M^Q and that in M^T with the help of Bob. As mentioned in (2) and (3), there are two distances needing to be calculated: the spatial distance and the directional difference. The spatial distance discussed in this section is Euclidean distance. M^T is the encrypted version with Alice's private key while M^Q is in the clear. Here we denote the encrypted template as $EM^T = \{EM_i = ([x_i], [x_i^2], [y_i], [y_i^2], [\theta_i]) \mid 1 \le i \le N_T\}$. EM^T is held by Bob. Since Alice holds the private key, to keep his privacy, Bob can blind M^T in advance and Alice can only compute the blind distance. To do so, he blinds EM with the uniformly random numbers r_1, r_2, r_3, r_4 from the plaintext space to get the following cipher texts by using the homomorphic property: $[a] = [x][r_1] = [x + r_1]$, $[b] = [y][r_2] = [y + r_2]$, $[c] = [x^2][r_3] = [x^2 + r_3]$, $[d] = [y^2][r_4] = [y^2 + r_4]$. Then he

sends these cipher texts to Alice. Alice decrypts them to get a, b, c,and d. She calculates

$$\begin{aligned} ed &= c + d - 2x'a - 2y'b + x'^2 + y'^2 \\ &= x^2 - 2xx' + x'^2 + y^2 - 2yy' + y'^2 \\ &\quad + r_3 + r_4 - 2r_1x' - 2r_2y'. \end{aligned} \tag{4}$$

To further compute the blind distance, Alice needs $2r_1x' + 2r_2y'$ which can be easily gotten by interacting with Bob: Alice transfers $[2x']$ and $[2y']$ to Bob. The latter computes and returns $[t] = [2x']^{r_1}[2y']^{r_2}[r_5]$, where $r_5 \in_R Z_N$. It is easy to observe that $[t] = [2x'r_1 + 2y'r_2 + r_5]$. Alice decrypts $[t]$ and calculates the blind distance $bd = ed + t = d_E + r_3 + r_4 + r_5 = d_E + r$.

Bob further blinds $[\theta]$ with a random number by computing $e = [r_6][\theta] = [r_6 + \theta]$, where r_6 is a random number and then transfers it to Alice.

Improvement. In order to improve the efficiency of the above approach, we choose the shorter random masks and pack multiple values into a single cipher text. Assume that $x(x')$, $y(y')$, x^2, y^2, and θ are ρ-, ρ-, 2ρ-, 2ρ-, and μ-bit positive integers, respectively, and the random masks r_1, r_2, r_3, r_4, r_5, and r_6 are η_{r_1}-, η_{r_2}-, η_{r_3}-, η_{r_4}-, η_{r_5}-, and η_{r_6}-bit positive integers, respectively. The resulting blind values $x + r_1$, $y + r_2$, $x^2 + r_3$, $y^2 + r_4$, and $\theta + r_6$ can be packed into a single cipher text. The cross item such as $t = 2x'r_1 + 2y'r_2 + r_5$ can also be packed. The storage size of the encrypted minutiae template can also be reduced by using the packing technique. That is, when generating the encrypted template EM^T, we firstly pad some zeros before x, y, x^2, y^2, and θ to increase their lengths to $\eta_{r_1} + 1, \eta_{r_2} + 1, \eta_{r_3} + 1, \eta_{r_4} + 1, \eta_{r_6} + 1$, respectively, and concatenate them together. As described later, we have $\eta_{r_1} = \rho + \delta, \eta_{r_2} = \rho + \delta, \eta_{r_3} = 2\rho + \delta, \eta_{r_4} = 2\rho + \delta, \eta_{r_6} = \eta + \delta$ (δ is a security parameter which will be explained later). Then a unit chunk representing a minutia $M_i = (x_i, y_i, \theta_i)$ is written down as: $mp_i = 0^{\delta+1}\|x_i\|\ 0^{\delta+1}\|y_i\|0^{\delta+1}\|x_i^2\|0^{\delta+1}\|y_i^2\|0^{\delta+1}\|\theta_i$. The purpose of padding $\delta + 1$ zeroes before each component is to prevent the possible overflow when the component is added to the corresponding mask. Therefore, one cipher text can contain $N_P = \lceil T/(6\rho + \mu + 5\delta + 5)\rceil$ minutiae. The number of cipher texts of EM^T is only $N_M = \lfloor N_T/N_P \rfloor$. Compared with the basic approach, the method saves $(1 - N_M/5N_T)$% storage space. We rewrite EM^T as: $EM^T = \{cp_1 = [mp_1\| \cdots \| mp_{N_P}], \ldots, cp_{N_M} = [mp_{(N_M-1)N_P+1}\| \cdots \| mp_{N_T}]\}$. These methods reduce the communicational and computational complexity because each cipher text carries multiple blind minutiae.

Enrollment. When Alice registers herself to Bob, she creates an encrypted template EM^T by employing the aforementioned method. She sends EM^T to Bob who stores EM^T in a safe database.

Protocol. In the authentication phase, Alice and Bob carry out the protocol Euclidean-Distance to compute the blind distances of the minutiae (Algorithm 1). The detail is given as follows. For simplicity, we assume that two parties have learned the number of minutiae in the template and the query, that is, N_T and N_Q, and exchanged the public key.

Packing Size. We have supposed that x, y, x^2, y^2, and θ are ρ, ρ, 2ρ, 2ρ, and μ bits, respectively, and random masks r_1, r_2, r_3, r_4, r_6 are η_{r_1}, η_{r_2}, η_{r_3}, η_{r_4}, and η_{r_6} bits, respectively. To keep statistical security, we need $\eta_{r_1} > \rho$, $\eta_{r_2} > \rho$, $\eta_{r_3} > 2\rho$, $\eta_{r_4} > 2\rho$, and $\eta_{r_6} > \mu$. In fact, our random masks are longer than the corresponding blind values by δ bits; that is, $\eta_{r_1} = \rho + \delta$, $\eta_{r_2} = \rho + \delta$, $\eta_{r_3} = 2\rho + \delta$, $\eta_{r_4} = 2\rho + \delta$, and $\eta_{r_6} = \mu + \delta$. Besides, we need to handle the possible overflow of the intermediate values in computation. Therefore, the length of these values will be determined as follows: a, b, c, d, and e (step 1) are $\rho + \delta + 1$-, $\rho + \delta + 1$-, $2\rho + \delta + 1$-, $2\rho + \delta + 1$-, and $\mu + \delta + 1$-bit values. The main task of the protocol is computing the cross item $t_{i,j}^4 = 2r_i^1x_j' + 2r_i^2y_j' + r_{i,j}^5$ (step 4 (i)). We need $(\eta_{r_1} + \rho + 1) + 1 = 2\rho + \delta + 2$ bits to represent $2r_i^1x_j' + 2r_i^2y_j'$, and $\eta_{r_5} = 2\rho + \delta + 2$ to represent the random mask $r_{i,j}^5$. Accordingly, it is clear that $\lambda = \eta_{r_5} + 1 = 2\rho + \delta + 3$ bits are sufficient (step 2 (ii)). We pad $\lambda - \rho - 1$ zeros before $2x_j'$ and $2y_j'$ to form λ-bit u_i^1 and u_i^2 because this ensures that no overflow happens when computing $2r_i^1x_j' + 2r_i^2y_j'$ (step 2 (ii)). The blind squared-distance $bd(i, j)$ and the blind orientation difference $bo(i, j)$ are λ-bit and $\mu + \delta + 1$-bit, respectively.

Correctness. Here, we prove that the equality in Euclidean-Distance, $bd(i, j) = ed_{ij} + t_{i,j}^4 = d_E(i, j) + r_{ij}$ is established. In step 2 of the protocol, since u_i^1 is $\lambda = 2\rho + \delta + 3$-bit, r_i^1, $\rho + \delta$-bit, and $2x'$, $\rho + 1$-bit binary string, respectively; in step 3,

$$\begin{aligned} [B_1]^{r_i^1} &= \left[u_1^1 \| \cdots \| u_{N_X}^1\right]^{r_i^1} = \left[\sum_{j=1}^{N_X} u_j^1 \cdot 2^{(N_X-j)\lambda}\right]^{r_i^1} \\ &= \left[\sum_{j=1}^{N_X} u_j^1 \cdot r_i^1 \cdot 2^{(N_X-j)\lambda}\right] = \left[\sum_{j=1}^{N_X} 2x_j' \cdot r_i^1 \cdot 2^{(N_X-j)\lambda}\right] \\ &= \left[2x_1'r_i^1 \left\|2x_2'r_i^1\right\| \cdots 2x_{N_X}'r_i^1\right], \\ [B_2]^{r_i^1} &= \left[2x_{N_X+1}'r_i^1 \| \cdots\right], \end{aligned} \tag{5}$$

and so forth.

Similarly,

$$\begin{aligned} [C_1]^{r_i^1} &= \left[u_1^2 \| \cdots \| u_{N_X}^2\right]^{r_i^1} = \left[2y_1'r_i^2 \left\|2y_2'r_i^2\right\| \cdots 2y_{N_X}'r_i^2\right], \\ [C_2]^{r_i^1} &= \left[2y_{N_X+1}'r_i^2 \| \cdots\right], \end{aligned} \tag{6}$$

and so forth. So,

$$\begin{aligned} \left[t_{i,1}^1\right] &= [B_1]^{r_i^1}[C_1]^{r_i^2} = \left[r_i^1B_1 + r_i^2C_1\right] \\ &= \left[2x_1'r_i^1 + 2y_1'r_i^2 \| \cdots \| 2x_{N_X}'r_i^1 + 2x_{N_X}'r_i^2\right], \\ \left[t_{i,2}^1\right] &= \left[2x_{N_X+1}'r_i^1 + 2y_{N_X+1}'r_i^2 \| \cdots\right], \end{aligned} \tag{7}$$

and so forth.

Input: the encrypted template $EM^T = \{cp_1 = [mp_1 \| \cdots \| mp_{N_P}], \ldots, cp_{N_M} = [mp_{(N_M-1)N_P+1} \| \cdots \| mp_{N_T}]\}$ from Bob, and query minutiae from Alice M^Q

Output: Alice gets the blind squared-distance bd (i, j) and blind direction bo (i, j) between $M_i = (x_i, y_i, \theta_i)$ and $M'_j = (x'_j, y'_j, \theta'_j)$.

(1) Bob chooses r_1^i, $r_2^i \in_R \{0,1\}^{\rho+\delta}$, r_3^i, $r_4^i \in_R \{0,1\}^{2\rho+\delta}$, $r_6^i \in_R \{0,1\}^{\mu+\delta}$, $1 \le i \le N_T$, concatenates them as strings:
$R_1 = rt_1 \| \cdots \| rt_{N_P}, \ldots, R_{N_M} = rt_{(N_M-1)N_P+1} \| \cdots \| rt_{N_T}$,
where $rt_i = 0 \| r_1^i \| 0 \| r_2^i \| 0 \| r_3^i \| 0 \| r_4^i \| 0 \| r_6^i \|$.
For $k = 1, \ldots, N_M$, he computes $[A_k] = cp_k [R_k]$,
where $[A_1] = [a_1^1 \| b_2^1 \| c_3^1 \| d_4^1 \| e_5^1 \| \cdots \| a_1^{N_P} \| b_2^{N_P} \| c_3^{N_P} \| d_4^{N_P} \| e_5^{N_P}]$, and so forth
and $a_i = r_1^i + x_i, b_i = r_2^i + y_i, c_i = r_3^i + x_i^2, d_i = r_4^i + y_i^2, e_i = r_6^i + \theta_i$. He sends these cipher texts to Alice.

(2) Alice decrypts $[A_1], \ldots, [A_{N_M}]$ and obtains $a_i = r_1^i + x_i, b_i = r_2^i + x_i, c_i = r_3^i + x_i^2, d_i = r_4^i + y_i^2$,
and $e_i = r_6^i + \theta_i (1 \le i \le N_T)$ by parsing $A_1, \ldots, A_{N_M}$ into $\rho + \delta + 1$-, $\rho + \delta + 1$-, $2\rho + \delta + 1$-,
$2\rho + \delta + 1$- and $\mu + \delta + 1$-bit chunks respectively. And:

(i) For $i = 1, \ldots, N_T$ and $j = 1, \ldots, N_Q$, she calculates $ed_{ij} = c_i + d_i - 2x'_j a_i - 2y'_j b_i + x'^2_j + y'^2_j$.

(ii) She pads $\lambda - \rho - 1$ zeros before each $2x'_j$ and $2y'_j$,
constructs the λ-bit strings: $u_j^1 = 0^{(\lambda-\rho-1)} \| 2x'_j$ and $u_j^2 = 0^{(\lambda-\rho-1)} \| 2y'_j$, where $\lambda = 2\rho + \delta + 3$.
She then concatenates these strings together respectively: $B_1 = u_1^1 \| \cdots \| u_{N_X}^1, \ldots,$
$B_{N_B} = u_{(N_B-1)N_X+1}^1 \| \cdots \| u_{N_Q}^1, C_1 = u_1^2 \| \cdots \| u_{N_X}^2, \ldots, C_{N_B} = u_{(N_B-1)N_X+1}^2 \| \cdots \| u_{N_Q}^2$,
where $2x'_j$ and $2y'_j$ are both $\rho + 1$-bit values, u_i^1 and u_i^2 are both λ-bit values, $N_X = \lceil T/\lambda \rceil$
and $N_B = \lfloor N_Q/N_X \rfloor$.

(iii) For $k = 1, \ldots, N_B$, she computes $[B_k]$ and $[C_k]$, transfers them to Bob.

(3) Bob receives the cipher texts. Then:

(i) For $i = 1, \ldots, N_T$ and $k = 1, \ldots, N_B$, he computes $[t_{i,k}^1] = [B_k]^{r_i^1} [C_k]^{r_i^2} = [r_i^1 B_k + r_i^2 C_k]$.

(ii) He chooses $r_{i,j}^5 \in_R \{0,1\}^{\lambda-1}$, $1 \le i \le N_Q$, $1 \le j \le N_Q$, pads a zero before each number,
concatenates each N_X numbers as strings with respect to j:
$t_{i,1}^2 = 0 \| r_{i,1}^5 \| \cdots \| 0 \| r_{i,N_X}^5, \ldots, t_{i,N_B}^2 = 0 \| r_{i,(N_B-1)N_X+1}^5 \| \cdots \| 0 \| r_{i,N_Q}^5$, and computes
$[t_{i,1}^2], \ldots, [t_{i,N_B}^2]$.

(iii) For $i = 1, \ldots, N_T$ and $k = 1, \ldots, N_B$, he computes $[t_{i,k}^3] = [t_{i,k}^1][t_{i,k}^2] = [t_{i,k}^1 + t_{i,k}^2]$
and sends them to Alice.

(4) Alice receives and decrypts $[t_{i,k}^3]$, $1 \le i \le N_T$, $1 \le k \le N_B$. Then:

(i) For $i = 1, \ldots, N_T$, she unpacks each $t_{i,k}^3$ by parsing it into λ-bit chunks to obtain
$t_{i,1}^4, \ldots, t_{i,N_X}^4, \ldots, t_{i,(N_B-1)N_X+1}^4, \ldots, t_{i,N_Q}^4$, where $t_{i,j}^4 = 2r_i^1 x'_j + 2r_i^2 y'_j + r_{i,j}^5$.

(ii) For $i = 1, \ldots, N_T$ and $j = 1, \ldots, N_Q$, she computes the blind squared-distance
$\mathrm{bd}(i, j) = ed_{ij} + t_{i,j}^4$ and the blind orientation difference $\mathrm{bo}(i, j) = e_i - \theta'_j$
between $M'_j \in M^Q$ and each $M_i \in M^T$.

Algorithm 1: Euclidean-distance (EM^T, M^Q).

One can further verify that

$$\begin{aligned}
[t_{i,1}^3] &= [t_{i,1}^1 + t_{i,1}^2] \\
&= [2r_i^1 x'_1 + 2r_i^2 y'_1 + r_{i,1}^5 \| \cdots \| 2r_i^1 x'_{N_X} + 2r_i^2 y'_{N_X} \\
&\quad + r_{i,N_X}^5], \ldots, \\
[t_{i,N_B}^3] &= [t_{i,N_B}^1 + t_{i,N_B}^2] \\
&= [2r_i^1 x'_{(N_B-1)N_X+1} + 2r_i^2 y'_{(N_B-1)N_X+1} \\
&\quad + r_{i,(N_B-1)N_X+1}^5 \| \cdots \| 2r_i^1 x'_{N_Q} + 2r_i^2 y'_{N_Q} + r_{i,N_Q}^5].
\end{aligned} \tag{8}$$

Thus,

$$\begin{aligned}
t_{i,j}^4 &= 2r_i^1 x'_j + 2r_i^2 y'_j + r_{i,j}^5 \ (\text{step } 4), \\
bd(i, j) &= ed_{ij} + t_{i,j}^4 = (x_i - x'_j)^2 + (y_i - y'_j)^2 + r_i^3 + r_i^4 + r_{i,j}^5 \\
&= d_E(i, j) + r_i^3 + r_i^4 + r_{i,j}^5 = d_E(i, j) + r_{ij}.
\end{aligned} \tag{9}$$

Obviously, $bo(i, j) = e_i - \theta'_j = \theta_i - \theta'_j + r_i^6$, which is the blind difference between θ_i and θ'_j.

Security. The security of the protocol depends on whether the short masks can adequately blind the values that Bob is unwilling to reveal to Alice. Since the addition in the homomorphic encryption is computed over the integers rather than modulo addition, we only obtain statistical hiding

Input: the encrypted template
$EM^T = \left\{cp_1 = \left[mp_1 \parallel \cdots \parallel mp_{N_P}\right], \ldots, cp_{N_M} = \left[mp_{(N_M-1)N_P+1} \parallel \cdots \parallel mp_{N_T}\right]\right\}$
from Bob, and query minutiae from Alice M^Q
Output: Alice gets a blind list $\{bx(i, j), by(i, j), \text{bo}(i, j) \mid 1 \le i \le N_T, 1 \le j \le N_Q\}$
between $M_i = (x_i, y_i, \theta_i)$ and $M'_j = (x'_j, y'_j, \theta'_j)$.
(1) Bob chooses $r_1^i, r_2^i \in_R \{0,1\}^{\rho+\delta}, r_3^i \in_R \{0,1\}^{\mu+\delta}$, $1 \le i \le N_T$, concatenates them as strings:
$R_1 = rt_1 \parallel \cdots \parallel rt_{N_P}, \cdots, R_{N_M} = rt_{(N_M-1)N_P+1} \parallel \cdots \parallel rt_{N_T}$,
where $rt_i = 0 \parallel r_1^i \parallel 0 \parallel r_2^i \parallel 0 \parallel r_3^i$. For $k = 1, \ldots, N_M$,
he computes $[A_k] = cp_k[R_k]$, where $[A_1] = \left[a_1^1 \parallel b_2^1 \parallel e_5^1 \parallel \cdots \parallel a_1^{N_P} \parallel b_2^{N_P} \parallel e_5^{N_P}\right]$,
and so forth and $a_i = r_1^i + x_i$, $b_i = r_2^i + y_i$, $e_i = r_6^i + \theta_i$. He sends these cipher texts to Alice.
(2) Alice decrypts $[A_1], \ldots, [A_{N_M}]$ and obtains $a_i = r_1^i + x_i, b_i = r_2^i + x_i$ and $e_i = r_6^i + \theta_i (1 \le i \le N_T)$
by parsing $A_1, \cdots, A_{N_M}$ into $\rho + \delta + 1$-, $\rho + \delta + 1$- and $\mu + \delta + 1$-bit chunks respectively.
For $i = 1, \ldots, N_T$ and $j = 1, \ldots, N_Q$, she computes $bx(i, j) = a_i - x'_j = r_i^1 + x_i - x'_j$,
$by(i, j) = b_i - y'_j = r_i^2 + y_i - y'_j$
and $\text{bo}(i, j) = e_i - \theta'_j = r_i^6 + \theta_i - \theta'_j$.

Algorithm 2: City-block-distance (EM^T, M^Q).

rather than perfect hiding. If w is ρ-bit integer and r is uniform η-bit integer, then $v = w + r$ gives statistical security roughly $2^{\rho-\eta}$ for w, where w can stand for x, y, x^2, y^2, and θ. This probability can be lowered arbitrarily by choosing η properly. However, a longer η will increase computational and communicational complexity. Since x, y, and θ are ρ-, ρ-, and μ-bit integers, respectively, and the probability of guessing right x, y, and θ is $2^{-\rho}$, $2^{-\rho}$, and $2^{-\mu}$, respectively, it is sufficient that $\eta_{r_1} - \rho, \eta_{r_2} - \rho$ are not less than ρ and $\eta_{r_6} - \mu$ is not less than μ. Assume that ρ is equal to μ; thus, $\delta = \rho = \mu$ is satisfied.

Complexity. Since the computational complexity of Euclidean-Distance is dominated by operations related to Paillier encryption, such as exponentiation with an exponent of length T (Exp), encryption (Enc), and decryption (Dec), we only take their costs into consideration. The overall complexity of Euclidean-Distance is given in Table 1. The computational complexity of the protocol can be further reduced. The operations in step 1 of the protocol can be precomputed, and, thus, Bob saves N_M Encryption.

4.2. City-Block-Distance Protocol. In this section, we present a protocol based on city block distance (Algorithm 2). As there are no quadric components in the computation of city block distance, the size of the encrypted template can also be reduced. Let us describe how to generate the encrypted template firstly. For a minutia $M_i = (x_i, y_i, \theta_i)$, we pad $\delta + 1$ zeroes before each component and concatenate them to mp_i, where $mp_i = 0^{\delta+1} \| x_i \| 0^{\delta+1} \| y_i \| 0^{\delta+1} \| \theta_i$. The encrypted template can be created as: $EM^T = \{cp_1 = [mp_1 \| \cdots \| mp_{N_P}], \ldots, cp_{N_M} = [mp_{(N_M-1)N_P+1} \| \cdots \| mp_{N_T}]\}$, where $N_P = \lfloor T/(2\rho + \mu + 3\delta + 3) \rfloor$ and $N_M = \lceil N_T/N_P \rceil$. Compared with the template in Section 4.1, the size of this template is further reduced.

In the authentication phase, Alice and Bob firstly carry out the protocol City-Block-Distance to calculate a blind list for further computation of city block by using garbled circuits. We detail the protocol as follows.

Compared with Euclidean-Distance, this protocol is rather simple. It is easy to verify the correctness of the protocol by employing the homomorphic property. The secure analysis is similar to that of Euclidean-Distance. Table 2 shows its complexity. The computational complexity can be further reduced by precomputing step 1 of the protocol.

5. Circuits for Minutiae Matching

Garbled circuit is employed to complete minutiae matching with respect to Section 3. Section 5.1 describes the circuits related to Euclidean distance and Section 5.2 presents the circuits related to city block distance.

5.1. Circuits for Euclidean Distance. After the protocol Euclidean-Distance is carried out, Alice learns the blind Euclidean distance $bd(i, j) = d_E(i, j) + r_{ij}$ and the blind directional difference $bo(i, j) = \theta_i - \theta'_j + r_i^6$. The remaining tasks of authentication are implemented by using garbled circuits. The circuits firstly take off the random masks covered on $bd(i, j)$ and $bo(i, j)$ to get the Euclidean distances and the directional differences. Then, for each minutia (x_i, y_i, θ_i) belonging to $M^T (1 \le i \le N_T)$, the closest matching minutia which belongs to M^Q is found according to (2) and (3). To do so, the circuits choose the minutiae belonging to M^Q as the candidates, whose directional differences are smaller than the threshold T_θ (see (3)). The candidates' Euclidean distances are then fed into the minimum circuit to find the minimum. And then the minimum is checked whether it is smaller than the threshold T_D (see (2)). Finally, the number of minutiae belonging to M^Q meeting the above two conditions is counted. We adopt the efficient building blocks from [23, 24] to design our circuit: addition ADD, subtraction SUB, comparison CMP, and multiplexer MUX circuits.

Figure 1 shows the circuit ORIDIFF to take off the mask of $bo(i, j)$ and compute the result according to (3). Alice's

Table 1: Complexity of the protocol Euclidean-distance.

Round complexity	Communication complexity (bits)	Asymptotic computation complexity
3	Bob $\rightarrow$ Alice: $(N_M + N_T \times N_B) \times 2T$ Alice $\rightarrow$ Bob: $2N_B \times 2T$	Bob: $(N_M + N_T \times N_B)$Enc + $(2 \times N_T \times N_B)$Exp Alice: $2N_B$Enc + $(N_M + 2 \times N_T \times N_B)$Dec

Table 2: Complexity of protocol City block distance.

Round complexity	Communication complexity (bits)	Asymptotic computation complexity
1	Bob $\rightarrow$ Alice : $N_M \times 2T$	Bob: N_M (Enc) Alice: N_M (Dec)

input is $bo(i, j)$ and Bob's, r_i^6 and T_θ. If $\theta_i - \theta'_j$ is smaller than T_θ, ORIDIFF outputs oa_{ij} = "1"; otherwise oa_{ij} = "0". Since $bo(i, j)$ is the result of $e_i - \theta'_j$, $bo(i, j)$ is likely to be negative, and it must be represented as a signed integer. We represent it as a $\mu+\delta+2$-bit integer in two's complement representation to cater to the requirement of circuit SUB [24], and so does r_i^6. In ORIDIFF, the first SUB(in Figure 1, from left to right) outputs the result of $\theta_i - \theta'_j$. However, we want to get its absolute value according to (3). Since the result is represented in two's complement, if it is negative, the second SUB computes its magnitude by subtracting it from $2^{\mu+\delta+2}$. The most significant bit (MSB) of the output of the first SUB, which is the signed bit, controls the selection of the first MUX in ORIDIFF. If the MSB is "0," the MUX chooses the output of the first SUB, otherwise the output of the second SUB. Thus, the first MUX outputs $|\theta_i - \theta'_j|$ which is smaller than 360°. However, the output of the MUX is $\mu + \delta + 1$ bits. Since the bits length of the orientation is μ bits, μ low bits of the output can be only preserved for the next computation. As $\mu + \delta + 1$ is significantly bigger than μ, this method substantially reduces the number of gates. The third SUB in ORIDIFF computes the result of $360 - |\theta_i - \theta'_j|$. And the second MUX in ORIDIFF outputs $od = \min(|\theta_i - \theta'_j|, 360 - |\theta_i - \theta'_j|)$. If there exists the forged input, the $\delta + 1$ high bits of the output of the first MUX may not be zeros. When this happens, we need to set the output of ORIDIFF "0." Besides, the bit lengths of od and T_θ are μ and σ, respectively, and $\sigma$is smaller than μ. So if the high order $\mu-\sigma$ bits are not zeros, od must be greater than T_θ, and there is no need to compare the other bits. Considering the above two cases, in ORIDIFF, we compute the logical OR of high order $\delta+2$ bits of the output of the first MUX and high order $\mu-\sigma$ bits of the second MUX. If the result is 1, the third MUX chooses $2^{\sigma-1}$ as its output, otherwise, the low σ bits of the second MUX. The OR operation can be implemented by $\lambda - \tau - 1$ two-inputs-OR gates one by one. At last, the output of the third MUX is compared against T_θ, if it is greater than T_θ, o_{ij}; the output of ORIDIFF is set to 1, otherwise, 0, which will be further utilized to control the computation related to spatial distance. Bob generates $N_T \times N_Q$ ORIDIFF for Alice to evaluate (3) where $i = 1, \ldots, N_T$ and $j = 1, \ldots, N_Q$.

Figure 2 reveals the circuit of MATCH which serves to uncover the masks, find the matched minutiae, and count their number. The functions of the modules SPADIS, MINM, and COUNTER will be detailed later. Alice's inputs are blind squared-distances $bd(1, 1), \ldots, bd(1, N_Q), \ldots, bd(i, 1), \ldots, bd(i, N_Q), \ldots, bd(N_T, 1), \ldots, d(N_T, N_Q)$, and the corresponding $o_{11}, \ldots, o_{1,N_Q}, \ldots, o_{i,1}, \ldots, o_{i,N_Q}, o_{N_Q,1}, \ldots, o_{N_T,N_Q}$; Bob's inputs are random masks $r_{11}, \ldots, r_{1,N_Q}, \ldots, r_{i1}, \ldots, r_{i,N_Q}, \ldots, r_{N_T,N_Q}$, where $r_{ij} = r_i^3 + r_i^4 + r_{i,j}^5$ and $1 \le i \le N_T$,$1 \le j \le N_Q$. Note that $bd(i, j)$ is larger than r_{ij} and they are both λ-bit positive integers. However, the result of $bd(i, j) - r_{ij}$ is the squared distance $d_E(i, j)$ which can be represented as 2ρ-bit integer ($\lambda > 2\rho$). Let $d_E(i, 1), \ldots, d_E(i, N_Q)$denote the distances between a minutia M_i in M^T and each minutia in M^Q, respectively, $(1 \le i \le N_T)$. Among these distances, only the smallest one that simultaneously meets the requirement specified in (3) is picked out to compare with the threshold T_D. Let τ denote its bits length (note that $\tau < 2\rho$). To reduce the complexity of overall circuits, instead of directly comparing $d_E(i, j)$ with T_D, we generate $d^*(i, j)$ which is only a τ bits value as follows and compare it with T_D later, which can avoid unnecessary bit operations:

$$d^*(i, j) = \begin{cases} 2^\tau - 1, & \text{if } bd(i, j) - r_{ij} \ge 2^\tau \text{ or } o_{ij} = 1 \\ \tau \text{ low-order bits} & \text{if } bd(i, j) - r_{ij} < 2^\tau, o_{ij} = 0, \\ & \text{of } bd(i, j) - r_{ij}. \end{cases} \tag{10}$$

The module SPADIS shown in Figure 3(a) uncovers the masks and compute $d^*(i, j)$. The logical OR of the high $\lambda - \tau$ bits of $bd(i, j) - r_{ij}$ is computed. If the result is 1, $bd(i, j) - r_{ij}$ must be greater than T_D. Hence, there is no need to compare the other bits. The $\lambda-\tau$-bit OR operation can be implemented by $\lambda-\tau-1$ two-inputs-OR gates one by one. Besides, if the bit o_{ij} is 1, it indicates that the orientation difference between M_i and M_j is too large to meet the requirement, so OR of o_{ij} with the result of the above OR operation is further computed. For simplicity, we draw only one OR block standing for these operations in Figure 3(a). The result of the OR block controls the MUX to select $2^\tau - 1$ or the low τ bits of $bd(i, j) - r_{ij}$. Since τ is significantly smaller than λ, the method saves a mass of gates in MIN and CMP circuits.

The module of MINM is presented in Figure 3(b), which takes as the inputs $d^*(i, 1), \ldots, d^*(i, N_Q)$ where $1 \le i \le N_T$

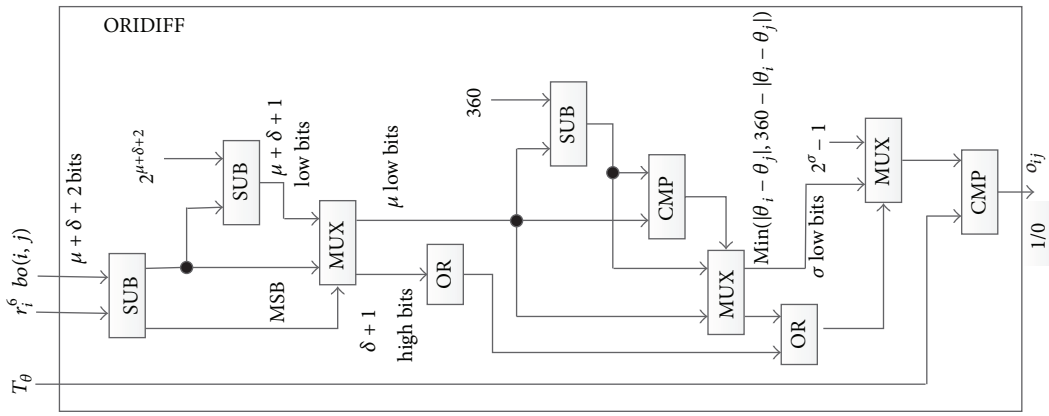

Figure 1: ORIDIFF circuit for computing (3).

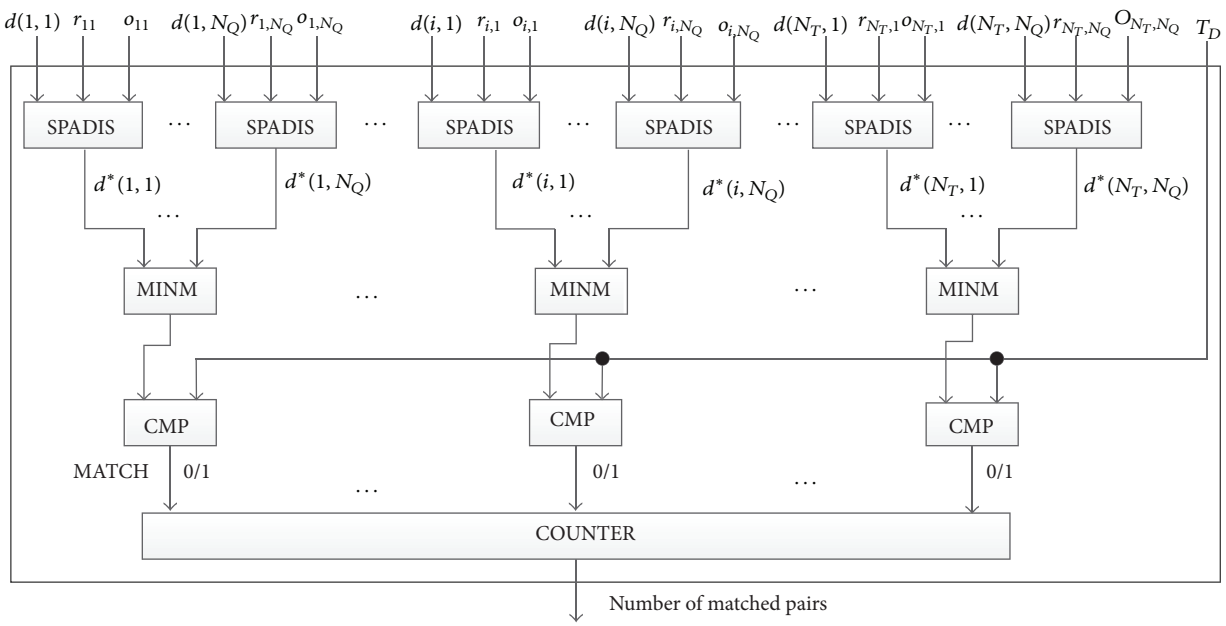

Figure 2: Finding matched minutiae and counting their number.

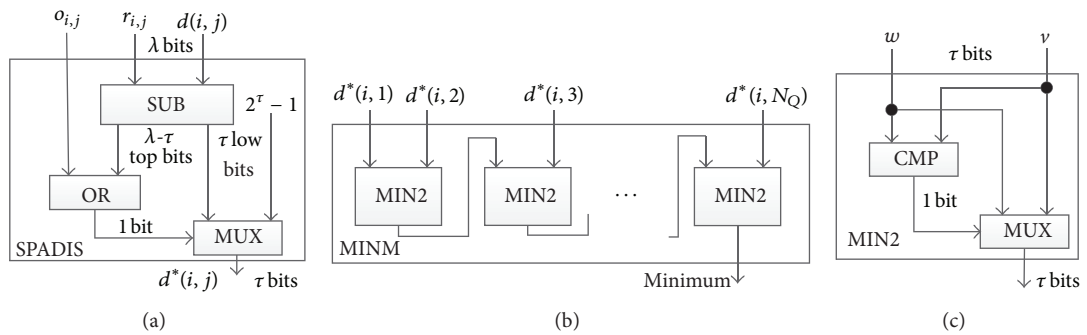

Figure 3: Structure of module SPADIS for Euclidean distance, MINM, and MIN2. (a) SPADIS, (b). MINM, and (c). MIN2.

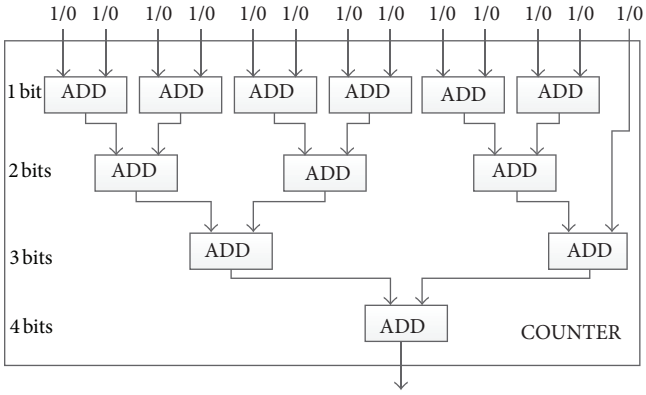

Figure 4: An example of COUNTER with $N_T = 13$.

and picks out the minimum. The circuit MIN2 shown in Figure 3(c) is the functional unit of MINM, which compares two inputs w against v and selects the smaller one.

In Figure 2, if the output of ith CMP is 1 ($1 \le i \le N_T$), it indicates that there exists a minutia belonging to M^Q that matches the ith minutia M_i belonging to M^T. The module of COUNTER further counts the number of these "1"s. Obviously, the number is at most N_T, so $n = \lfloor \log N_T \rfloor$ bits are needed to represent it. To reduce the complexity, we do not use N_T n-bit adders one by one to construct COUNTER. Instead, we use a hierarchical structure, which includes n levels. The first level is composed of 1-bit ADDs with the number of $\lceil N_T/2 \rceil$. The second level is composed of 2-bit ADD with the number of $\lceil \lfloor N_T/2 \rfloor /2 \rceil$, and so forth. The nth level is composed of only one n-bit ADD. Figure 4 shows an example for constructing COUNTER with N_T being thirteen.

We can estimate the cost of circuits used in this section. The blocks here adopt the technique of "free" XOR [23, 24], which do not contribute significantly to the cost of garbled circuits since they need no communicational or cryptographic operations, so we just consider the number of non- XOR gates in the circuits. Table 3 gives the number of non-XOR gates in each of the circuits and the total number.

To implement the authentication, Bob prepares a garbled version of the circuits described above and transfers them to Alice, as well as the garbled values of his inputs—the random masks and the thresholds, T_θ and T_D. Alice carries out the OT protocol along with Bob to obtain the garbled values corresponding to her inputs—$bd(i, j)$ and $bo(i, j)$ for $1 \le i \le N_T$, $1 \le k \le N_B$. She first evaluates ORIDIFF and then MATCH. The final result is the garbled version of the number of the matched minutiae. She sends it to Bob who gets the actual value and computes the matching score according to (1). If the score is greater than the threshold T_M, Bob will accept Alice's identity, otherwise, reject her.

5.2. Matching Circuit for City Block Distance. After the protocol City-Block-Distance is performed, Alice only get the intermediate data—the blind list $\{bx(i, j), by(i, j), bo(i, j) \mid 1 \le i \le N_T, 1 \le j \le N_Q\}$. In this section, we present the

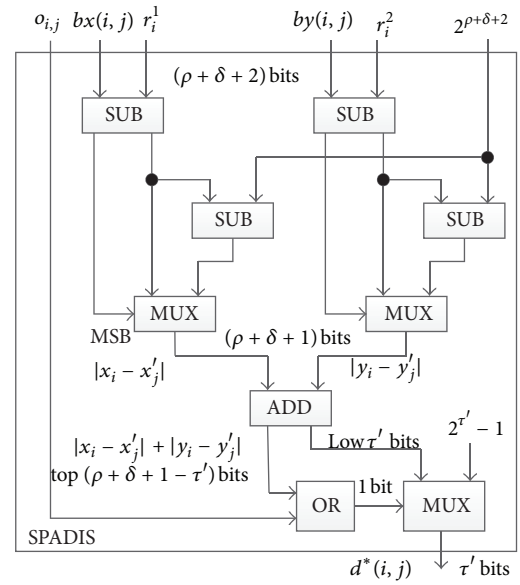

Figure 5: Structure of SPADIS for city block distance.

circuit to complete the computation of city block distance as well as the other subtasks. Firstly, Alice computes the circuits containing $N_T \times N_Q$ ORIDIFF to take off the random masks of $bo(i, j)$ and compute the directional differences, select the minutiae belonging to M^Q, whose directional differences are smaller than T_θ as the candidates. Secondly, she uses the circuits shown in Figure 2 to calculate the distances, find the minima among the candidates, and check whether they are smaller than T_D. However, the module SPADIS of Figure 2 should be replaced with Figure 5 designed for city block distance. Finally, she counts the number of the matched minutiae by using COUNTER described in Section 5.1.

SPADIS in Figure 5 takes as the inputs $bx(i, j)$, r_1^i, $by(i, j)$, r_2^i, and $o_{i,j}$. It firstly takes off the random masks (r_1^i, r_2^i) on $bx(i, j)$ and $by(i, j)$. Note that $bx(i, j)$ and $(by(i, j)$ may be negative since there exists subtraction operation in step 2) of City-Block-Distance, so $bx(i, j)$, r_1^i, $by(i, j)$,and r_2^i all need an additive bit as the signed bits. Therefore, their bits lengths are set to be $\rho + \delta + 2$ when they enter SPADIS. In Figure 5, the left two SUBs and a MUX serve to compute $|x_i - x'_j|$ and the right correspondences serve to compute $|y_i - y'_j|$. The output of ADD is exactly the city block distance $|x_i - x'_j| + |y_i - y'_j|$. To reduce the complexity of whole circuit, we take the same method which has been used in SPADIS for Euclidean distance—the module does not output city block distance directly. Instead, it computes $d^*(i, j)$:

$$d^*(i, j) = \begin{cases} 2^{\tau'} - 1, & \text{if } \left|x_i - x'_j\right| + \left|y_i - y'_j\right| \ge 2^{\tau'} \text{ or } o_{ij} = 1 \\ \tau' \text{ low-order bits of } \left|x_i - x'_j\right| + \left|y_i - y'_j\right|, & \text{if } \left|x_i - x'_j\right| + \left|y_i - y'_j\right| < 2^{\tau'}, \ o_{ij} = 0. \end{cases} \quad (11)$$

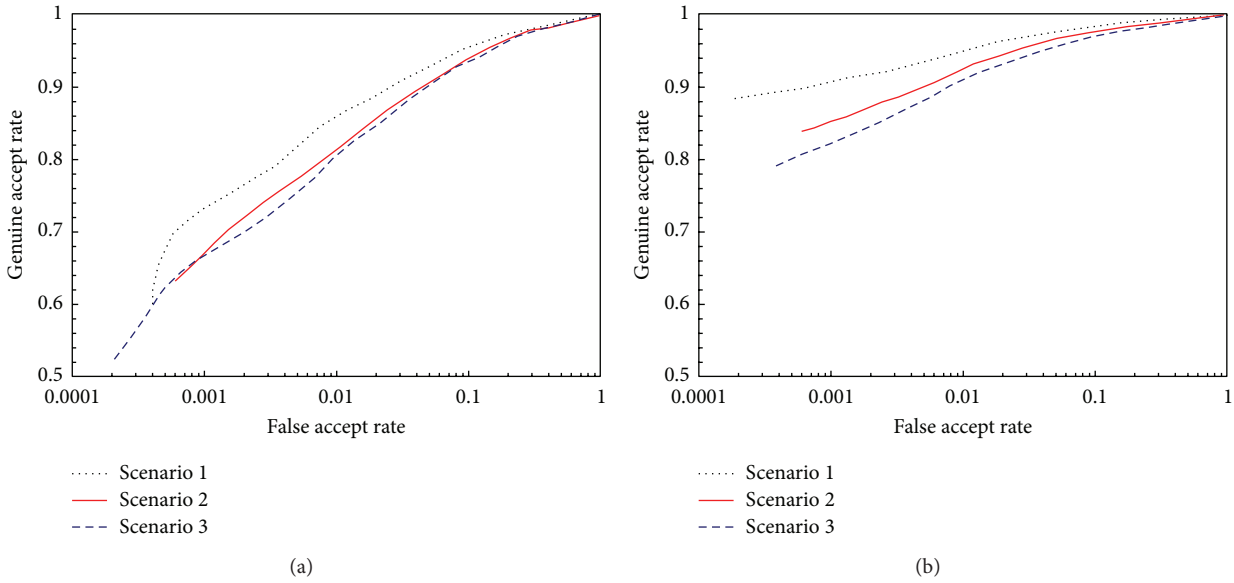

Figure 6: ROC curves for authentication. (a) Euclidean distance, (b) city block distance.

Table 3: Number of non-XOR gates in circuit for Euclidean distance.

ORIDIFF	SPADIS	MIN2	MINM	COUNTER	MATCH	Total
$77\mu+4\delta+\sigma+5$	$4\rho+2\delta+12$	2τ	$2\tau N_Q$	m^*	$(4\rho+2\delta+6+2\tau)N_TN_Q$ $+\tau N_T+m$	$(7\mu+4\rho+6\delta+\sigma+17+2\tau)N_TN_Q$ $+\tau N_T+m$

Note: $^*m = \lceil N_T/2\rceil + 2\lceil\lfloor N_T/2\rfloor/2\rceil + \cdots + \lfloor \log N_T\rfloor$.

The bits length of $d^*(i, j)$ is τ' which is that of the threshold T_D. It is half of τ since the latter is the bits length of squared threshold. The gates of the subsequent circuit also employ this length, so the complexity of the circuit is remarkably reduced. Table 4 gives the complexity for city block distance.

6. Experimental Results and Discussion

As described in the above sections, the shorter bits length for representing a minutia can significantly lead to lower communicational and computational overloads. However, the shorter bits length consequentially may decrease the accuracy of matching. Hence, in this section, we try to evaluate the performance of the matching system and the effect of the bits length on the accuracy through the experiments. To achieve the goals, we implemented the whole system and tested on FVC2002-DB1 fingerprint database [26]. The database contains 8 images of 100 fingers, thus 800 images in total. The size of each image is 388×374 pixels. Consequently, in order to represent each entry of a minutia completely, nine bits are required. The minutiae of each fingerprint were extracted and prealigned by using the algorithm of [27]. We considered the following three scenarios of the bits length: eight bits, seven bits, and six bits. That is, each entry of a minutia, x, y, and θ, was linearly mapped and rounded to 8-bit, 7-bit, and 6-bit integers, respectively. The genuine accept rate (GAR) and false accept rate (FAR) are tested in accordance with the requirements of FVC2002. For a genuine match, each impression of each finger was compared with other impressions of the same finger. There were totally 28 combinations per finger and a total of $28 \times 100 = 2800$ tests that were done for GAR. The cross tests were also done among 100 fingers to evaluate FAR. The first impression of each finger was compared with the first impression of other fingers. Totally, $99 \times 100/2 = 4950$ tests were done.

There are several implementation tools of a generic secure two-party computation that have been developed in the past few years which serve to build privacy-preserving protocols, for example, Fairplay [28], TASTY [29], and FSCUGC [30]. The approach of FSCUGC allows the users to write their programs with a combination of high-level and circuit-level Java code and provides more efficiency and scalability than others with the pipeline technique. So, we employed Java and the library provided by FSCUGC implemented the protocols. In our experimental setting, the server (Bob) and the client (Alice) were set up on different PCs connected by a LAN. Both PCs were configured with AMD 1055T 2.8 G CPU and 8 GB DDR3 memory. We used Paillier encryption scheme with a 1024-bit modulus and 80 bits for symmetric and statistical security. The other parameters of the protocols used in our experiments are listed in Table 5.

Table 6 gives the average running time of our protocols on FVC2002-DB1, in which "*Enc*" refers to the time related to Paillier encryption; "*OT*" refers to that of Oblivious Transfer; and "*Circuit*" refers to that of garbled circuit. "Prep," the preparation phase, only needs to be performed once. Since garbled circuits consume a large amount of memory with

Table 4: Number of non-XOR gates in circuit for city block distance.

ORIDIFF	SPADIS	MIN2	MINM	COUNTER	MATCH	Total
$77\mu+4\delta+\sigma+5$	$8\rho + 8\delta + 10$	$2\tau'$	$2\tau' N_Q$	m^*	$(8\rho + 8\delta + 10 + 2\tau')$ $\times N_T N_Q + \tau' N_T + m$	$(7\mu+8\rho+12\delta+\sigma+15+2\tau')$ $N_T N_Q + \tau' N_T + m$

Note: $^*m = \lceil N_T/2 \rceil + 2\lceil \lfloor N_T/2 \rfloor /2 \rceil + \ldots + \lfloor \log N_T \rfloor$.

Table 5: Bits length of parameters in our experiments.

	ρ	μ	δ	σ	$\tau/\tau'\tau/\tau'$	η_{r_1}	η_{r_2}	η_{r_3}	η_{r_4}	η_{r_5}	η_{r_6}	λ
Scenario 1	8	8	8	4	8/4	14	14	22	22	24	14	25
Scenario 2	7	7	7	3	6/3	13	13	20	20	22	14	23
Scenario 3	6	6	6	2	4/2	12	12	18	18	20	14	21

Table 6: Running time of the proposed methods.

Average times		Euclidean distance			City block distance		
		Scenario 1	Scenario 2	Scenario 3	Scenario 1	Scenario 2	Scenario 3
		s	s	s	s	s	s
Prep	*Enc*	0.158	0.158	0.158	0.048	0.033	0.031
	OT	0.703	0.703	0.696	0.703	0.703	0.698
	Circuit	20. 818	16.392	14.010	28.974	25.869	22.893
Exec	*Enc*	3.301	3.089	2.991	0.048	0.030	0.028
	OT	3.980	3.465	2.904	5.291	4.308	3.497
	Circuit	9.732	8.870	8.684	15.475	13.238	11.772
Average total times		17.013	15.424	14.579	20.814	17.576	15.297

Table 7: Accuracy performance of the proposed methods.

ERR	Euclidean distance			City block distance		
	Scenario 1	Scenario 2	Scenario 3	Scenario 1	Scenario 2	Scenario 3
$T_D T_D = 15$, $T_\theta = 20$	0.039	0.05	0.051	0.025	0.036	0.045
$T_D = 20$, $T_\theta = 30$	0.087	0.092	0.095	0.06	0.075	0.082

the increment of $N_T \times N_Q$, and the average number of the minutiae of a fingerprint approximates to 43 in FVC2002DB1, in order to reduce an excessive exhaustion of the memory, the upper bounds of N_T and N_Q were both set to be 90. Accordingly, the excessive minutiae will be discarded. Step (1) of Euclidean-Distance or City-Block-Distance was computed in the preparation of "*Enc.*" In the preparation phase of "*Circuit,*" the gabled circuits for $N_T = 90$ and $N_Q = 90$ were generated, except for the wire labels and garbled tables which would be regenerated for each execution. The preparation phase of *OT* protocol depends only on the security parameters we choose. The "Exec" in Table 6 refers to execution phase which must be operated for each fingerprint. For the gabled circuits, since he had generated 90×90 circuits, in this phase, the server reset the corresponding wire labels according to actual N_T and N_Q and transferred them to the client. As expected, the average execution time of *Enc* in the case of city block distance is much smaller than that of Euclidean distance. However, the running time is dominated by the computation related to garbled circuits, so the average total time of city block distance is greater than that of Euclidean distance since the circuits of the former are more complex than those of the latter.

Table 7 shows the obtained accuracy measured by Equal Error Rate (EER), while Figure 6 shows the curves of receiver operating characteristic (ROC) of the matcher systems, in which Figure 6(a) is that of Euclidean distance and Figure 6(b) is that of city block distance. As expected, the quantization has effect on the accuracy of the matching algorithm. In both cases, the longer the bits length is, the higher the accuracy is. This can be explained by the fact that that quantization compresses the feature space. The two distinct minutiae with the distance less than a quantization step may be mapped to the same point. And the shorter the bits length is, the higher the probability is. This quantization effect increases FAR, so it lowers the accuracy of the matcher system. Surprisingly, the contrast between Figures 6(a) and 6(b) shows that the matcher based on city block distance has a better accuracy. The result can be explained as follows. We judge whether two minutiae are matched just according to

the distance between them. The decision region formed by Euclidean distance is a circle whereas the one by city block distance is a diamond (square). For the same threshold T_D, the radius of the circle is T_D, while the side length of the diamond is $\sqrt{2}T_D$. Thus, the area of the diamond is only $(\sqrt{2}T_D)^2/\pi(T_D)^2 \approx 63.7\%$ of that of the circle. As mentioned above, the quantization may lead to the merging of distinct minutiae. The bigger decision area, though it increases the GAR, meanwhile, increases FAR more. Hence, it reduces the overall accuracy of the matcher based on Euclidean distance. On the contrary, the smaller area lowers the probability of false matching, thus resulting in the improvement of the accuracy. Compared with the implementation of fingercode version of fingerprint [15], our schemes based on minutiae show a better accuracy performance when $T_D = 15$, $T_\theta = 20$. And, compared with the results of [15], even six bits length also achieved a rather competitive accuracy (ERR = 0.051 for Euclidean distance and ERR = 0.045 for city block distance, resp.) when $T_D = 15$, $T_\theta = 20$.

7. Conclusion

Biometry serves as an excellent mechanism for the authentication of individuals while biometric data are extremely sensitive and must be well protected. Furthermore, once leaked, the data cannot be revoked or replaced. The use of privacy-preserving protocol is a very desirable solution to improving security in biometric applications. In this paper we have addressed the construction of privacy-preserving protocols of fingerprint minutiae based on the combination of garbled circuit and homomorphic encryption. The proposed scheme provides protection for both template and transaction privacy. The template stored on the server is encrypted by the user's private key. Therefore, the template can be updated or revoked by reencryption. Two hybrid protocols with the combination of homomorphic encryption and garbled circuit are presented to fulfill the minutiae matching, in one of which, Euclidean distance is utilized as distance measures and in the other, city block distance is adopted. We have designed the efficient circuits to implement the corresponding tasks. The experimental results on FVC2002-DB2 show that the proposed scheme has acceptable verification accuracy. Future work could be oriented to the application of the results we obtained and to the development of privacy-preserving systems with a higher accuracy and efficiency.

Conflict of Interests

We do not have a direct financial relation that might lead to a conflict of interests for any of the authors.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (61062012).

References

[1] A. K. Jain, K. Nandakumar, and A. Nagar, "Biometric template security," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, Article ID 579416, pp. 1–17, 2008.

[2] A. Juels and M. Wattenberg, "A fuzzy commitment scheme," in *Proceedings of the 6th ACM Conference on Computer and Communications Security (CCS '99)*, pp. 28–36, Singapore, November 1999.

[3] A. Juels and M. Sudan, "A fuzzy vault scheme," in *Proceedings of the IEEE International Symposium on Information Theory*, p. 408, IEEE, Lausanne, Switzerland, July 2002.

[4] A. T. B. Jin, D. N. C. Ling, and A. Goh, "Biohashing: two factor authentication featuring fingerprint data and tokenised random number," *Pattern Recognition*, vol. 37, no. 11, pp. 2245–2255, 2004.

[5] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: how to generate strong keys from biometrics and other noisy data," in *Advances in Cryptology—EUROCRYPT 2004*, vol. 3027 of *Lecture Notes in Computer Science*, pp. 523–540, Springer, Berlin, Germany, 2004.

[6] N. K. Ratha, S. Chikkerur, J. H. Connell, and R. M. Bolle, "Generating cancelable fingerprint templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, pp. 561–572, 2007.

[7] E. Liu, H. Zhao, J. Liang, L. Pang, H. Chen, and J. Tian, "Random local region descriptor (RLRD): a new method for fixed-length feature representation of fingerprint image and its application to template protection," *Future Generation Computer Systems*, vol. 28, no. 1, pp. 236–243, 2012.

[8] W. J. Scheirer and T. E. Boult, "Cracking fuzzy vaults and biometric encryption," Tech. Rep., University of Colorado at Colorado Springs, 2007, http://vast.uccs.edu/~tboult/PAPERS/Scheirer-Boult-BCC07–Crack-Fuzzy-Vault.pdf.

[9] A. Kong, K.-H. Cheung, D. Zhang, M. Kamel, and J. You, "An analysis of BioHashing and its variants," *Pattern Recognition*, vol. 39, no. 7, pp. 1359–1368, 2006.

[10] F. Quan, S. Fei, C. Anni, and Z. Feifei, "Cracking cancelable fingerprint template of Ratha," in *Proceedings of the International Symposium on Computer Science and Computational Technology (ISCSCT '08)*, pp. 572–575, IEEE Computer Society, Shanghai, China, December 2008.

[11] J. Bringer and H. Chabanne, "An authentication protocol with encrypted biometric data," in *Progress in Cryptology—AFRICACRYPT 2008*, vol. 5023 of *Lecture Notes in Computer Science*, pp. 109–124, Springer, Berlin, Germany, 2008.

[12] Q. Tang, J. Bringer, H. Chabanne, and D. Pointcheval, "A formal study of the privacy concerns in biometric-based remote authentication schemes," in *Information Security Practice and Experience*, vol. 4991 of *Lecture Notes in Computer Science*, pp. 56–70, Springer, Berlin, Germany, 2008.

[13] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Privacy Enhancing Technologies*, vol. 5672 of *Lecture Notes in Computer Science*, pp. 235–253, Springer, Berlin, Germany, 2009.

[14] A. Sadeghi, T. Schneider, and I. Wehrenberg, "Efficient privacy preserving face recognition," in *Information, Security and Cryptology—ICISC 2009*, vol. 5984 of *Lecture Notes in Computer Science*, pp. 235–253, Springer, Berlin, Germany, 2009.

[15] M. Barni, T. Bianchi, D. Catalano et al., "A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates," in *Proceedings of the 4th IEEE*

International Conference on Biometrics: Theory, Applications and Systems (BTAS '10), pp. 27–29, IEEE, Washington, DC, USA, September 2010.

[16] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti, "Filterbank-based fingerprint matching," *IEEE Transactions on Image Processing*, vol. 9, no. 5, pp. 846–859, 2000.

[17] Y. Huang, L. Malka, D. Evans, and J. Katz, "Efficient privacy-preserving biometric identification," in *Proceedings of the 18th Network and Distributed System Security Conference*, pp. 6–9, Internet Society, San Diego, Calif, USA, 2011.

[18] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C. V. Jawahar, "Blind authentication: a secure crypto-biometric verification protocol," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 255–268, 2010.

[19] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology—EUROCRYPT '99*, vol. 1592 of *Lecture Notes in Computer Science*, pp. 223–238, Springer, Berlin, Germany, 1999.

[20] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank, "Extending oblivious transfers efficiently," in *Advances in Cryptology—CRYPTO 2003*, vol. 2729 of *Lecture Notes in Computer Science*, pp. 145–161, Springer, Berlin, Germany, 2003.

[21] A. C. Yao, "How to generate and exchange secrets," in *Proceedings of the 27th IEEE Annual Symposium on Foundations of Computer Science*, pp. 162–167, IEEE, Toronto, Canada, 1986.

[22] Y. Lindell and B. Pinkas, "A proof of security of Yao's protocol for two-party computation," *Journal of Cryptology*, vol. 22, no. 2, pp. 161–188, 2009.

[23] V. Kolesnikov and T. Schneider, "Improved garbled circuit: free XOR gates and applications," in *Proceedings of the International Colloquium on Automata, Languages and Programming*, pp. 486–498, EATCS, Reykjavik, Iceland, 2008.

[24] V. Kolesnikov, A. R. Sadeghi, and T. Schneider, "Improved garbled circuit building blocks and applications to auctions and computing minima," in *Cryptology and Network Security*, vol. 5888 of *Lecture Notes in Computer Science*, pp. 1–20, Springer, Berlin, Germany, 2009.

[25] B. Pinkas, T. Schneider, N. Smart, and S. Williams, "Secure two-party computation is practical," in *Advances in Cryptology—ASIACRYPT 2009*, vol. 5912 of *Lecture Notes in Computer Science*, pp. 250–267, Springer, Berlin, Germany, 2009.

[26] D. Maio, D. Maltoni, J. L. Wayman, and A. K. Jain, "FVC2002: second fingerprint verification competition," in *Proceedings of the International Conference on Pattern Recognition*, pp. 811–814, IEEE, Quebec, Canada, 2002.

[27] X. H. Xie, F. Su, and A. N. Cai, "A robust fingerprint minutiae matching algorithm based on the support model," in *Biometric Authentication*, vol. 3072 of *Lecture Notes in Computer Science*, pp. 316–323, Springer, Berlin, Germany, 2004.

[28] D. Malkhi, N. Nisan, B. Pinkas, and Y. Sella, "Fairplay—a secure two-party computation system," in *Proceedings of the 13th USENIX Security Symposium*, pp. 287–302, USENIX, San Diego, Calif, USA, 2004.

[29] W. Henecka, S. Kögl, A.-R. Sadeghi, T. Schneider, and I. Wehrenberg, "TASTY: tool for automating secure two-party computations," in *Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS '10)*, pp. 451–462, Chicago, Ill, USA, October 2010.

[30] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *Proceedings of the 20th USENIX Security Symposium*, pp. 539–554, USENIX, San Francisco, Calif, USA, 2011.