Private Fingerprint Matching

Siamak F. Shahandashti[1], Reihaneh Safavi-Naini[2], and Philip Ogunbona[3]

[1] University of Paris 8 and École Normale Supérieure, Paris, France
fshahand@di.ens.fr
[2] Institute for Security, Privacy, and Info. Assurance, University of Calgary, Canada
rei@ucalgary.ca
[3] ICT Research Institute, University of Wollongong, Australia
philipo@uow.edu.au

Abstract. We propose a fully private fingerprint matching protocol that compares two fingerprints based on the most widely-used minutia-based fingerprint matching algorithm. The protocol enables two parties, each holding a private fingerprint, to find out if their fingerprints belong to the same individual. Unlike previous works, we do not make any simplifying assumption on the matching algorithm or use generic multiparty computation protocols in our constructions. We employ a commonly-used algorithm that works by first comparing minutia pairs from the two fingerprints based on their types, locations, and orientations, and then checking if the number of matching minutia pairs is more than a threshold, and we propose a concrete, scalable, and modular protocol. We prove security against honest-but-curious adversaries and discuss how security against malicious adversaries can be achieved using standard cryptographic techniques. Our protocol is realized using common cryptographic primitives and do not require pairing- or lattice-based cryptography.

1 Introduction

Fingerprints authentication systems are increasingly used as means of verifying the identities of individuals in everyday life. This trend calls for the invention and deployment of more efficient privacy enhancing technologies to protect fingerprint privacy. Fundamental to any fingerprint-based system is the fingerprint matching algorithm which decides whether or not two given fingerprints belong to the same individual. Often the two fingerprints in question are held by two separate entities that are not willing to share unnecessary information with each other. Hence, protocols are required that enable the two parties decide whether or not their private fingerprints match without revealing any further information about them. We call such a protocol a *private fingerprint matching* protocol.

A fingerprint is usually represented by a tuple of *minutiae*, where each minutia is in the form of a vector of elements representing e.g. the type, location, and orientation of the minutia. The fingerprint matching algorithms run in two general steps. First minutia pairs from the two fingerprints are compared based on e.g. the Euclidean distance of their locations and the angular difference of their orientations, and matching pairs are identified. Then a decision is made on whether the fingerprints match based on the outcomes of minutia pair matchings, e.g. based on the number and certain patterns of matching minutiae.

Applications of private fingerprint matching protocols in fingerprint-based authentication systems range from the more classic match-on-server applications to the newer match-on-card applications. The former category of applications employ private fingerprint matching as a building block and include e.g. biometric authentication systems using extended private information retrieval [7, 8]. The latter category of applications employ private fingerprint matching protocols directly and include e.g. the proposal of [2] in which a protocol similar to private fingerprint matching is used. Further applications of private fingerprint matching protocols are conceivable in any situation where the two fingerprints being matched are held by two separate privacy-conscious parties.

Protocols for private fingerprint matching proposed in the literature fall short of providing full privacy without the presence of a trusted third party. In this paper, we propose a private fingerprint

matching protocol which guarantees that *no* information other than the outcome of the matching is revealed to the participating entities.

1.1 Our Contributions

We propose a fully private fingerprint matching protocol that compares two fingerprints based on the most widely-used minutia-based fingerprint matching algorithm. The protocol enables Alice to compare a (private) fingerprint that she holds, with another (private) fingerprint that Bob holds, such that at the end of the protocol, Alice only learns if her fingerprint 'matches' Bob's or not, and Bob does not learn anything.

Unlike (all but one) previous works, we do not make any simplifying assumption on the matching algorithm and base our protocol on a commonly used algorithm that compares minutia pairs of the two fingerprints based on their types, locations, and orientations. A pair of minutiae are defined to match if they have the same type (i.e. they are both ridge endings, bifurcations, etc.), their locations are within a threshold Euclidean distance, and their orientations are within a threshold angular difference. Two fingerprints 'match' if the number of their matching minutia pairs is more than a threshold. The only previous work that is based on the same fingerprint matching protocol uses generic multiparty computation as a building block. Our protocol is concrete and does not rely on any generic building blocks.

One of the main building blocks of our constructions is a primitive that we call *aided computation*. Consider a polynomial $P(x) = \sum a_k x^k$ and a homomorphic encryption algorithm E for which Alice knows the decryption key. It is known that given $\{\mathsf{E}(a_k)\}_{\forall k}$ and x, the homomorphic property of E enables Bob to compute $\mathsf{E}(P(x))$. Aided computation, on the other hand, enables Bob to compute $\mathsf{E}(P(x))$ given $\{a_k\}_{\forall k}$ and $\mathsf{E}(x)$. The underlying idea, which has appeared before in the cryptographic literature, is to simply get Alice to *obliviously* help with the calculation. This can be done in situations like ours where Alice is available for interaction and in effect enables the kinds of computations that only a fully-homomorphic encryption permits, to be performed using only a homomorphic encryption.

A simplified description of our proposed private fingerprint matching protocol is as follows. Let $\{p_i\}_{i=1}^{n}$ and $\{p'_j\}_{j=1}^{m}$ denote Alice and Bob's fingerprint minutiae, respectively. We first use the properties of homomorphic encryption schemes to privately calculate and compare the Euclidean distance and angular difference for each pair of minutiae (p_i, p'_j) to given thresholds. Our protocol enables Bob to calculate $\mathsf{E}(z_{ij})$ for each pair (p_i, p'_j), such that z_{ij} is zero if the two minutiae match and non-zero otherwise. Then, using the aided computation idea, Bob calculates $\mathsf{E}(R(z_{ij}))$, where R is a polynomial that maps zero to one and non-zero values to zero. Let $\sigma = \sum R(z_{ij})$. Now, using the homomorphic property of the encryption scheme $\mathsf{E}(\sigma)$ can be calculated by Bob. Note that σ equals the total number of minutia matchings. Finally, the homomorphic property can be used to finalize the protocol and let Alice find out if σ is greater than or equal to a threshold τ or not.

We provide standard two-party computation security definitions for our protocol and prove its security against honest-but-curious adversaries. Furthermore, we discuss how security against malicious adversaries can be achieved using standard cryptographic techniques. Our protocol is constructed using common cryptographic primitives, such as homomorphic encryption schemes, and do not require pairing- or lattice-based cryptography that imply complex structures and high computational complexity. As an example application, we show how our private fingerprint matching protocol can be integrated into a previously proposed fingerprint-based remote authentication system to enhance the level of privacy it provides for the entities in the system.

1.2 Related Work

There has been proposals in the literature for fingerprint matching protocols in which the two fingerprints are held by two entities. They include the externalized fingerprint matching protocol of [2], the private fingerprint verification protocol of [21], and the minutiae-based fingerprint identification protocol of [3]. We compare our protocol to these proposals in detail in Section 3.2. In brief, the first

two works do not guarantee a high level of privacy, and the last work uses generic multiparty computation protocols in its construction. Therefore, our proposal can be seen as the first fully private minutiae-based fingerprint matching protocol that does not involve generic multiparty computation.

A considerable number of publications that propose match-on-server fingerprint authentication systems are based on approaches to fingerprint matching that either are oversimplifying or simply do not perform as well as minutiae-based matching. Examples of the former include protocols from [27, 7, 8, 28, 5, 6] that privately calculate the Hamming distance between representations of the two fingerprints involved. Examples of the latter are the protocols of [1, 17] that privately implement the FingerCode matching approach which is based on directly comparing the fingerprints as images. Such approaches do not conform with the commonly-preferred matching algorithms based on minutiae and hence we do not consider them solving quite the same problem as ours.

Privacy-preserving protocols for matching other biometric features have been proposed in the literature. Examples include private protocols for identification based on face [26, 12] and iris [3, 4]. Although these works can be seen as related works, best-performing matching algorithms for each feature are highly specialized and do not necessarily work for other features.

There is a considerable body of work concerning biometric template protection, e.g. [20, 11, 19] and subsequent works thereof. Although these works can also be employed to realize authentication systems, they solve inherently different problems and do not consider the case where the inputs to the matching process are held by separate privacy-conscious entities.

The area of privacy-preserving data mining (see e.g. [22, 29]) and specially fuzzy matching (see e.g. [18, 9, 31]) is a related area in that protocols for generic fuzzy matching can be viewed as approximate solutions for the above protocols given appropriate mappings that translate "close"-enough fingerprints or locations to "similar"-enough strings. However, proposals in this area that focus on more generic distance functions that do not solve our specific problems efficiently since precise appropriate mappings that satisfy the above criterion are not easy to find.

2 Preliminaries

In the following we discuss the notation and preliminary definitions that we use throughout the paper.

2.1 Notation

We use the notation $(O_{\mathsf{A}}, O_{\mathsf{B}}) \leftarrow \mathsf{P}[\mathsf{Alice}(I_{\mathsf{A}}) \leftrightarrow \mathsf{Bob}(I_{\mathsf{B}})](I_{\mathrm{p}})$ to denote that a protocol P between a party Alice with private input I_{A} and a party Bob with private input I_{B} is run with public protocol input (i.e. input to both parties) I_{p} and at the end of the protocol Alice's output is O_{A} and Bob's output is O_{B}. If a party has no input or output we use the placeholder $-$. We use $\overset{\mathrm{c}}{\equiv}$ to denote computational indistinguishability of random variables. PPT stands for probabilistic polynomial-time. We use $\mathsf{E}(\cdot)$ and $\mathsf{D}(\cdot)$ short for $\mathsf{E}_{pk}(\cdot)$ and $\mathsf{D}_{sk}(\cdot)$ to denote encryption and decryption using public key pk and secret key sk, respectively.

2.2 Homomorphic Encryption Schemes

An (additively-) homomorphic encryption scheme is one that supports the calculation of addition in the ciphertext domain. That is, for a homomorphic encryption defined on a plaintext field with operations $(+, \cdot)$, with encryption and decryption schemes (E, D), there exists a public operation $\oplus$ such that for any plaintexts a and b: $\mathsf{E}(a + b) = \mathsf{E}(a) \oplus \mathsf{E}(b)$. The existence of such an operation automatically implies that there exists a second public operation $\odot$ such that for any plaintext a and any scalar c: $\mathsf{E}(c \cdot a) = c \odot \mathsf{E}(a)$.

The above properties enable a range of computations in the ciphertext domain. Among these is the ability to compute the encrypted value of a polynomial in a fixed point given the encrypted

values of the polynomial coefficients as follows. Let $P(x) = \sum_{k=0}^{n} a_k x^k$ be a polynomial with coefficients a_k defined over the field. We have the following: $\mathsf{E}(P(x)) = \mathsf{E}(\sum_{k=0}^{n} a_k x^k) = \bigoplus_{k=0}^{n} \mathsf{E}(a_k x^k) = \bigoplus_{k=0}^{n} (x^k \odot \mathsf{E}(a_k))$. We consider schemes that are semantically secure (i.e. ciphertext-indistinguishable under chosen-plaintext attack) [16]. An example of such schemes is Paillier's encryption scheme [24]. We assume that the resulting ciphertexts are rerandomized when $\odot$ and $\oplus$ operations are carried out.

A fully-homomorphic encryption scheme is one that supports the calculation of both addition and multiplication in the ciphertext domain. That is, for such a scheme there exist two public operations $\oplus$ and $\otimes$ such that for any a, b, and c: $\mathsf{E}(a+b) = \mathsf{E}(a) \oplus \mathsf{E}(b)$ and also $\mathsf{E}(c \cdot a) = \mathsf{E}(c) \otimes \mathsf{E}(a)$. Fully-homomorphic encryption schemes have been recently realized [13] but they remain too inefficient to be widely used in practice.

2.3 Security Definitions

Throughout this paper we consider two-party protocols between Alice and Bob in which Bob has no output and Alice's output is supposed to be a function of the protocol's private and public inputs; that is, we consider protocols of the following description: $(f(I_\mathsf{A}, I_\mathsf{B}, I_\mathrm{p}), -) \leftarrow \mathsf{P}[\mathsf{Alice}(I_\mathsf{A}) \leftrightarrow \mathsf{Bob}(I_\mathsf{B})](I_\mathrm{p})$. In the following we review the security definitions for such protocols which are based on the simulation paradigm (a.k.a. the real-vs.-ideal-world paradigm) originating from multi-party computation literature such as [15]. We follow the formalization of [14] and tailor it for protocols of the above-mentioned description.

A *private* protocol is one in which both Alice and Bob learn only their respective protocol outputs and nothing else about the input of the other party. We consider two types of adversary: *honest-but-curious* (a.k.a. semi-honest) and *malicious* (a.k.a. dishonest) parties. An honest-but-curious party is a party that follows the instructions of the protocol, but may record the communications it receives and try to infer extra information using such recordings. A malicious party is a part that does not necessarily follow the instructions of the protocol. First we need to define some notations. The *view* of a party during an execution of the protocol is defined as a tuple containing the party's (private and public) input, random coins, and all the messages it receives during an execution of the protocol. The random variable representing the view of Alice (resp. Bob) during an execution of the protocol P, with Alice's private input I_A, Bob's private input I_B, and public input I_p is denoted here by $\mathrm{View}^{\mathsf{P}}_{\mathsf{A}}(I_\mathsf{A}, I_\mathsf{B}, I_\mathrm{p})$ (resp. $\mathrm{View}^{\mathsf{P}}_{\mathsf{B}}(I_\mathsf{A}, I_\mathsf{B}, I_\mathrm{p})$).

Definition 1 (Privacy against Honest-but-curious Adversaries). *We say that* P privately *computes a function f for an* honest-but-curious *Alice (resp. Bob) if there exists a PPT algorithm* Sim_A *(resp.* Sim_B*) that is able to simulate the view of Alice (resp. Bob) in* P*, given only Alice's (resp. Bob's) respective (private and public) input and output; i.e., for all I_A, I_B, and I_p:* $\mathrm{View}^{\mathsf{P}}_{\mathsf{A}}(I_\mathsf{A}, I_\mathsf{B}, I_\mathrm{p}) \stackrel{\mathrm{c}}{\equiv} \mathsf{Sim}_\mathsf{A}(I_\mathsf{A}, I_\mathrm{p}, f(I_\mathsf{A}, I_\mathsf{B}, I_\mathrm{p}))$ *(resp.* $\mathrm{View}^{\mathsf{P}}_{\mathsf{B}}(I_\mathsf{A}, I_\mathsf{B}, I_\mathrm{p}) \stackrel{\mathrm{c}}{\equiv} \mathsf{Sim}_\mathsf{B}(I_\mathsf{B}, I_\mathrm{p})$*).*

Intuitively, the above definition guarantees that any information revealed to a party during the protocol can be generated by the party on its own from its input and output. Hence, the parties gain no extra "knowledge" as a result of participating in the protocol other than what they are supposed to. For definitions of security against malicious adversaries please refer to [14].

2.4 Fingerprint Minutiae and Fingerprint Matching

The most common method for fingerprint matching is through extraction of minutiae, comparing them based on their types, locations, and orientations, and deciding based on the number of minutia matchings [23]. A fingerprint minutia is in the form (t, x, y, θ), where t, (x, y), and θ determine the type, location, and orientation of the minutia. Minutia type represents the fact that the minutia is e.g. either a ridge ending, bifurcation point, or of other types. We also assume that the size and orientation of each fingerprint is adjusted after acquisition. To this end, first the orientation needs to be adjusted using widely-used *pre-alignment* techniques (see [23] and the references within.). Then

$(-, \mathsf{E}(P(\beta))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(\beta), P)](pk, \eta)$

1. Bob randomizes its input by computing $\mathsf{E}(r\beta) = r \odot \mathsf{E}(\beta)$ for a random r, and sends the result to Alice.
2. Alice decrypts the received ciphertext to get $r\beta$, calculates $\{(r\beta)^k\}_{k=2}^{\eta}$, encrypts the results to ciphertexts $\{\mathsf{E}((r\beta)^k)\}_{k=2}^{\eta}$, and sends them to Bob.
3. Bob derandomizes the results by computing for all $k = 2$ to η: $\mathsf{E}(\beta^k) = r^{-k} \odot \mathsf{E}((r\beta)^k)$ and then computes $\mathsf{E}(P(\beta)) = \bigoplus_{k=0}^{\eta} (a_k \odot \mathsf{E}(\beta^k))$.

Fig. 1. The aided computation protocol

the resulting fingerprint image can be resampled/resized so that the resolution/size of the image conforms to a fixed value.

Two minutiae are considered to match if their types are the same, their locations are closer than a threshold Euclidean distance d_{E}, and their orientations are within an angular difference d_{a} of each other. Let us define the following distance functions for two minutiae $p = (t, x, y, \theta)$ and $p' = (t', x', y', \theta')$: $\mathrm{dis}_{\mathrm{E}}((x, y), (x', y')) = \sqrt{(x - x')^2 + (y - y')^2}$ and $\mathrm{dis}_{\mathrm{a}}(\theta, \theta') = \min(|\theta - \theta'|, 2\pi - |\theta - \theta'|)$ The two minutiae are defined to match if $t = t'$, $\mathrm{dis}_{\mathrm{E}}((x, y), (x', y')) \leq d_{\mathrm{E}}$, and $\mathrm{dis}_{\mathrm{a}}(\theta, \theta') \leq d_{\mathrm{a}}$.

A fingerprint matching can be carried out by checking if the two sets of extracted minutiae have at least a threshold number of matching minutiae in common. In this paper we propose a protocol to carry out the above fingerprint matching *privately*.

2.5 Aided Computation

Assume Alice is in possession of a decryption key for a homomorphic encryption scheme. The basic idea in aided computation is for Bob to employ the ability to interact with Alice to enable him to use the homomorphic encryption scheme as a fully homomorphic one. That is, Alice aids Bob in carrying out operations in the encrypted domain that are not supported by the homomorphic encryption but are possible with the knowledge of the secret key. The aid by Alice must be performed *obliviously* to ensure privacy of the plaintexts corresponding to the involved ciphertexts. This is achieved simply as follows. Bob randomizes the operands, sends the randomized versions to Alice, who then performs the calculations on the randomized plaintexts and sends them back to Bob. Bob eventually derandomizes the received quantity and gets the desired result. Basically, for any encrypted multiplicands $\mathsf{E}(a)$ and $\mathsf{E}(b)$, assuming non-zero a and b, Bob chooses r and s as randomizers and calculates and sends $\mathsf{E}(ra) = r \odot \mathsf{E}(a)$ and $\mathsf{E}(sb) = s \odot \mathsf{E}(b)$ to Alice. Alice decrypts the received quantities and retrieves ra and sb, calculates the product $ra\ sb$, encrypts it, and sends $\mathsf{E}(ra\ sb)$ back to Bob. Bob is now able to calculate an encryption of the product of a and b by derandomizing the encrypted product of ra and sb that he receives from Alice as follows: $\mathsf{E}(ab) = r^{-1}s^{-1} \odot \mathsf{E}(ra\ sb)$. Note that, assuming non-zero a and b, Alice only sees randomized values during this interaction and will not find out any information about a, b, or even ab.

In our protocols, given $\mathsf{E}(\beta)$ Bob often needs to calculate $\mathsf{E}(P(\beta))$ for a polynomial $P(x) = \sum_{k=0}^{\eta} a_k x^k$. Hence we formalize an aided computation protocol $\mathsf{AidComp}$ in which Alice has private input sk, Bob has private input $(\mathsf{E}(\beta), P)$, the public input is a pair (pk, η), and Bob needs to calculate $\mathsf{E}(P(\beta))$. The protocol we propose is based on the idea mentioned above and depicted in Figure 1. We will use this protocol as a building block for our constructions.

An important note is that the protocol requires non-zero β so that $\mathsf{E}(\beta)$ and $\mathsf{E}(r\beta)$ do not leak information about β. Let us call this *the non-zero requirement*. We make sure this condition is met in our constructions.

3 The Private Fingerprint Matching Protocol

We propose the following protocol between Alice and Bob for private fingerprint matching. Let Alice have as private input a set of minutiae $F = \{p_1, \ldots, p_n\}$ where for each $i = 1$ to n: $p_i = (t_i, x_i, y_i, \theta_i)$

and Bob have as private input a set of minutiae $F' = \{p'_1, \ldots, p'_m\}$ where for each $j = 1$ to m: $p'_j = (t'_j, x'_j, y'_j, \theta'_j)$. Alice's private output of the protocol is the binary predicate whether or not there are at least τ number of her minutiae matching those of Bob's, where two minutiae are defined to match if their types are the same, their locations are closer than a threshold Euclidean distance d_{E}, and their orientations are within an angular difference d_{a} of each other.

Let us denote the set of all possible minutia types by T. Consider the set of minutiae $F = \{p_1, \ldots, p_n\}$ where for each $i = 1$ to n: $p_i = (t_i, x_i, y_i, \theta_i)$. For each p_i, let the polynomial Q_i be defined and calculated via the Lagrange interpolation such that we have the following:

$$Q_i(x) = \sum_{k=0}^{|T|-1} b_{ik} x^k = \begin{cases} 0 & \text{if } x = t_i \\ 1 & \text{if } x \in T \setminus \{t_i\} \end{cases} \tag{1}$$

Let us denote the set of all possible Euclidean distances (resp. angular differences) between two arbitrary points (resp. orientations) by D_{E} (resp. D'_{a}). Let $\Delta(d_{\mathrm{E}})$ (resp. $\Delta'(d_{\mathrm{a}})$) denote the subset of D_{E} (resp. D'_{a}) that includes the quantities less than or equal to d_{E} (resp. d_{a}). Assume $N_{\mathrm{E}} = |D_{\mathrm{E}}|$ and $N_{\mathrm{a}} = |D_{\mathrm{a}}|$. Let the polynomials Q_{E} and Q_{a} be defined and calculated via the Lagrange interpolation such that we have the following:

$$\begin{aligned} Q_{\mathrm{E}}(x) = \textstyle\sum_{k=0}^{N_{\mathrm{E}}-1} e_k x^k &= \begin{cases} 0 \text{ if } \sqrt{x-1} \in \Delta(d_{\mathrm{E}}) \\ 1 \text{ if } \sqrt{x-1} \in D_{\mathrm{E}} \setminus \Delta(d_{\mathrm{E}}) \end{cases} \quad \text{and} \\ Q_{\mathrm{a}}(x) = \textstyle\sum_{k=0}^{N_{\mathrm{a}}-1} a_k x^k &= \begin{cases} 0 \text{ if } \sqrt{x-1} \in \Delta'(d_{\mathrm{a}}) \\ 1 \text{ if } \sqrt{x-1} \in D'_{\mathrm{a}} \setminus \Delta'(d_{\mathrm{a}}) \end{cases} \end{aligned} \tag{2}$$

Let the polynomials R and S be defined for τ, m and n and written as follows, where $\tau' = \min(m, n) - \tau + 1$.

$$\begin{aligned} R(x) = \textstyle\sum_{k=0}^{3} R_k x^k &= \begin{cases} 1 & \text{if } x - 1 = 0 \\ 0 & \text{if } x - 1 \in \{1, 2, 3\} \end{cases} \quad \text{and} \\ S(x) = \textstyle\sum_{k=0}^{\tau'} s_k x^k &= \prod_{k=\tau}^{\min(m,n)} (x - 1 - k) \end{aligned} \tag{3}$$

Let $\|F|F'\|_{d_{\mathrm{E}},d_{\mathrm{a}}}$ return the number of minutia pairs that match in the two fingerprints F and F'. We propose the private fingerprint matching protocol PFM depicted in Fig. 2.

Through the initial steps of the protocol, Alice and Bob's interaction enables Bob to compute $\mathsf{E}(Q_i(t'_j))$, $\mathsf{E}(Q_{\mathrm{E}}(d_{ij}^2+1))$, and $\mathsf{E}(Q_{\mathrm{a}}(\gamma_{ij}^2+1))$. Based on Equation 1, $Q_i(t'_j)$ is zero if $t_i = t'_j$ and is one otherwise. Similarly, based on Equation 2, $Q_{\mathrm{E}}(d_{ij}^2 + 1)$ is zero if $d_{ij} \in \Delta(d_{\mathrm{E}})$ and is one otherwise; and $Q_{\mathrm{a}}(\gamma_{ij}^2 + 1)$ is zero if $\gamma_{ij} \in \Delta'(d_{\mathrm{a}})$ and is one otherwise. Hence, z_{ij} is zero if the two minutiae p_i and p'_j match and is one, two, or three otherwise. Therefore, based on Equation 3, $R(z_{ij} + 1)$ is zero if p_i and p'_j match and is one otherwise. This implies that $\sigma = \sum_{\forall i,j} R(z_{ij} + 1)$ reflects the total number of minutia pair matchings $\|F|F'\|_{d_{\mathrm{E}},d_{\mathrm{a}}}$. Now, based on Equation 3, $S(\sigma + 1)$ is zero if $\sigma \geq \tau$ and is non-zero otherwise. Hence, $r'S(\sigma)$ is equal to zero if $\|F|F'\|_{d_{\mathrm{E}},d_{\mathrm{a}}} \geq \tau$ and is random otherwise. Hence, the protocol correctly computes the predicate $\|F|F'\|_{d_{\mathrm{E}},d_{\mathrm{a}}} \geq \tau$. Note that the the non-zero requirement is met.

Security against honest-but-curious adversaries is guaranteed by Theorem 1. The proof comes in Appendix A. Security against malicious adversaries can be achieved via standard cryptographic techniques. Further discussion on how to achieve this can be also found in the appendix.

Theorem 1. *Protocol* PFM *privately computes the predicate* $\|F|F'\|_{d_{\mathrm{E}},d_{\mathrm{a}}} \geq \tau$ *for an honest-but-curious Alice and an honest-but-curious Bob.*

3.1 Practical Considerations

The number of minutiae used by Alice and Bob in protocol PFM, namely n and m, are considered to be public. If privacy of the number of minutiae is required, Alice and Bob can simply agree on a size (or two sizes) beforehand and then adjust the number of minutiae they use as input by either

$(\|F|F'\|_{d_\mathrm{E},d_\mathrm{a}} \geq \tau, -) \leftarrow \mathsf{PFM}[\mathsf{Alice}(F) \leftrightarrow \mathsf{Bob}(F')](n, m, d_\mathrm{E}, d_\mathrm{a}, \tau)$

1. Alice generates a key pair for a homomorphic encryption scheme and sends the public key to Bob. Let the polynomials Q_i, Q_E, Q_a, R, and S be defined as in Equations 1–3. Let $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$.
2. Alice computes for all i: $\{\mathsf{E}(b_{ik})\}_{k=0}^{|T|-1}$, $\mathsf{E}(x_i^2 + y_i^2)$, $\mathsf{E}(x_i)$, $\mathsf{E}(y_i)$, $\mathsf{E}(\theta_i^2)$, and $\mathsf{E}(\theta_i)$ and sends them to Bob.
3. Let $d_{ij} = \mathrm{dis}_\mathrm{E}((x_i, y_i), (x'_j, y'_j))$ and $\gamma_{ij} = |\theta_i - \theta'_j|$. Bob computes for all i and j: $\mathsf{E}(d_{ij}^2 + 1) = \mathsf{E}(x_i^2 + y_i^2) \oplus (-2x'_j \odot \mathsf{E}(x_i)) \oplus (-2y'_j \odot \mathsf{E}(y_j)) \oplus \mathsf{E}(x'^2_j + y'^2_j) \oplus \mathsf{E}(1)$ and $\mathsf{E}(\gamma_{ij}^2 + 1) = \mathsf{E}(\theta_i^2) \oplus (-2\theta'_j \odot \mathsf{E}(\theta_i)) \oplus \mathsf{E}(\theta'^2_j) \oplus \mathsf{E}(1)$.
4. Aided computation $\forall i, j$: $(-, \mathsf{E}(Q_\mathrm{E}(d_{ij}^2 + 1))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(d_{ij}^2 + 1), Q_\mathrm{E})](pk, N_\mathrm{E} - 1)$.
5. Aided computation $\forall i, j$: $(-, \mathsf{E}(Q_\mathrm{a}(\gamma_{ij}^2 + 1))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(\gamma_{ij}^2 + 1), Q_\mathrm{a})](pk, N_\mathrm{a} - 1)$.
6. Let $z_{ij} = Q_i(t'_j) + Q_\mathrm{E}(d_{ij}^2 + 1) + Q_\mathrm{a}(\gamma_{ij}^2 + 1)$. Bob calculates for all i and j: $\mathsf{E}(Q_i(t'_j)) = \bigoplus_{k=0}^{|T|-1} (t'^k_j \odot \mathsf{E}(b_{ik}))$ and $\mathsf{E}(z_{ij} + 1) = \mathsf{E}(Q_i(t'_j)) \oplus \mathsf{E}(Q_\mathrm{E}(d_{ij}^2 + 1)) \oplus \mathsf{E}(Q_\mathrm{a}(\gamma_{ij}^2 + 1)) \oplus \mathsf{E}(1)$.
7. Aided computation $\forall i, j$: $(-, \{\mathsf{E}(R(z_{ij} + 1))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(z_{ij} + 1), R)](pk, 3)$.
8. Let $\sigma = \sum_{\forall i,j} R(z_{ij} + 1)$. Bob computes $\mathsf{E}(\sigma + 1) = \bigoplus_{\forall i,j} \mathsf{E}(R(z_{ij} + 1)) \oplus \mathsf{E}(1)$.
9. Aided computation: $(-, \mathsf{E}(S(\sigma + 1))) \leftarrow \mathsf{AidComp}[\mathsf{Alice}(sk) \leftrightarrow \mathsf{Bob}(\mathsf{E}(\sigma + 1), S)](pk, \tau')$.
10. Bob calculates $\mathsf{E}(r'S(\sigma + 1)) = r' \odot \mathsf{E}(S(\sigma + 1))$ for a random r' and sends it to Alice.
11. Alice decrypts the received ciphertext and decides that $\|F|F'\|_{d_\mathrm{E},d_\mathrm{a}} \geq \tau$ if the resulting plaintext is equal to zero and $\|F|F'\|_{d_\mathrm{E},d_\mathrm{a}} < \tau$ otherwise.

Fig. 2. The private fingerprint matching protocol

omitting a number of minutiae or adding a number of chaff minutiae to their set. Note that since the typical size of a cryptographic plaintext domain is much larger than that of a domain of quantities representing minutiae, it is always possible to generate safe 'unmatchable' chaff minutiae for the above purpose.

Let μ denote the number of minutia pair matchings that need to be carried out in the fingerprint matching protocol. In the case of the PFM protocol in Fig. 2, we have $\mu = nm$; that is, we are assuming that each and every minutia of F is matched against every minutia of F'. Alice's computational complexity in PFM is dominated by $\mathrm{O}(n|T| + \mu(N_\mathrm{E} + N_\mathrm{a}) + \tau')$ encryption computations. Bob's computational complexity in PFM is dominated by $\mathrm{O}(\mu(|T| + N_\mathrm{E} + N_\mathrm{a}) + \tau')$ computations of the operations $\oplus$ and $\odot$. Now note that, the number of possible minutia types $|T|$ is normally only a few, including usually the two types: ridge ending and bifurcation, and possibly up to a couple more. In addition, τ' is less than $\min(m, n)$. Considering the above two facts, Alice's computational complexity boils down to $\mathrm{O}(\mu(N_\mathrm{E} + N_\mathrm{a}))$ encryption computations and Bob's does to $\mathrm{O}(\mu(N_\mathrm{E} + N_\mathrm{a}))$ computations of the operations $\oplus$ and $\odot$.

N_E and N_a respectively represent the number of possible Euclidean distances and angular differences between two minutiae. Consider a fingerprint scanner that outputs fingerprints of the size $w \times h$ pixels and angular precision of $\frac{2\pi}{a}$ radians. Each extracted minutia hence can be represented by coordinates $x \in \{1, \ldots, w\}$ and $y \in \{1, \ldots, h\}$ and an orientation $\theta \in \{0, 1, \ldots, a - 1\}$ (representing radian values $\{0, \frac{2\pi}{a}, \ldots, (a - 1)\frac{2\pi}{a}\}$). Now, we have around $N_\mathrm{E} \approx wh/2$ possible Euclidean distances and $N_\mathrm{a} = a - 1$ possible angular differences between two minutiae. Practically, w and h can have typical values of up to hundreds of pixels but a is usually less than a hundred. Hence the computation complexity is normally dominated by the $\mathrm{O}(\mu N_\mathrm{E})$ factor.

In practice, there is no need to match every minutia of F against every minutia of F'. A large number of minutia matchings can be avoided by fingerprint partitioning; that is, each fingerprint Cartesian plane is divided into a number of regions and minutia matching is only carried out for pairs of minutiae belonging to the same region in the two fingerprints. The partitioning scheme is public and agreed upon by Alice and Bob prior to the execution of the protocol. Assuming that partitioning

is in a way that minutiae are equally likely to fall in regions, each fingerprint region is expected to include n/p minutiae (m/p respectively), where p is the number of regions. Hence, for each pair of corresponding regions $(m/p)(n/p) = mn/p^2$ minutia matchings are required, which implies that the total number of minutia matchings is expected to be $\mu = p \times mn/p^2 = mn/p$. Therefore, the total number of required minutia matchings is reduced by a factor of p using fingerprint partitioning. Furthermore, since the minutia pairs being matched belong to the same region, the number of possible Euclidean distances between the two minutiae is also reduced by a factor of p. The reason is that each region has an expected dimension of $\frac{w}{\sqrt{p}} \times \frac{h}{\sqrt{p}}$. Hence, the total computational complexity of the protocol is reduced by a factor of around p^2 using fingerprint partitioning.

Note that as the number of regions p grows, the precision of fingerprint matching decreases, because matchings between the minutia pairs on the immediate sides of the regions borders are lost as a result of partitioning. Hence there is a trade-off between the computational cost of the protocol and its precision.

Protocol PFM can be used along with fingerprint partitioning with the modification that all the ciphertexts computed by Bob corresponding to minutia pairs (p_i, p'_j) are only for the pairs for which both p_i and p'_j belong to the same fingerprint region rather than for all possible pairs of minutiae. These ciphertexts include $\mathsf{E}(Q_i(t'_j))$, $\mathsf{E}(d_{ij}^2)$, $\mathsf{E}(\gamma_{ij}^2)$, $\mathsf{E}(Q_{\mathrm{E}}(d_{ij}^2))$, $\mathsf{E}(Q_{\mathrm{a}}(\gamma_{ij}^2))$, $\mathsf{E}(z_{ij})$, and $\mathsf{E}(R(z_{ij}))$. To this end, Alice and Bob can carry out the internal steps of algorithm PFM region-by-region. A similar method as was discussed before can be used to hide the number of minutiae in each region if needed.

Let us consider a typical fingerprint reader that can record $n \approx 30$ minutia points with an angular precision of $\frac{\pi}{25}$ radians, i.e. $a = 50$. Also consider a typical dimension of 250×350 pixels, i.e. $w = 250$ and $h = 350$. Employing a partitioning with around a hundred regions reduces the computation complexity of PFM down to as little as around a hundred encryptions for Alice and around a hundred computations of operations $\oplus$ and $\odot$ for Bob, a price that seems quite affordable to be paid for total privacy. Note that homomorphic encryptions are realizable using textbook cryptography and do not require comparatively less widespread, more complex, and computationally more expensive systems such as pairing- or lattice-based cryptography.

We discuss an application of our private fingerprint matching protocol in realizing a remote fingerprint authentication system in Appendix B.

3.2 Comparison

Among the many related works in the literature, there are three proposals that deal with the same problem of private fingerprint matching as we define: the private fingerprint verification protocol by Kerschbaum, Atallah, M'Raïhi, and Rice [21] (KAMR from now on), the externalized fingerprint matching protocol by Barral, Coron, and Naccache [2] (BCN from now on), and the minutiae-based fingerprint identification protocol of Blanton and Gasti [3] (BG from now on). We compare our protocol with these protocols.

The protocol KAMR consists of an alignment phase and a comparison phase. In the alignment phase, the two fingerprints are aligned based on the types, locations, and orientations of their minutiae. During this phase, randomized pairs of minutiae (belonging to the same fingerprint) are sent from one party to the other. In the comparison phase, the fingerprint template containing minutiae is first rasterized to a binary matrix with each element representing if a square-shaped region of the template contains any minutiae or not. Then a private Hamming distance calculation protocol is carried out to find the Hamming distance between the two matrices.

In terms of privacy, the protocol reveals type, location distance, and orientation difference of minutia pairs in the alignment phase and the Hamming distance between the rasterized templates in the comparison phase. The Hamming distance between the rasterized templates in turn reveals the number of minutiae that have the same approximate location in the two fingerprint templates. In contrast, our PFM protocol only reveals the binary outcome of the matching between the two fingerprints.

The underlying fingerprint matching algorithm of the protocol KAMR is based on merely the locations of the minutiae; that is, a minutia pair is considered a matching if they belong to the same square-shaped region of a fingerprint. Furthermore, the square-shaped partitioning of the fingerprint decreases the accuracy of minutiae matching. In contrast, the underlying fingerprint matching algorithm of our protocols is based on both the locations and the orientations of minutiae. In addition, minutia locations are compared based on the Euclidean distance, a meter which is more natural and more precise than the approximate method of checking whether or not minutiae belong to the same square-shaped region. Our protocols are, unlike KAMR, compatible with the most widely-used fingerprint matching algorithms based on Euclidean distances and angular differences of minutiae.

The protocol BCN involves one party sending a set of minutiae to the other party that includes both true minutiae extracted from a fingerprint and false minutiae generated randomly. The receiving party is challenged to distinguish the true minutiae from the false ones.

The protocol does not provide a reasonable level of privacy since the true fingerprint minutiae are revealed in plaintext along with the false minutiae. For example, if the set of minutiae includes n true and n' false minutiae, then any randomly chosen minutia in the set is a true minutia with probability $\frac{n}{n+n'}$. Furthermore, because the false minutiae remain unchanged between multiple protocol runs, all interactions involving the same fingerprint are linkable.

Nevertheless, it should be noted that both of the above protocols incur less computational cost than our protocols. Let us denote the $\oplus$ and $\odot$ homomorphic operations in the ciphertext domain simply by homomorphic operations. Protocol KAMR has a computation complexity of $\mathrm{O}(nm)$ encryptions on one side and $\mathrm{O}(nm)$ homomorphic operations on the other side, where n and m are the number of minutiae in each fingerprint on the two sides. Protocol BCN incurs a computational complexity of only $\mathrm{O}(n)$ exclusive-ors on one side and $\mathrm{O}(nm)$ normal minutia matchings on the other side. However, our PFM protocol incurs a computational complexity of $\mathrm{O}(\mu N_{\mathrm{E}})$ encryptions on one side and $\mathrm{O}(\mu N_{\mathrm{E}})$ homomorphic operations on the other side, where μ is the number of minutia pair matchings (which is initially mn but can be reduced by partitioning), and N_{E} and d_{E} respectively denote the number of possible Euclidean distances (which can also be reduced by partitioning) and the threshold Euclidean distance.

The BG protocol calculates the distance between minutiae using homomorphic encryption techniques similar to ours, but implements the functionalities of counting of the number of matches and comparing it with the threshold, using generic private multiparty computation protocols of garbled circuit evaluation [30] and oblivious transfer [25]. Although the authors show that such generic techniques can be practical for specific choices of parameters and schemes, the constructions lack modularity and immediate scalability. In particular, the protocol seem to work only for a specific homomorphic encryption scheme and the circuits for performing the counting and comparing functionalities need to be redesigned in case of a change in specific parameters such as the number of minutiae in a fingerprint. In contrast, our protocols work for any choice of parameters and homomorphic encryption schemes and are scalable.

In terms of privacy our PFM protocol and BG provide the same guarantee; that is, they both guarantee that no information is revealed to the parties other than the binary output of fingerprint matching.

Based on the above discussions, our fingerprint matching algorithms (as well as BG) are more suitable for cases in which privacy is crucial and the parties are willing to pay the higher computational complexity price, whereas KAMR protocol provides less privacy for less computational complexity, and finally BCN is more suitable for situations where computational resources are scarce on one side.

4 Concluding Remarks

We have proposed for the first time a protocol for private fingerprint verification without generic protocols. The proposed protocol has a linear computation complexity in the number of minutia

matchings and provides the highest possible privacy guarantee, and hence is highly suitable for match-on-server biometric applications. Designing private fingerprint matching protocols with less computation complexity that are more suitable for match-on-card applications remains a challenging problem for future research.

Acknowledgement

The first author would like to thank Josef Pieprzyk and Macquarie University for hosting him during part of this research.

References

1. M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. D. Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and A. Piva. Privacy-Preserving Fingercode Authentication. In *ACM Workshop on Multimedia and Security (MM&Sec '10)*, pages 231–240, 2010.
2. C. Barral, J.-S. Coron, and D. Naccache. Externalized Fingerprint Matching. In Zhang and Jain [32], pages 309–315.
3. M. Blanton and P. Gasti. Secure and Efficient Protocols for Iris and Fingerprint Identification. In V. Atluri and C. Díaz, editors, *ESORICS*, volume 6879 of *Lecture Notes in Computer Science*, pages 190–209. Springer, 2011.
4. C. Blundo, E. De Cristofaro, and P. Gasti. EsPRESSo: Efficient Privacy-Preserving Evaluation of Sample Set Similarity. arXiv report 1111.5062v2 [cs.CR], 2011. `http://www.arxiv.org/abs/1111.5062v2`.
5. J. Bringer and H. Chabanne. An Authentication Protocol with Encrypted Biometric Data. In S. Vaudenay, editor, *AfricaCrypt '08*, volume 5023 of *Lecture Notes in Computer Science*, pages 109–124. Springer, 2008.
6. J. Bringer and H. Chabanne. Another Look at Extended Private Information Retrieval Protocols. In B. Preneel, editor, *AfricaCrypt '09*, volume 5580 of *Lecture Notes in Computer Science*, pages 305–322. Springer, 2009.
7. J. Bringer, H. Chabanne, M. Izabachène, D. Pointcheval, Q. Tang, and S. Zimmer. An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication. In J. Pieprzyk, H. Ghodosi, and E. Dawson, editors, *ACISP '07*, volume 4586 of *Lecture Notes in Computer Science*, pages 96–106. Springer, 2007.
8. J. Bringer, H. Chabanne, D. Pointcheval, and Q. Tang. Extended Private Information Retrieval and Its Application in Biometrics Authentications. In F. Bao, S. Ling, T. Okamoto, H. Wang, and C. Xing, editors, *CANS '07*, volume 4856 of *Lecture Notes in Computer Science*, pages 175–193. Springer, 2007.
9. Ł. Chmielewski and J.-H. Hoepman. Fuzzy Private Matching (Extended Abstract). In S. Jakoubi, S. Tjoa, and E. R. Weippl, editors, *ARES '08*, pages 327–334. IEEE Computer Society, 2008.
10. B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private Information Retrieval. *J. ACM*, 45(6):965–981, 1998.
11. Y. Dodis, L. Reyzin, and A. Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In C. Cachin and J. Camenisch, editors, *EuroCrypt '04*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540. Springer, 2004.
12. Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, , and T. Toft. Privacy-Preserving Face Recognition. In *9th International Symposium on Privacy Enhancing Technologies (PETS '09)*, pages 235–253, 2009.
13. C. Gentry. Fully Homomorphic Encryption Using Ideal Lattices. In M. Mitzenmacher, editor, *STOC '09*, pages 169–178. ACM, 2009.
14. O. Goldreich. *Foundations of Cryptography - Volume II: Basic Applications*. Cambridge University Press, 2004.
15. O. Goldreich, S. Micali, and A. Wigderson. Proofs that Yield Nothing But their Validity and a Methodology of Cryptographic Protocol Design (Extended Abstract). In *FOCS '86*, pages 174–187. IEEE, 1986.
16. S. Goldwasser and S. Micali. Probabilistic Encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.
17. Y. Huang, L. Malka, D. Evans, and J. Katz. Efficient Privacy-Preserving Biometric Identification. In *18th Network and Distributed System Security Conference (NDSS '11)*, 2011.

18. P. Indyk and D. P. Woodruff. Polylogarithmic Private Approximations and Efficient Matching. In S. Halevi and T. Rabin, editors, *TCC '06*, volume 3876 of *Lecture Notes in Computer Science*, pages 245–264. Springer, 2006.
19. A. Juels and M. Sudan. A Fuzzy Vault Scheme. *Des. Codes Cryptography*, 38(2):237–257, 2006.
20. A. Juels and M. Wattenberg. A Fuzzy Commitment Scheme. In *ACM Conference on Computer and Communications Security*, pages 28–36, 1999.
21. F. Kerschbaum, M. J. Atallah, D. M'Raïhi, and J. R. Rice. Private Fingerprint Verification without Local Storage. In Zhang and Jain [32], pages 387–394.
22. Y. Lindell and B. Pinkas. Privacy Preserving Data Mining. *J. Cryptology*, 15(3):177–206, 2002.
23. D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition*. Springer, 2009.
24. P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *EuroCrypt '99*, pages 223–238, 1999.
25. M. O. Rabin. How To exchange Secrets with Oblivious Transfer. Technical Report TR-81, Aiken Computation Lab, Harvard University, 1981.
26. A. Sadeghi, T. Schneider, and I. Wehrenberg. Efficient Privacy-Preserving Face Recognition. In *12th Annual International Conference on Information Security and Cryptology (ICISC '09)*, pages 235–253, 2009.
27. B. Schoenmakers and P. Tuyls. Efficient Binary Conversion for Paillier Encrypted Values. In S. Vaudenay, editor, *EuroCrypt '06*, volume 4004 of *Lecture Notes in Computer Science*, pages 522–537. Springer, 2006.
28. Q. Tang, J. Bringer, H. Chabanne, and D. Pointcheval. A Formal Study of the Privacy Concerns in Biometric-Based Remote Authentication Schemes. In L. Chen, Y. Mu, and W. Susilo, editors, *ISPEC '08*, volume 4991 of *Lecture Notes in Computer Science*, pages 56–70. Springer, 2008.
29. V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-Art in Privacy Preserving Data Mining. *SIGMOD Record*, 33(1):50–57, 2004.
30. A. Yao. How to Generate and Exchange Secrets. In *IEEE Symposium on Foundations of Computer Science (FOCS '86)*, pages 162–167, 1986.
31. Q. Ye, R. Steinfeld, J. Pieprzyk, and H. Wang. Efficient Fuzzy Matching and Intersection on Private Datasets. In D. Lee and S. Hong, editors, *ICISC '09*, volume 5984 of *Lecture Notes in Computer Science*, pages 211–228. Springer, 2009.
32. D. Zhang and A. K. Jain, editors. *Biometric Authentication, First International Conference, ICBA 2004, Hong Kong, China, July 15-17, 2004, Proceedings*, volume 3072 of *Lecture Notes in Computer Science*. Springer, 2004.

A Security of PFM

A.1 Proof of Theorem 1

Proof. For protocol PFM we have: $I_{\mathsf{A}} = F = \{p_1, \ldots, p_n\}$ where for each i: $p_i = (t_i, x_i, y_i, \theta_i)$, $I_{\mathsf{B}} = F' = \{p'_1, \ldots, p'_m\}$ where for each j: $p'_j = (t'_j, x'_j, y'_j, \theta'_j)$, $I_{\mathrm{p}} = (n, m, d_{\mathrm{E}}, d_{\mathrm{a}}, \tau)$, and $f(I_{\mathsf{A}}, I_{\mathsf{B}}, I_{\mathrm{p}}) = (\|F|F'\|_{d_{\mathrm{E}}, d_{\mathrm{a}}} \geq \tau)$. Let us denote the random values Bob generates for the four aided computation protocols during PFM respectively by $\{u_{ij}\}_{\forall i,j}$, $\{v_{ij}\}_{\forall i,j}$, $\{r_{ij}\}_{\forall i,j}$, and ρ. Let us also define $\forall i, j$: $\psi_{ij} = u_{ij}(d_{ij}^2 + 1)$, $\chi_{ij} = v_{ij}(\gamma_{ij}^2 + 1)$, $\omega_{ij} = r_{ij}(z_{ij} + 1)$, $\varsigma = \rho(\sigma + 1)$, $C = \mathsf{E}(r'S(\sigma + 1))$.

First we prove security against an honest-but-curious Alice. To do so we introduce an algorithm $\mathsf{Sim}_{\mathsf{A}}$ that can simulate Alice's view which includes her inputs, randomnesses she uses, and messages she receives. Alice has inputs I_{A} and I_{p} and generates no random values (other than those for encryption) during the protocol. She receives the following messages from Bob during an execution of PFM: $\{\mathsf{E}(\psi_{ij})\}_{\forall i,j}$, $\{\mathsf{E}(\chi_{ij})\}_{\forall i,j}$, $\{\mathsf{E}(\omega_{ij})\}_{\forall i,j}$, $\mathsf{E}(\varsigma)$, and eventually C. Note that all these ciphertexts are encryptions of random plaintexts except for the last one. $\mathsf{E}(r'S(\sigma + 1))$ is either an encryption of zero if $\|F|F'\|_{d_{\mathrm{E}}, d_{\mathrm{a}}} \geq \tau$ or an encryption of a random plaintext otherwise. We define $\mathsf{Sim}_{\mathsf{A}}$ as follows. It takes input I_{A}, I_{p}, and $f(I_{\mathsf{A}}, I_{\mathsf{B}}, I_{\mathrm{p}})$. It generates random plaintexts $\{(\tilde{\psi}_{ij}, \tilde{\chi}_{ij}, \tilde{\omega}_{ij})\}_{\forall i,j}$, $\tilde{\varsigma}$, and $\tilde{P}$. It then sets $\tilde{C} = \mathsf{E}(0)$ if $f(I_{\mathsf{A}}, I_{\mathsf{B}}, I_{\mathrm{p}})$ is true and $\tilde{C} = \mathsf{E}(\tilde{P})$ otherwise. Eventually, it outputs I_{A} and I_{p} as Alice's inputs and $\{\mathsf{E}(\tilde{\psi}_{ij})\}_{\forall i,j}$, $\{\mathsf{E}(\tilde{\chi}_{ij})\}_{\forall i,j}$, $\{\mathsf{E}(\tilde{\omega}_{ij})\}_{\forall i,j}$, $\mathsf{E}(\tilde{\varsigma})$, and eventually $\tilde{C}$ as the simulated messages she receives. The distribution of this output is equal to (and hence

indistinguishable from) the distribution of $\mathrm{View}_{\mathsf{A}}^{\mathsf{PFM}}(I_{\mathsf{A}}, I_{\mathsf{B}}, I_{\mathrm{p}})$. Hence the protocol is secure against an honest-but-curious Alice.

Next, we prove security against an honest-but-curious Bob. To do so we introduce an algorithm $\mathsf{Sim}_{\mathsf{B}}$ that can simulate Bob's view includes his inputs, randomnesses he uses, and messages he receives. His inputs are I_{B} and I_{p}. The random values he generates include: $\{u_{ij}\}_{\forall i,j}$, $\{v_{ij}\}_{\forall i,j}$, $\{r_{ij}\}_{\forall i,j}$, ρ, and r'. The messages he receives include:

$$\begin{aligned}&(pk,\ \{\ \{\mathsf{E}(b_{ik})\}_{k=0}^{|T|-1}, \mathsf{E}(x_i^2+y_i^2), \mathsf{E}(x_i), \mathsf{E}(y_i), \mathsf{E}(\theta_i^2), \mathsf{E}(\theta_i)\ \}_{\forall i},\\ &\quad \{\ \{\mathsf{E}(\psi_{ij}^k)\}_{k=2}^{N_{\mathrm{E}}-1}, \{\mathsf{E}(\chi_{ij}^k)\}_{k=2}^{N_{\mathrm{a}}-1}, \{\mathsf{E}(\omega_{ij}^k)\}_{k=2}^{3}\ \}_{\forall i,j},\ \{\mathsf{E}(\varsigma^k)\}_{k=2}^{\tau'}\ ).\end{aligned}$$

Let $\mathsf{Sim}_{\mathsf{B}}$ be defined as follows. It takes input I_{B} and I_{p}. It first generates a key pair $(\tilde{sk}, \tilde{pk})$ for a homomorphic encryption scheme and also picks a random fingerprint $\tilde{F} = \{\tilde{p}_1, \ldots, \tilde{p}_n\}$ where for each i: $\tilde{p}_i = (\tilde{t}_i, \tilde{x}_i, \tilde{y}_i, \tilde{\theta}_i)$. For each $\tilde{p}_i$, let the polynomial $\tilde{Q}_i$ be defined and calculated via the Lagrange interpolation such that we have the following:

$$\tilde{Q}_i(x) = \sum_{k=0}^{|T|-1} \tilde{b}_{ik} x^k = \begin{cases} 0 & \text{if } x = \tilde{t}_i \\ 1 & \text{if } x \in T \setminus \{\tilde{t}_i\} \end{cases}$$

Let the polynomials Q_{E}, Q_{a}, R, and S be defined as in Equations 2 and 3. It then runs the protocol for both Alice and Bob, assuming that the private input for Alice is $\tilde{F}$, the private input for Bob is I_{B}, and the public input is I_{p}. Hence it calculates the following results and outputs for all i and j for each step of the protocol, respectively: $\{\mathsf{E}(\tilde{b}_{ik})\}_{k=0}^{|T|-1}$, $\mathsf{E}(\tilde{x}_i^2+\tilde{y}_i^2)$, $\mathsf{E}(\tilde{x}_i)$, $\mathsf{E}(\tilde{y}_i)$, $\mathsf{E}(\tilde{\theta}_i^2)$, and $\mathsf{E}(\tilde{\theta}_i)$ for all $i = 1$ to n for Alice, $\mathsf{E}(\tilde{Q}_i(t'_j))$, $\mathsf{E}(\tilde{d}_{ij}^2+1)$, $\mathsf{E}(\tilde{\gamma}_{ij}^2+1)$, $\mathsf{E}(\tilde{\psi}_{ij})$ where $\tilde{\psi}_{ij} = \tilde{u}_{ij}(\tilde{d}_{ij}^2+1)$, and $\mathsf{E}(\tilde{\chi}_{ij})$ where $\tilde{\chi}_{ij} = \tilde{v}_{ij}(\tilde{\gamma}_{ij}^2+1)$ for Bob, $\{\mathsf{E}(\tilde{\psi}_{ij}^k)\}_{k=2}^{N_{\mathrm{E}}-1}$, and $\{\mathsf{E}(\tilde{\chi}_{ij}^k)\}_{k=2}^{N_{\mathrm{a}}-1}$ for Alice, $\mathsf{E}(Q_{\mathrm{E}}(\tilde{d}_{ij}^2+1))$, $\mathsf{E}(Q_{\mathrm{a}}(\tilde{\gamma}_{ij}^2+1))$, $\mathsf{E}(\tilde{z}_{ij}+1)$, and $\mathsf{E}(\tilde{\omega}_{ij})$ where $\tilde{\omega}_{ij} = \tilde{r}_{ij}(\tilde{z}_{ij}+1)$ for Bob, $\{\mathsf{E}(\tilde{\omega}_{ij}^k)\}_{k=2}^{3}$ for Alice, $\mathsf{E}(R(\tilde{z}_{ij}+1))$, $\mathsf{E}(\tilde{\sigma}+1)$, and $\mathsf{E}(\tilde{\varsigma})$ where $\tilde{\varsigma} = \tilde{\rho}(\tilde{\sigma}+1)$ for Bob, $\{\mathsf{E}(\tilde{\varsigma}^k)\}_{k=2}^{\tau'}$ for Alice, and finally $\mathsf{E}(\tilde{r'}S(\tilde{\sigma}+1))$ for Bob. Eventually the protocol outputs I_{B} and I_{p} as Bob's inputs, $\{\tilde{u}_{ij}\}_{\forall i,j}$, $\{\tilde{v}_{ij}\}_{\forall i,j}$, $\{\tilde{r}_{ij}\}_{\forall i,j}$, $\tilde{\rho}$, and $\tilde{r'}$ as the simulated random values he generates, and the following as the simulated messages he receives:

$$\begin{aligned}&(\tilde{pk},\ \{\ \{\mathsf{E}(\tilde{b}_{ik})\}_{k=0}^{|T|-1}, \mathsf{E}(\tilde{x}_i^2+\tilde{y}_i^2), \mathsf{E}(\tilde{x}_i), \mathsf{E}(\tilde{y}_i), \mathsf{E}(\tilde{\theta}_i^2), \mathsf{E}(\tilde{\theta}_i)\ \}_{\forall i},\\ &\quad \{\ \{\mathsf{E}(\tilde{\psi}_{ij}^k)\}_{k=2}^{N_{\mathrm{E}}-1}, \{\mathsf{E}(\tilde{\chi}_{ij}^k)\}_{k=2}^{N_{\mathrm{a}}-1}, \{\mathsf{E}(\tilde{\omega}_{ij}^k)\}_{k=2}^{3}\ \}_{\forall i,j},\ \{\mathsf{E}(\tilde{\varsigma}^k)\}_{k=2}^{\tau'}\ ).\end{aligned}$$

We claim that the distribution of this output is indistinguishable from the distribution of $\mathrm{View}_{\mathsf{B}}^{\mathsf{PFM}}(I_{\mathsf{A}}, I_{\mathsf{B}}, I_{\mathrm{p}})$ for all I_{A}, I_{B}, and I_{p}. Otherwise, there exists some set of inputs F, F', and $(n, m, d_{\mathrm{E}}, d_{\mathrm{a}}, \tau)$ such that $\mathsf{Sim}_{\mathsf{B}}(F', (n, m, d_{\mathrm{E}}, d_{\mathrm{a}}, \tau))$ and $\mathrm{View}_{\mathsf{B}}^{\mathsf{PFM}}(F, F', (n, m, d_{\mathrm{E}}, d_{\mathrm{a}}, \tau))$ are distinguishable. This implies that there exists a PPT algorithm that can distinguish the two pairs $(\tilde{A}, \tilde{B})$ and (A, B), where:

$$\begin{aligned}\tilde{A} &= (\ \{\ \{\mathsf{E}(\tilde{b}_{ik})\}_{k=0}^{|T|-1}, \mathsf{E}(\tilde{x}_i^2+\tilde{y}_i^2), \mathsf{E}(\tilde{x}_i), \mathsf{E}(\tilde{y}_i), \mathsf{E}(\tilde{\theta}_i^2), \mathsf{E}(\tilde{\theta}_i)\ \}_{\forall i}\ )\ ,\\ \tilde{B} &= (\ \{\ \{\mathsf{E}(\tilde{\psi}_{ij}^k)\}_{k=2}^{N_{\mathrm{E}}-1}, \{\mathsf{E}(\tilde{\chi}_{ij}^k)\}_{k=2}^{N_{\mathrm{a}}-1}, \{\mathsf{E}(\tilde{\omega}_{ij}^k)\}_{k=2}^{3}\ \}_{\forall i,j},\ \{\mathsf{E}(\tilde{\varsigma}^k)\}_{k=2}^{\tau'}\ )\ ,\\ A &= (\ \{\ \{\mathsf{E}(b_{ik})\}_{k=0}^{|T|-1}, \mathsf{E}(x_i^2+y_i^2), \mathsf{E}(x_i), \mathsf{E}(y_i), \mathsf{E}(\theta_i^2), \mathsf{E}(\theta_i)\ \}_{\forall i}\ )\ ,\text{and}\\ B &= (\ \{\ \{\mathsf{E}(\psi_{ij}^k)\}_{k=2}^{N_{\mathrm{E}}-1}, \{\mathsf{E}(\chi_{ij}^k)\}_{k=2}^{N_{\mathrm{a}}-1}, \{\mathsf{E}(\omega_{ij}^k)\}_{k=2}^{3}\ \}_{\forall i,j},\ \{\mathsf{E}(\varsigma^k)\}_{k=2}^{\tau'}\ )\ .\end{aligned}$$

since other parts of the simulator output and the view have equal distributions, respectively. Now note that B is calculated based on the description of the protocol using A, F', $\{u_{ij}, v_{ij}, r_{ij}\}_{\forall i,j}$, and ρ. In addition, $\tilde{B}$ is also calculated based on the same set of instructions, but using a different set of inputs, namely $\tilde{A}$, F', $\{\tilde{u}_{ij}, \tilde{v}_{ij}, \tilde{r}_{ij}\}_{\forall i,j}$, and $\tilde{\rho}$. Hence, because $(F', \{u_{ij}, v_{ij}, r_{ij}\}_{\forall i,j}, \rho)$ has the same distribution as $(F', \{\tilde{u}_{ij}, \tilde{v}_{ij}, \tilde{r}_{ij}\}_{\forall i,j}, \tilde{\rho})$, if A and $\tilde{A}$ are indistinguishable then B and $\tilde{B}$ are indistinguishable as well. Thus the fact that there exists a PPT algorithm that can distinguish the two pairs implies that A and $\tilde{A}$ are distinguishable. Using a hybrid argument, this implies that at least one of the elements of A is distinguishable from its counterpart in $\tilde{A}$, which is in contradiction with the semantic security of the encryption scheme. □

A.2 Achieving Security against Malicious Adversaries

Security against malicious adversaries can be achieved via standard cryptographic techniques. To this end, the following steps need to be added to the protocol:

- before Alice starts, Bob must generate random values $\{u_{ij}\}_{\forall i,j}$, $\{v_{ij}\}_{\forall i,j}$, $\{r_{ij}\}_{\forall i,j}$, ρ, and r' and send a commitment to each to Alice;
- then along with pk, Alice must provide a proof that she knows the corresponding secret key;
- along with $\{\mathsf{E}(b_{ik})\}_{k=0}^{|T|-1}$, $\mathsf{E}(x_i^2+y_i^2)$, $\mathsf{E}(x_i)$, $\mathsf{E}(y_i)$, $\mathsf{E}(\theta_i^2)$, and $\mathsf{E}(\theta_i)$, Alice must provide proofs that she knows some set of minutiae that generate these ciphertexts;
- along with $\mathsf{E}(u_{ij}(d_{ij}^2+1))$ and $\mathsf{E}(v_{ij}(\gamma_{ij}^2+1))$, Bob must provide proofs that he knows $\{u_{ij}\}_{\forall i,j}$, $\{v_{ij}\}_{\forall i,j}$, and some set of minutiae such that $\{u_{ij}\}_{\forall i,j}$, $\{v_{ij}\}_{\forall i,j}$ are the values committed before and $\mathsf{E}(u_{ij}(d_{ij}^2+1)) = u_{ij} \odot [\mathsf{E}(x_i^2+y_i^2) \oplus (-2x'_j \odot \mathsf{E}(x_i)) \oplus (-2y'_j \odot \mathsf{E}(y_i)) \oplus \mathsf{E}({x'_j}^2+{y'_j}^2) \oplus \mathsf{E}(1)]$ and $\mathsf{E}(v_{ij}(\gamma_{ij}^2+1)) = v_{ij} \odot [\mathsf{E}(\theta_i^2) \oplus (-2\theta'_j \odot \mathsf{E}(\theta_i)) \oplus \mathsf{E}({\theta'}_j^2) \oplus \mathsf{E}(1)]$;
- along with $\{\mathsf{E}((u_{ij}(d_{ij}^2+1))^k)\}_{k=2}^{N_{\mathrm{E}}}$ and $\{\mathsf{E}((v_{ij}(\gamma_{ij}^2+1))^k)\}_{k=2}^{N_{\mathrm{a}}}$, Alice must provide proofs that they are encryptions of $(u_{ij}(d_{ij}^2+1))^k)$ and $(v_{ij}(\gamma_{ij}^2+1))^k$;
- along with $\mathsf{E}(r_{ij}(z_{ij}+1))$, Bob must provide proofs that these ciphertexts are computed correctly based on the committed u_{ij}, v_{ij}, and r_{ij}, and the ciphertexts received from Alice;
- along with $\{\mathsf{E}((r_{ij}(z_{ij}+1))^k)\}_{k=2}^{3}$, Alice must provide proofs that they are encryptions of $(r_{ij}(z_{ij}+1))^k$;
- along with $\mathsf{E}(\rho(\sigma+1))$, Bob must provide a proof that this ciphertext is computed correctly based on the committed r_{ij} and ρ, and the ciphertexts received from Alice;
- along with $\{\mathsf{E}((\rho(\sigma+1))^k)\}_{k=2}^{\tau'}$, Alice must provide proofs that they are encryptions of $(\rho(\sigma+1))^k$; and
- finally along with $\mathsf{E}(r'S(\sigma+1))$, Bob must provide a proof that he knows an r' such that r' is the value committed before and $\mathsf{E}(r'S(\sigma+1)) = r' \odot \bigoplus_{k=0}^{\tau'}(s_k \odot \rho^{-k} \odot \mathsf{E}((\rho(\sigma+1))^k))$.

Again, assuming that Paillier homomorphic encryption and Pedersen trapdoor commitment are used, all the above proofs are instances of standard discrete-logarithm based witness-indistinguishable proofs that can be realized using techniques in the literature. Furthermore, the witness indistinguishability of the proofs and the trapdoor property of the commitment scheme enables us to prove security against malicious adversaries.

B An Application of PFM in Remote Fingerprint Authentication

As an example match-on-server application, we show that our private fingerprint matching protocol can be used to implement a (remote) fingerprint-based authentication system.

Let us first briefly introduce the system model. The system naturally includes human users U who wish to be authenticated to a possibly remote authentication servers S. Beside the above entities, the model also consists of authentication clients C that extract users' fingerprints through sensors and databases DB that store user fingerprint information. The clients are assumed to be tamper-proof and trusted entities. The databases are also trusted. Links between entities are private and authenticated.

We describe how to use our private fingerprint matching protocol to authenticate a user U to a server S in the above system. S is assumed to hold a key pair for a homomorphic encryption scheme. In the enrollment phase, U registers her (real or pseudonymous) identity ID with S and also registers the same ID along with her fingerprint F' with a database DB. In the authentication phase, first a client C captures a fresh fingerprint F from the user. Then, C, S, and DB run the protocol PFM, with C and S playing Alice and DB playing Bob. C and S carry out steps of the protocol that require the knowledge of F and sk, respectively. Note that in each step of the protocol only one of these two is required. ID identifies the record in the database that must be used as an input to the protocol. At the end, S gets a binary output indicating whether or not the freshly captured fingerprint by C matches the registered fingerprint at the database.

A secure PFM protocol guarantees that S does not find out anything about the registered fingerprint other than the outcome of matching and DB does not find out anything about the fresh fingerprint. Note that since C is assumed to be trusted, our security proofs still hold in the new three-party scenario.

In an alternative scenario, ID is not revealed to DB during the protocol and DB calculates and withholds $\mathsf{E}(r'S(\sigma))$ for multiple records in the database. Eventually, a private information retrieval protocol [10] between S and DB is carried out, with ID acting as the record index, which lets S retrieve the outcome of the matching for the record identified by ID without DB finding out which record among the multiple ones was queried for matching. A desirable property of both of the above scenarios is that U does not need to store any information.

In contrast with the previously proposed match-on-server fingerprint authentication systems for more-or-less the same model as described above that are based on privately calculating the Hamming distance between some representations of the two fingerprints, our private fingerprint matching protocol is based on the widely used minutiae-based matching approach. Our protocol provides a more accurate fingerprint matching based on Euclidean distance, since the Hamming-distance-based protocols are naturally more suitable for fuzzy matching applications such as iris matching.