RICE UNIVERSITY

Robust Privacy-Preserving Fingerprint Authentication

by

Ye Zhang

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Science

APPROVED, THESIS COMMITTEE

Farinaz Koushanfar, Chair
Associate Professor of Electrical and
Computer Engineering

Michael Orchard
Professor of Electrical and Computer
Engineering

Ashok Veeraraghavan
Assistant Professor of Electrical and
Computer Engineering

HOUSTON, TEXAS
December 2015

ABSTRACT

Robust Privacy-Preserving Fingerprint Authentication

by

Ye Zhang

This paper presents the first scalable, efficient, and reliable privacy-preserving minutiae-based fingerprint authentication. Our method is provably secure by leveraging Yao's Garbled Circuit (GC) protocol. While the concept of using GC for secure fingerprint matching has been suggested earlier, to the best of our knowledge, no prior reliable method or implementation applicable to real fingerprint data has been available. Our technique achieves both accuracy and practicability by customizing a widely adopted minutiae-based fingerprint matching algorithm, Bozorth matcher, as our core authentication engine. We modify the Bozorth matcher and identify certain sensitive parts of this algorithm. For these critical parts, we create a sequential circuit description which can be efficiently synthesized using TinyGarble framework. We show evaluations of our modified matching algorithm on a standard fingerprint database to demonstrate its reliability. The implementation of privacy-preserving fingerprint authentication using Synopsis Design Compiler shows the efficiency and scalability of the proposed methodologies.

Acknowledgments

First and foremost, I offer my sincerest gratitude to my advisor, Prof. Farinaz Koushanfar, who has supported me throughout my thesis with her patience and knowledge. I attribute the level of my Master's degree to her encouragement and effort and also without her this thesis would not have been accomplished or written.

In my daily work I have been blessed with a friendly and cheerful group of fellow students. Siam Hussain, Ebrahim Songhori and Sadegh Riazi have provided great suggestions on crypto tools and compiler. Jiandong Mu essentially taught me everything I know about Verilog.

Special thanks are given to Zhenwei Luo, who has helped me in almost every aspect of my academic study and research during my two years at Rice.

The Department of Electrical and Computer Engineering has provided the support and equipment I have needed to produce and complete my thesis.

Finally, I thank my parents for supporting me throughout all my studies at University.

Contents

Acknowledgments.....	ii
Contents	iii
List of Figures	v
List of Tables	vi
Nomenclature	vii
Introduction	8
1.1. Prior Work	12
1.2. Our approach.....	13
Secure Function Evaluation.....	15
2.1. Oblivious Transfer	16
2.2. Yao’s GC Protocol	17
2.3. State-of-the-art GC Optimization	18
2.4. Sequential GC Protocol.....	19
Fingerprint Matcher	21
3.1. Global Flow of a Fingerprint Recognition System	21
3.2. Definition of Minutiae	23
3.3. Bozorth Algorithm and Modification	24
3.3.1. Original Bozorth Description	25
3.3.2. Our Adaptions.....	28
Privacy-Preserving Fingerprint Authentication	30
4.1. A secure 1-to-1 Protocol Construction.....	31
4.2. Security Argument.....	33
4.3. Further Discussion	35
4.4. Implementation.....	36
4.4.1. Circuit Generation.....	36
4.4.2. Synthesis Library	36
4.4.3. Technology Library.....	37
4.4.4. Sequential Customized Bozorth Matcher.....	37

4.5. Evaluation	39
4.5.1. Reliability	39
4.5.2. Timing Performance	41
Conclusion	45
References	46

List of Figures

Figure 3-1 Global Flow of a Fingerprint Recognition System	22
Figure 3-2 Ridge Bifurcation and Ridge Ending	23
Figure 3-3 Minutia-pair	27
Figure 3-4 Minutia-triplet	29
Figure 4-1 Secure Protocol Construction.....	33
Figure 4-2 Customized Bozorth matcher architecture	39
Figure 4-3 GAR and T v.s. Table Length	43

List of Tables

Table 4.5-1 GAR vs FAR for Different Table Length	40
Table 4.5-2 Circuit Size and Timing Evaluation	42

Nomenclature

GAR	Genuine Acceptance Rate
FAR	False Acceptance Rate
NIST	National Institute of Standards and Technology
GC	Garbled Circuit
HE	Homomorphic Encryption

Chapter 1

Introduction

Biometric based methods have been widely deployed in various application domains, ranging from government programs to personal devices. Examples of such applications include international visa system, national ID card, and personal information access control on mobile phones. Biometric traits have the properties of universality (the characteristic is owned by each person), distinctiveness (two individuals are sufficiently different regarding to the characteristic), permanence (invariant of time with respect to matching requirements), and collectability (quantitatively measurable), which ensure the usefulness and high reliability of biometric based applications [1]. However, the popularity of biometric data raises significant privacy concerns as the lost or theft of biometric data is unrecoverable. There are lots of possible attacks that can be applied to a fingerprint authentication system, in our work, we focus on the situations where matching process is

performed in partially untrusted environments. It is of paramount importance to develop techniques that process the biometric data in a privacy-preserving way.

Earlier work in securing fingerprint authentication can be broadly classified into three categories. First, heuristic methods for fingerprint privacy-preservation are available, but these lack provable security guarantees [2]. Second, methods based on trusted hardware or TPM only work in certain scenarios such as mobile phones, but they are inapplicable to third party authentication applications [3]. Third, biometric matching based on secure function evaluation methods, e.g., Homomorphic Encryption (HE) or GC, have only been conceptually suggested without actual realization or implementation [4], [5]. For example, the work in [4] suggested application of HE on an efficient fingerprint matching algorithm based on FingerCode, which did not include implementation or evaluations. The reason is likely the unreliability of the FingerCode for authenticating noisy biometric data. As another example, the authors in [5] proposed using GC on a standard minutiae-based fingerprint matching algorithm which computes the minimum Euclidean distance between two minutiae sets. However, the method assumed a pre-alignment procedure is available and performed matching on adjusted fingerprints to have a reliable match. Having pre-alignment is an unrealistic assumption and including the alignment in their matching algorithm would introduce a huge computational overhead.

In this thesis, we propose the first provably secure, efficient, discriminative and scalable minutiae-based fingerprint authentication that is built upon the classic

Yao's GC protocol. The fingerprint matching problem is as follows. Assume there are two parties, Alice and Bob, where Alice possesses a personal fingerprint image (which has passed the liveness tests linking it to an alive person), and Bob holds a private database with a collection of user fingerprint data. The two parties aim to execute the fingerprint authentication algorithm without revealing their private fingerprint data. In particular, the authentication algorithm determines whether the fingerprint data provided by Alice belongs to the database held by Bob. The settings correspond to real world scenarios where Alice receives authentication requests from a partially untrusted server Bob. For Alice, directly sending her plain fingerprint template to the database is privacy intrusive, as Bob would be able to record Alice's profile if she's not already registered in the database. Also Bob cannot risk exposing its private data to external parties. To achieve a privacy-friendly solution, Alice and Bob agree that the matching algorithm is publicly available for both sides except for the threshold that determines the Genuine Acceptance Rate (GAR) and False Acceptance Rate (FAR) of the fingerprint database system, which is privately owned by Bob. Meanwhile, after running the algorithm Alice must learn only a result showing if a match is found or not. We also assume that Alice and Bob both operate in a semi-honest manner; namely, they are supposed to process the computation following the protocol, but they may try to learn additional information about the other side's private data during protocol execution.

This thesis suggests customizing a highly discriminative alignment-free fingerprint matching algorithm as our core fingerprint authentication engine. As the

accuracy of our modified algorithm is achieved by comparison operations, it is impractical to perform circuit generation based on GC approaches that only accept function description as combinational circuits. A recent breakthrough in this domain was introduced by TinyGarble, which allows describing the function as a sequential circuit and enables utilizing hardware synthesis methods to address the problem [6]. We create a sequential circuit description of selected parts of our customized Bozorth matcher so as to achieve compactness and scalability while adopting the TinyGarble framework.

Our contributions are briefly summarized as follows:

- Introduction of the first scalable, efficient and reliable methodology for privacy-preserving fingerprint authentication based on minutiae representation.
- Development of an efficient and reliable minutiae-based fingerprint authentication algorithm by customizing a widely used fingerprint matching algorithm -- Bozorth matcher.
- Construction of a privacy-preserving protocol for performing fingerprint authentication based on our customized Bozorth matcher, which is described as a sequential logic to reduce circuit size and memory requirement by adopting the TinyGarble framework. Our construction is generalizable to secure multi-party fingerprint authentication.

- Proof-of-concept implementation of our privacy-preserving fingerprint matching algorithm using Synopsys Design Vision on an Intel processor to assess circuit size and timing performance. For instance, we have achieved a highly compact and reliable implementation, which requires 134 KB for storage and 2.24×10^9 clock cycles (0.67s) for circuit garbling/evaluation.

1.1. Prior Work

Several earlier works focus on the area of secure multiparty computation, ranging from the optimization of related cryptographic tools to customized secure protocol generation [7]–[11]. The existence of these tools has motivated new approaches for secure biometric data processing, e.g., [4], [5], [12]–[16]. The available literature has mostly relied on homomorphic encryption which encrypts the data in a way that applying mathematical function on the encrypted message and then decrypts it, generates the same results as performing the operations on the plain message. For example, Barni *et al.* report a secure protocol of FingerCode based authentication by leveraging homomorphic encryption (HE) [4]. Even with protocol-level improvements, their implementation is still impractical for real applications due to the lack of efficiency using HE. By cooperating garbled circuits with homomorphic encryption, Blanton *et al.* present a privacy-preserving protocol for FingerCode based identification with notable reduction in the overhead [5]. FingerCode is a filter-based algorithm, which uses Gabor filters to abstract both local and global features in a fingerprint [17]. The matching proceeds by computing

the Euclidean distance of two corresponding FingerCode vectors. The algorithm is considered to be suitable for efficient implementation in the privacy-preserving framework due to its simplicity. However, as FingerCode is not considered as a discriminative representation of a fingerprint, such approach is unreliable, thus impractical for real applications.

Based on GC, [5] has also proposed a minutiae-based method by finding the maximum number of exclusive minutiae pairs between two fingerprints. Given a minutia in a fingerprint, it is paired with a minutia in another fingerprint which has the minimum Euclidean distance among all unpaired minutiae. However, without pre-alignment, the method cannot handle image rotation and translation, for, e.g., a rotated variant where a certain minutia is moved to a new location far from its original one will not be matched properly, thus directly omitting a pre-alignment stage will dramatically decrease the reliability yet including it in garbled circuit evaluation would introduce huge computation. Their method is not suitable for real world applications either.

1.2. Our approach

In contrast with previous fingerprint matchers, we adopt a robust and highly reliable minutiae-based matching algorithm. The reliability is ensured by employing the relative measurements of each minutia with all the others in a fingerprint. Furthermore, we evaluate our matching algorithm on a standard fingerprint database.

To employ GC for a secure evaluation of the function, a crucial step is to map and optimize the function into Boolean logic. A number of compiler/software techniques have been proposed to realize this conversion [9], [18], [19]. However, they mainly focus on translating the function into combinational description, which incurs huge memory requirements for evaluating functions with a large input size. In biometric applications, the memory requirement for comparing two sets of biometric features is proportional to the square of the size of feature set. A very recent work reported by Songhori *et al.* introduces a novel automated method, TinyGarble, for generating and optimizing compressed Boolean circuits used in Yao's GC protocol [20]. A great improvement in circuit compactness and scalability is achieved by using a sequential circuit description for GC and TinyGarble optimizations.

Chapter 2

Secure Function Evaluation

In this chapter, we will briefly introduce the basic idea of secure function evaluation and some general approaches to implement it. The secure function evaluation problem is often formulated as follows: a given number of participants, $p_1, p_2, \dots, p_n$, which hold private data, $d_1, d_2, \dots, d_n$, respectively, want to compute the value of public function on that private data, $F(d_1, d_2, \dots, d_n)$, while keeping their own inputs private. In our thesis, we mainly focus on the fingerprint authentication problem, which is naturally a two-party secure function evaluation problem, namely, $n = 2$ in previous problem statement. A number of protocols are proposed to address the two-party secure function evaluation problem. A non-negligible question about those protocols is when such a two-party computation protocol is secure. Actually, the security of a two-party protocol is defined against a specific type of adversary. There are essentially two types of adversaries:

- **Semi-Honest Adversary:** In this case, the corrupted parties follow the protocol but try to gather information out of the protocol.
- **Malicious Adversary:** In this case, the corrupted parties may arbitrary deviate from the protocol execution.

A widely adopted protocol that can deal with semi-honest adversary is Yao's garbled circuits (GC) [21]. Yao's GC protocol views the function as Boolean circuit and divides two parties into a circuit generator and a circuit evaluator. The circuit generator needs to prepare a mapping to encrypt the inputs and circuits that can be evaluated by using encrypted inputs. The evaluator encrypts its own inputs according generator's mapping and evaluates the circuits. An essential component of Yao's GC protocol is oblivious transfer, which enables the evaluator to encrypts its inputs conforming with the generator's mapping while learning nothing about the mapping.

2.1. Oblivious Transfer

Oblivious Transfer (OT) is a cryptographic protocol in which a sender transfers one of a set of possible messages to a receiver, yet remains oblivious about which one has been sent. In a 1-out-of-2 OT, S (sender) provides a two-tuple (x_1, x_2) ; R (receiver) provides a selection bit $(0,1)$. S transfers x_σ to R without revealing other values in the tuple, while learning nothing about R's selection [22].

2.2. Yao's GC Protocol

Yao's garbled circuits is a protocol that enables secure two-party computation [21]. Specifically, two parties, Alice and Bob, aim to jointly compute a function f on their secret inputs. The function needs to be represented as a combinational Boolean circuit. The secure computation begins with garbling the circuit. Alice generates random strings corresponding to each of two possible values of a wire. She then maps the plain values of input wires to random symmetric keys and generates an encrypted truth table according to the possible combinations of the input keys. Alice then sends to Bob the encrypted truth table and key corresponding to her input value. Now to evaluate f , Bob needs to obtain his input without revealing it to Alice. To do this, Bob obviously receives the key corresponding to his input through oblivious transfer (OT). Bob obtains the output key based on the received input keys and the encrypted truth table. Finally, Alice provides a mapping from the encrypted output to plain output. This protocol can be easily generalized to circuit consists of multiple gates. In the garbled circuit generation phase, Alice chooses keys for all the wires and constructs encrypted truth tables for each gate. In the evaluation phase, Bob evaluates the circuit gate-by-gate using these keys and encrypted truth tables until he evaluates all gates. A full description and comprehensive proof of security of Yao's garbled circuits is given by Lindell and Pinkas [23].

2.3. State-of-the-art GC Optimization

After the first introduction of garbled circuits protocol by Yao, a number of optimizations have been proposed to reduce the communication and evaluation overhead. The state-of-the-art optimizations for garbled circuits can be mainly categorized into three aspects:

- **Free XOR [24]:** In this method, Alice generates a global random private key R . During the garbling stage, for each wire w_a , she only generates a token X_a^0 and computes the other token X_a^1 as $X_a^1 = X_a^0 \oplus (R || 1)$. Therefore, the token for the output wire of the XOR gates with input wires w_a, w_b and output wire w_c can be computed as $X_c = X_a \oplus X_b$. This technique enables XOR gate to be evaluated freely without garbling. The proof of security is given in [24].
- **Row Reduction [25]:** This optimization substitutes the random generation of token for the output wire of a gate by producing the output token as a function of the tokens of the inputs. Using this generation scheme, the first entry of the garbled table becomes all 0 and no longer needs to be sent, thus reducing the size of the tables for the non-XOR gates by 25%.
- **Garbling With a Fixed-key Block Cipher [26]:** In this garbling scheme, the output key X_c is encrypted with the input token X_a and X_b using the encryption function $E(X_a, X_b, T, X_c) = \pi(K) \oplus K \oplus X_c$, where $K = 2X_a \oplus 4X_b \oplus T$, π is a fixed-key block cipher, (e.g., instantiated with AES) and T is a unique-per-gate number (e.g., gate identifier) called tweak. This method allows to

efficiently garble and evaluate non-XOR gates using fixed-key AES. The proof of security is given in [26].

2.4. Sequential GC Protocol

For a long time, the research about GC protocol has been centered on improving the garbling/evaluation efficiency of combinational circuit. However, the combinational circuit suffers from scalability issue when the number of inputs increasing. The main reason is that the circuit size (or total number of gates) of combinational logic is proportional to the number of inputs. Another class of circuit whose circuit size (or total number of gates) is invariant to the number of inputs is sequential circuit. Therefore, the garbled sequential circuit should be more scalable than the garbled combinational circuit [20].

A sequential circuit can be represented as an ensemble of a combinational circuit and feedback loops with memory elements. At each clock cycle, circuit input as well as the present states are fed to the combinational part. Then it generates the outputs and next states which will be stored in the memory element for next cycle. Evaluating the sequential circuit for a specified c times can be visualized as evaluating the unrolled combinational circuit which is comprised of c combinational parts of the sequential circuit. Therefore, the garbling and evaluation of the garbled sequential circuit is equivalent to combinational circuit after a proper treatment against memory element.

During generation and evaluation of the garbled circuit, it must be ensured that the encryption tweaks T (see 2.3) for each gate is unique or otherwise the security is broken [20], [26]. In garbled sequential circuit, to ensure the uniqueness, Songhori *et al.* proposed setting tweak T for each gate to be the concatenation of the cycle index (cid) and the unique gate identifier (gid) in the combinational part of the sequential circuit, i.e., $T = \text{cid} || \text{gid}$ [20].

Chapter 3

Fingerprint Matcher

In this chapter, we first present a brief introduction of fingerprint matching, including the general mechanism of an automated fingerprint recognition system and the special features used for fingerprint template construction. Then we will focus on introducing a robust fingerprint matcher—Bozorth. Since the original Bozorth matcher is computational intensive, we propose a number of adaptations to minimize the computational requirements while preserving most of the discriminability. Such adaptations enable our fingerprint matcher to fit in real world privacy-preserving fingerprint authentication scenario.

3.1. Global Flow of a Fingerprint Recognition System

A fingerprint recognition system can be used for verification/authentication (1:1 matching) and identification (1:N matching). It contains two stages: an enrollment phase and an identification phase. Adapted from Jain et al. [27], Figure 3-1 outlines a typical automated fingerprint recognition system.

Figure 3-1 Global Flow of a Fingerprint Recognition System

During the enrollment phase, an optical sensor scans user's fingerprint and convert it to a digital file. Then a minutiae abstract extract the specific details of a fingerprint, called minutia points, which we will be discussing in more details in next section. The location and orientation of each minutia is stored to construct a fingerprint template, named gallery fingerprint template), which is stored together with user's demographic information in the database. Noticing that a typical

fingerprint template contains information of 20-70 minutia points; depending on the resolution of the image and size of the sensor surface.

During the identification phase, the user again presses his/her finger on the same sensor used for enrollment to generate a new fingerprint image. Minutiae extractor then record minutiae information and construct a new fingerprint template, named query fingerprint template. Then a matcher is responsible for determine the similarity between the query fingerprint template and the gallery fingerprint template in the database and compute a match score. Finally, the match score is compared with a threshold set by the system administrator to determine whether two fingerprints match or not.

3.2. Definition of Minutiae

Figure 3-2 Ridge Bifurcation and Ridge Ending

In biometrics, minutiae are the major features of fingerprint. As Figure 3-2 shows, these features include points in a finger's friction skin where ridges end (called a ridge *ending*) or split (called a ridge *bifurcation*). Typically, there are on the order of 100 minutiae on a tenprint. To characterize each minutia point, its coordinate location and the orientation is often recorded, while some minutia detection programs also record the type of each minutia point. The location of each minutia is represented by the pixel coordinate within the fingerprint image[28]. The orientation is represented in degrees, and measured counter-clockwise with zero degrees pointing horizontal and to the right[28].

3.3. Bozorth Algorithm and Modification

Various algorithms have been proposed and implemented for matching two fingerprints with minutiae representation. A type of method for matching two minutiae-based fingerprint templates contains an alignment stage, which maximizes the number of corresponding matching minutiae, and a Euclidean distance computation stage, in which the similarity of two fingerprints is evaluated [29, p. 200], [30], [31]. Since additional local features of a fingerprint, such as core or delta points, are always required to assist alignment, an optimal alignment cannot be guaranteed given images with lower resolution or partial information where reference points are missing. In order to keep generality, several research projects focus on developing rotation and translation invariant fingerprint matching algorithms [32], [33, p. 200], [34], among which Bozorth algorithm developed by

National Institute of Standards and Technology (NIST) stands out due to its robustness and reliability.

The official description of Bozorth matcher is given in [28]. Fingerprint templates with minutiae represented by location and local ridge orientation are taken as the inputs of an original Bozorth matcher. The algorithm includes three major steps:

- Constructing intra-fingerprint minutia-pair tables.
- Constructing inter-fingerprint pair-pair (compatibility) table by comparing the two intra-fingerprint minutiae pair tables, where sufficiently similar minutia pairs are considered to be compatible.
- Traversing the inter-fingerprint compatibility table to build a web/cluster and accumulate a match score.

3.3.1. Original Bozorth Description

The first step in Bozorth algorithm is to construct an intra-fingerprint minutia-pair table (or minutia-pair comparison table) by computing the relative measurements of each minutia in a fingerprint with all the others within a specified distance. As it is shown in Figure 3-3, each minutia pair consists of 6 pieces of information, namely, the distance between the pair d_{ij} , the orientations (β_1, β_2) of minutiae with respect to the intervening line connecting them, the indexes (i, j) of minutiae in a minutia pair, and the absolute angle θ_{ij} of the intervening line. Each

entry of the comparison table is represented by $\{d_{ij}, \beta_1, \beta_2, i, j, \theta_{ij}\}$. Among these relative measurements, d_{ij} , β_1 and β_2 remain nearly constant no matter how much the fingerprint is rotated or translated. This property ensures the algorithm's rotation and translation invariance. The next step is to find the compatible entries from minutia-pair comparison tables of two separate fingerprints. Two minutia-pairs are compatible if the corresponding distance d_{ij} and relative minutia angles (β_1, β_2) are within some specified tolerances. The entry of the compatibility table incorporates the information about two minutia-pairs. It consists of $\{\Delta(\theta_{ij,c}, \theta_{ij,s}), i_c, j_c, i_s, j_s\}$ represents the angular difference between the intervening lines of two minutia-pairs from two fingerprints, and (i_c, j_c) stand for the indices of minutia pair from client, (i_s, j_s) are the indices of corresponding minutia-pair from server. At this point, a compatibility table consists of a list of compatibility associations between two pairs of potentially corresponding minutiae has been constructed. The last step is traversing the compatibility table to find the longest path of linked minutia-pairs. We then assign the length of the longest path to match score.

Figure 3-3 Minutia-pair

3.3.2. Our Adaption

In the final stage, the time complexity for searching the longest path that links minutia-pairs exponentially grows with respect to the path length. In order to achieve an efficient implementation, we change the objective from finding the longest path to counting the number of compatible minutia-triplets. As it is shown in Figure 3-4, we define a compatible minutia-triplet (i, j, k) as two minutia-pairs, (i, j) and (i, k) in the compatibility table sharing the same starting point i . Two minutia-triplets are considered to be compatible if they satisfy two requirements. Firstly, they are formed by two entries in the compatibility table, one of whose starting minutia indexes are the same as the corresponding ones of the other ($i'_c = i''_c$ and $i'_s = i''_s$). Secondly, the difference between the global orientations of these two entries is within a certain acceptable tolerance, $(\Delta(\theta_{ij,c}, \theta_{ij,s})) < t$. We observe from the requirements that only the indices of starting points in each entry of the compatibility table are used for counting compatible triplets. Therefore, each entry of the compatibility table can be constructed in the form of $\{\theta'_{ij}, i_c, i_s\}$, where the indices of the ending points of the compatible minutia-pairs are omitted. We refer to this table as partial compatibility table.

So basically, what we did is that we construct a more discriminative local feature(minutia-triplet) by setting constraints to compatible minutia-pairs. As the objective of the last step in the original Bozorth algorithm is to find the maximum similar pattern that exist between two fingerprints, instead of directly searching for

the longest path, we measure these local features. It's very reasonable to assume the larger the number of compatible minutia-triplets, the more likely there exist larger similar pattern, the more likely two fingerprints match.

In the following section, we propose a privacy-preserving protocol for performing fingerprint authentication based on the customized Bozorth algorithm.

Figure 3-4 Minutia-triplet

Chapter 4

Privacy-Preserving Fingerprint Authentication

In this chapter, we introduce a secure realization for a 1-to-1 fingerprint matching algorithm, which can easily be extended to 1-to-n scenario. We work in an honest-but-curious attack model. In order to achieve an efficient implementation, we have designed a pre-computation stage for both parties to compute their interminutia-pair tables, and an offline computation stage on the client side to perform the final step of our matching algorithm, thus greatly reducing the amount of computation enrolled in garbled circuit. Besides, for security consideration, the intermediate results are encrypted by server before sending to client. In consistent with most related work, we do not set the user name as part of secret input for client.

4.1. A secure 1-to-1 Protocol Construction

Consider two parties, Client and Server (our earlier Alice and Bob), jointly run the customized Bozorth algorithm. We assume that both sides know the length of each other's intra fingerprint minutia-pair table so as to determine the total number of rounds (by multiplying the lengths of two minutia-pair tables) required to run the matching algorithm. S (Server) owns a database with templates in the form of $Y_c = \{(x_1^c, y_1^c, \alpha_1^c), \dots, (x_n^c, y_n^c, \alpha_n^c)\}$ and a private key $\mathbf{K}$, where (x, y) is the minutia location, and α is the direction of local ridge in which minutia resides, while C (Client) has a biometric template $X = \{(x_1, x_2, \alpha_1), \dots, (x_m, y_m, \alpha_m)\}$ consists of m minutiae. To perform authentication, C obtains the fingerprint template and claims his/her identity, then S provides a private key and the corresponding template with the same ID, Y_c , as the inputs of the protocol. The detailed protocol is presented in Figure 4-1. One point worth noticing is that the client is able to count the number of compatible triplets with the encrypted compatibility table it receives in step 3. The index of the starting minutia of a minutia-pair from the client's fingerprint in each compatibility table entry is transformed by linear transformation and then encrypted using XOR encryption. The linear transformation maps an index to a value in a larger range. The same index is mapped to the same value as we fix the linear transformation. Since the indexes are encrypted with the XOR encryption scheme, which is realized by performing XOR operation on a private key and plain text, the same plain text maps to the same encrypted text. Therefore, our scheme guarantees that the same index is encrypted to the same value. When counting the

number of compatible triplets, we only need to check whether the starting indexes are equal, thus our encryption scheme will not affect these comparisons.

Our scheme can be extended to a 1-to-n scenario. It requires three steps for the client to match a fingerprint in a database without claiming his/her identity. Firstly, the client sends his/her encrypted minutia-pair comparison table to the server. Subsequently, the similarity between the client's fingerprint and all the others in server's database is calculated by iteratively running the protocol. Since the identity should be reported together with the best match score, the server is also responsible to keep track of the IDs during protocol execution. Finally, the one with the maximum match score is selected to be the best matching fingerprint template; if it is within a specified threshold, a match result is reported with registered user ID.

INPUT : C provides a minutiae file $X = \{(x_1, y_1, \alpha_1), \dots, (x_m, y_m, \alpha_m)\}$, S provides a private key $\mathbf{K}$ and $Y_c = \{(x_1^c, y_1^c, \alpha_1^c), \dots, (x_n^c, y_n^c, \alpha_n^c)\}$.

OUTPUT : C learns a single-bit comparison result indicating if it belongs to a database held by S.

Protocol Steps:

- 1) Pre-Computation:
  - a) S computes a minutia pair table for the template registered in the database.
  - b) C computes a minutia pair table based on its fingerprint template.
- 2) Online computation:
  - a) C generates the garbled circuit and sends it to S together with its encrypted pair-wise minutia-pair data.
  - b) S gets its encrypted minutia-pair data through oblivious transfer and computes a garbled incomplete minutia-pair compatibility table, where the second column (the index of the starting point of a minutia-pair derived from C) is encrypted by S's private key $\mathbf{K}$ using a linear transformation and XOR encryption, then sends it back to C.
- 3) C decrypts the encrypted incomplete minutia-pair compatibility table and gets a partially encrypted incomplete compatible table, where the number of compatible triplets r_c can be calculated.
- 4) C and S jointly run a secure protocol based on garble circuits, which compares r_c with the threshold r_s provided by the server. S generates the circuit and sends it to C together with the encrypted-output-to-plain-output mapping. C then evaluates the circuit and learns the one-bit result.

Figure 4-1 Secure Protocol Construction

4.2. Security Argument

In this section, we briefly present a security argument for our scheme, where partial intermediate results are visible to one party. In particular, our goal is to demonstrate that, the client is not able to learn anything beyond what is revealed by the protocol, about the server's database, while the server should not get any information about the client's minutiae as well as the result of the matching process.

We analyze our protocol step by step. Firstly, since the construction of minutia-pair comparison table for both parties are carried out independently, no information is possibly revealed. In the online computation phase, secure data transmission is done through oblivious transfer, whose security has been discussed in previous section. Subsequently, it is clear that all messages the server received and processed is encrypted, thus a complete privacy is ensured. The final stage compares two values under garbled circuit protocol.

The remaining part is the third step of the protocol. One might be concerned that the client learns additional knowledge about the database as the client has access to the partially encrypted version of incomplete minutia-pair compatibility table. The second item of each entry in the compatibility table is the index of the starting minutia of a minutia-pair from the client's fingerprint, which is encrypted by the server using XOR encryption and a linear transformation during garbled circuits evaluation. By doing this, we first map the indices (typically from 1 to 40) to a larger range of integers. Then the XOR encryption perturbs the order of the transformed indices. Thus, owning the encrypted index, the client is unable to locate the corresponding minutia-pair in its own minutia-pair comparison table. It's unlikely that the server's minutiae information can be inferred. We thereby demonstrate the partially encrypted incomplete pair-pair (compatibility) table is of no use in recovering the other party's minutiae information.

4.3. Further Discussion

In this section, we first discuss what the two parties could learn besides the final matching result during the protocol execution and possible improvements or customization that could be applied on our protocol.

As the client is responsible for processing the encrypted incomplete compatibility table, it learns the table length, namely, the number of compatible minutia-pairs, as inputs. After the computation, it knows the number of compatible minutia-triplets. Since all computation performed by the server is evaluating garbled function, the protocol reveals no additional information.

In our settings, the threshold that determines the system's FAR and FNR is privately held by the server. We concern the situation where the client, who has already obtained the number of compatible minutia-triplets, if having access to the system threshold, would be able to estimate the similarity between the two fingerprints more accurately. This greatly increases the probability of a successful hill climbing attack to a fingerprint system. That's the reason we include one more garble circuit evaluation stage to hide the threshold. However, in scenarios where the client is nothing more than an honest-but-curious party, we could release this stage to the software as well.

4.4. Implementation

In this section, we present the implementation of the sensitive parts (inter-fingerprint pair-pair (compatibility) table construction) of our customized Bozorth matcher based on Yao's GC protocol. We start by introducing the optimized circuit generation for GC. Then we present the compact realizations for our matching algorithm based on a sequential circuit.

4.4.1. Circuit Generation

Yao's GC protocol requires a function to be represented as a Boolean circuit. To implement our fingerprint matching algorithm, we first describe it in Verilog, then use existing HDL synthesis tools to map the Verilog description to a list of basic binary gates, namely, netlist. GC implementation used in our work is based on TinyGarble [20].

4.4.2. Synthesis Library

The implementation of fingerprint matching algorithm requires a set of arithmetic and conditional operations, such as addition, subtraction and if-else. Synthesis library contains various implementations to convert those operations to their logical representation. Songhori *et al.* customized the synthesis library of TinyGarble to include the minimum non-XOR implementation of those operations [6]. However, their synthesis library only contains the implementation for unsigned operations, while our matching algorithm needs to deal with signed operations.

Thus we implement our own synthesis library which includes the implementation of signed operations to overcome this drawback.

4.4.3. Technology Library

The technology library contains logical descriptions of basic units and their parameters like delay and area, thus guiding the mapping of the logic to the circuit netlist. Since the GC protocol only involves 2-input gates, so does our custom technology library. Besides, the area of XOR and non-XOR gates are set to 0 and 1, respectively. By using this area setting and choosing minimization as the only optimization tool, the synthesis tool produces netlist with minimum possible number of non-XOR gates[6].

Some functions which are identical in GC protocol may represent by different number of gates using only standard gates. Songhori *et al.* have proposed four non-standard gates to increase the flexibility of mapping process [6]. We include those gates in our library as well. The library we include for circuit synthesis can be found at the TinyGarble repository on Github.

4.4.4. Sequential Customized Bozorth Matcher

A sequential circuit can be decomposed to a combinational circuit and a set of registers that hold the intermediate values. To garble the sequential circuits, the combinational part is garbled/evaluated in each sequential cycle, while the garbling keys are stored in the registers, which are used as inputs in the next cycle [20].

Figure 4-2 presents the complete circuit design, which is composed of 3 parts, a comparison module, a transformation module and a memory module. The comparison module determines if two minutia-pairs, one from the client and one from the server is compatible (shown in the up left). Once a compatible pair is found, it goes to the transformation module for further computation. The transformation module contains three sub modules: one to compute the difference of the global rotation of two pairs, another to encrypt the starting point of a minutia-pair from the client's input, and the third one to determine the starting point of the corresponding minutia-pair from the server's input (shown in the middle). The memory stores all compatible pairs (shown in the bottom right). Notice that the memory module is implemented using MUX and arrays of FFs, consistent with TinyGarble's sequential methodology [20].

Figure 4-2 Customized Bozorth matcher architecture

4.5. Evaluation

We evaluate our privacy-preserving fingerprint authentication system by testing the reliability of the core matching algorithm and assessing the timing performance of circuit execution.

4.5.1. Reliability

Our customized Bozorth matcher has been implemented in C so as to determine its reliability and accuracy. We use a standard fingerprint database FVC2002 DB2, which contains 10 fingerprint images of 8 distinct objects (80 in total) [34]. The images are of the size 296×560 and collected by using an optical

sensor. The minutiae files are created by using the minutiae detector MINDTCT, which is developed by NIST.

It is obvious that the number of features involved in the algorithm evaluation determines the detection rate. In our case, the number of features to be evaluated is mainly limited by the lengths of comparison and compatibility tables. Therefore, we have tested our customized Bozorth algorithm with different table lengths. For simplicity, we set the length of comparison table (denoted as l) and the length of compatibility table equal with each other. We test our matching algorithm on FVC2002 DB2 and report GAR and FAR for different values of l (shown in Table 4.5-1). We have also tested the original Bozorth algorithm on the same database, showing a 98.5% GAR with FAR at 0.1%. Noticing that the original Bozorth algorithm does not limit the comparison/compatibility table length. As expected, by sacrificing a little accuracy, we achieve efficiency and practicability.

FAR /	64	72	80	88	96	104	112	120	128
0%	74.2%	75.9%	81.8%	83.2%	84.7%	87.3%	87.3%	87.5%	90.2%
0.1%	84.2%	85.5%	86.3%	87.3%	88.8%	90.4%	91.0%	91.6%	92.0%
1.0%	89.4%	89.8%	90.0%	91.9%	93.4%	95.5%	95.7%	96.1%	96.5%

Table 4.5-1 GAR vs FAR for Different Table Length

4.5.2. Timing Performance

We also implement the core part of our matching algorithm in Verilog and generate circuits optimized for GC protocol using customized synthesis flow. The circuit generation is done by Synopsis Design Compiler 2010.03-SP4 on a system with Linux RedHat Server 5.6, 8 GB of memory, and Intel Xeon X5450 CPU at 3 GHz. For garbling and evaluation, we utilize JustGarble framework with modifications to support sequential circuits [20], [26]. The metrics used for the evaluation of our implementations are consistent with Songhori *et al.*'s work [6]. They are stated as follows:

1. The circuit size (CS) is calculated according to:

$$CS = 24 \times q,$$

where q is the total number of gates. For each gate, two indices(16B) and one type(8B) are stored in circuit description file.

2. The garbling/evaluating time is calculated as:

$$T = \# \text{ of non-XOR} \times T_{\text{non-XOR}} + \# \text{ of XOR} \times T_{\text{XOR}},$$

where $T_{\text{non-XOR}}$ is the execution time of a non-XOR gate and T_{XOR} is the execution time of an XOR gate, which are 164 clock cycles(cc) and 62 cc respectively measured on a conventional PC (Intel Core i7-2600 at 3.4 GHz) with Ubuntu 14.10 Desktop [6].

3. The total garbling time is calculated as:

$$T_{\text{total}} = \# \text{ of rounds} \times T,$$

where T is computed according to equation (2), and the number of rounds is determined as the multiplication of comparison table length from two parities.

Each minutia-pair is encoded as a 52-bit vector (12 bits for relative distance d_{ij} , 9 bits for relative angle β_1 , 9 bits for relative angle β_2 , 6 bits for index i , 6 bits for index j , and 10 bits for the global orientation θ_{ij}).

l	64	72	80	88	96	104	112	120	128
Total gates	5175	6406	7041	7684	8320	8963	9607	10281	10886
Non-XOR	2055	2319	2541	2767	2982	3190	3410	3678	3880
CS(KB)	134	150	165	180	195	210	225	241	255
$T(cc) \times 10^9$	2.24	3.61	5.53	6.73	7.95	9.18	10.09	10.85	11.57

Table 4.5-2 Circuit Size and Timing Evaluation

The circuit size and timing performance for different values of l are shown in Table 4.5-2. As can be seen, the circuit size grows linearly with minutia comparison/compatibility table length, since memory elements dominate in circuit

architecture. The largest CS achieved in our work is 255 KB ($l = 128$), which can easily fit in an embedded processor.

A highly efficient and compact implementation is achieved if setting l to be 64. The total time for evaluating the garbled algorithm is 2.24×10^9 clock cycles (0.67s) in our setting. The detection rate found in Table 4.5-2 shows a GAR at 84.2% with FAR at 0.1%, where FAR = 0.1% is a typical setting for real fingerprint authentication/identification systems. The detection rate is satisfying for applications with privacy guaranteed.

Figure 4-3 GAR and T v.s. Table Length

As shown in Figure4-3, we also emphasize that increasing circuit size (as well as T_{total}) does not have a notable impact on improving detection rate when $l > 104$. Therefore, $l = 104$ is recognized as the best point that balances the detection rate and circuit size (efficiency as well). The optimized implementation achieved in our work requires 210 KB for storage and 9.18×10^9 clock cycles for circuit garbling. Tested on FVC2002 DB2, GAR is reported to be 90.4% with FAR at 0.1%.

The offline pre-computation phase of the protocol execution is simulated on a conventional laptop with Intel core 5 at 2.6GHz. The offline computation on the client side includes the generation of a minutia-pair comparison table and a loop through the compatibility table. It runs 20 milliseconds per match on our specified system. On the server side, it requires 18 milliseconds to compute the minutia-pair comparison table. Notice that the server is able to provide a one-time generation of all comparison tables unless new entries registered or old data deleted. If performing fingerprint identification(1-to-n), it takes $18 \times n$ milliseconds in total to construct minutia-pair comparison tables for the entire fingerprint database. Compared with circuit evaluation stage, the timing for offline computation is trivial.

Conclusion

This thesis presents the first compact, efficient and reliable method for privacy-preserving fingerprint authentication based on minutiae representation. We construct the core fingerprint authentication algorithm by customizing a widely used matching algorithm so as to achieve an efficient implementation. In contrast to earlier suggested methods that were impractical and unscalable, we suggest the first sequential description of a fingerprint matching algorithm, which dramatically reduces the circuit size leveraging the TinyGarble methodology. The implementation of GC based fingerprint authentication is done by Synopsis Design Compiler. The optimized implementation achieved in our work requires 210 KB for storage and 9.18×10^9 clock cycles for circuit evaluation. In addition, it has a GAR as high as 90.4% with FAR at 0.1% when tested on a standard fingerprint database. The accuracy numbers are similar to the reported performance of the Bozorth algorithm on plaintext fingerprint data. Our methodology is very promising and practical for real world applications. Moreover, it can be extended to other biometric based applications, such as iris recognition, which we will be working on in the future.

References

- [1] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 14, no. 1, pp. 4–20, 2004.
- [2] D. M. Tumey, T. Xu, and C. Arndt, *Fingerprint verification system utilizing a facial image-based heuristic search method*. Google Patents, 2005.
- [3] Y. Moon, H. Ho, K. Ng, S. Wan, and S. Wong, "Collaborative fingerprint authentication by smart card and a trusted host," in *Electrical and Computer Engineering, 2000 Canadian Conference on*, 2000, vol. 1, pp. 108–112.
- [4] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. Donida Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and others, "Privacy-preserving fingerprint authentication," in *Proceedings of the 12th ACM workshop on Multimedia and security*, 2010, pp. 231–240.
- [5] M. Blanton and P. Gasti, "Secure and Efficient Iris and Fingerprint Identification," in *Biometric Security*, Cambridge Scholars Publishing, 2015.
- [6] E. M. Songhori, S. U. Hussain, A.-R. Sadeghi, and F. Koushanfar, "Compacting Privacy-Preserving k-Nearest Neighbor Search using Logic Synthesis," 2015.
- [7] R. Cramer, I. Damgård, and J. B. Nielsen, *Multiparty computation from threshold homomorphic encryption*. Springer, 2001.
- [8] T. Rabin and M. Ben-Or, "Verifiable secret sharing and multiparty protocols with honest majority," in *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, 1989, pp. 73–85.
- [9] W. Henecka, A.-R. Sadeghi, T. Schneider, I. Wehrenberg, and others, "TASTY: tool for automating secure two-party computations," in *Proceedings of the 17th ACM conference on Computer and communications security*, 2010, pp. 451–462.
- [10] D. Malkhi, N. Nisan, B. Pinkas, Y. Sella, and others, "Fairplay-Secure Two-Party Computation System,," in *USENIX Security Symposium*, 2004, vol. 4.
- [11] I. Damgård, M. Geisler, M. Krøigaard, and J. B. Nielsen, "Asynchronous multiparty computation: Theory and implementation," in *Public Key Cryptography-PKC 2009*, Springer, 2009, pp. 160–179.
- [12] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Privacy Enhancing Technologies*, 2009, pp. 235–253.
- [13] A.-R. Sadeghi, T. Schneider, and I. Wehrenberg, "Efficient privacy-preserving face recognition," in *Information, Security and Cryptology-ICISC 2009*, Springer, 2010, pp. 229–244.
- [14] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, "Scifi-a system for secure face identification," in *Security and Privacy (SP), 2010 IEEE Symposium on*, 2010, pp. 239–254.
- [15] D. Evans, Y. Huang, J. Katz, and L. Malka, "Efficient privacy-preserving biometric identification," in *Proceedings of the 17th conference Network and Distributed System Security Symposium, NDSS*, 2011.

- [16] W. A. Alberto Torres, N. Bhattacharjee, B. Srinivasan, and I. Khalil, "Privacy-preserving biometrics authentication systems using fully homomorphic encryption," *International Journal of Pervasive Computing and Communications*, vol. 11, no. 2, 2015.
- [17] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti, "Filterbank-based fingerprint matching," *Image Processing, IEEE Transactions on*, vol. 9, no. 5, pp. 846–859, 2000.
- [18] A. Holzer, M. Franz, S. Katzenbeisser, and H. Veith, "Secure two-party computations in ANSI C," in *Proceedings of the 2012 ACM conference on Computer and communications security*, 2012, pp. 772–783.
- [19] B. Kreuter, A. Shelat, B. Mood, and K. R. Butler, "PCF: A Portable Circuit Format for Scalable Two-Party Secure Computation," in *Usenix Security*, 2013, vol. 13, pp. 321–336.
- [20] E. M. Songhori, S. U. Hussain, A.-R. Sadeghi, T. Schneider, and F. Koushanfar, "TinyGarble: Highly Compressed and Scalable Sequential Garbled Circuits," in *IEEE S & P*, 2015.
- [21] A. Yao, "How to generate and exchange secrets," in *Foundations of Computer Science, 1986., 27th Annual Symposium on*, 1986, pp. 162–167.
- [22] M. Naor and B. Pinkas, "Computationally secure oblivious transfer," *Journal of Cryptology*, vol. 18, no. 1, pp. 1–35, 2005.
- [23] 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.
- [24] V. Kolesnikov and T. Schneider, "Improved garbled circuit: Free XOR gates and applications," in *Automata, Languages and Programming*, Springer, 2008, pp. 486–498.
- [25] M. Naor, B. Pinkas, and R. Sumner, "Privacy preserving auctions and mechanism design," in *Proceedings of the 1st ACM conference on Electronic commerce*, 1999, pp. 129–139.
- [26] M. Bellare, V. T. Hoang, S. Keelveedhi, and P. Rogaway, "Efficient garbling from a fixed-key blockcipher," in *Security and Privacy (SP), 2013 IEEE Symposium on*, 2013, pp. 478–492.
- [27] A. K. Jain, J. Feng, and K. Nandakumar, "Fingerprint matching," *Computer*, no. 2, pp. 36–44, 2010.
- [28] C. I. Watson, M. D. Garriss, E. Tabassi, C. L. Wilson, R. M. McCabe, and S. Janet, "User's guide to NIST fingerprint image software 2 (NFIS2)," *National Institute of Standards and Technology*, 2004.
- [29] D. Lee, K. Choi, and J. Kim, "A robust fingerprint matching algorithm using local alignment," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, 2002, vol. 3, pp. 803–806.
- [30] X. Luo, J. Tian, and Y. Wu, "A minutiae matching algorithm in fingerprint verification," in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, 2000, vol. 4, pp. 833–836.

- [31] A. Jain, L. Hong, and R. Bolle, "On-line fingerprint verification," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 4, pp. 302–314, 1997.
- [32] X. Jiang and W.-Y. Yau, "Fingerprint minutiae matching based on the local and global structures," in *Pattern recognition, 2000. Proceedings. 15th international conference on*, 2000, vol. 2, pp. 1038–1041.
- [33] J. Feng, "Combining minutiae descriptors for fingerprint matching," *Pattern Recognition*, vol. 41, no. 1, pp. 342–352, 2008.
- [34] B. Bhanu and X. Tan, "Fingerprint indexing based on novel features of minutiae triplets," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 5, pp. 616–622, 2003.
- [35] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain, "FVC2002: Second fingerprint verification competition," in *Pattern recognition, 2002. Proceedings. 16th international conference on*, 2002, vol. 3, pp. 811–814.