Robust Privacy-Preserving Fingerprint Authentication

Ye Zhang and Farinaz Koushanfar
Dept. of ECE, Rice University
Houston, Texas, USA
{yz61, farinaz}@rice.edu

***Abstract*—This paper presents the first scalable, efficient, and reliable privacy-preserving fingerprint authentication based on minutiae representation. Our method is provably secure by leveraging the Yao's classic 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 and customized to GC using the TinyGarble framework. We show evaluations of our modified matching algorithm on a standard fingerprint database FVC2002 DB2 to demonstrate its reliability. The implementation of privacy-preserving fingerprint authentication using Synopsis Design Compiler on a commercial Intel processor shows the efficiency and scalability of the proposed methodologies.**

***Index Terms*—Secure Function Evaluation, Secure Multiparty Computation, Fingerprint Authentication, Garbled Circuit**

I. 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, especially when the matching process is performed in partially untrusted environments. Therefore, 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 paper, 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 liveliness 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 sides private data during protocol execution.

This paper suggests customizing a highly reliable alignment-free fingerprint matching algorithm as our core fingerprint authentication engine. As our modified algorithm evaluates larger number of inputs, 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:

1) Introduction of the first scalable, efficient and reliable methodology for privacy-preserving fingerprint authentication based on minutiae representation.
2) Development of an efficient and reliable minutiae-based fingerprint authentication algorithm by customizing a widely used fingerprint matching algorithm – Bozorth matcher.
3) 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.
4) 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.

II. Prior Work

Several earlier works focus on the area of secure multi-party computation, ranging from the optimization of related cryptographic tools to customized secure protocol generation [7], [8], [9], [10]. The existence of these tools has motivated new approaches for secure biometric data processing, e.g., [11], [12], [4], [5], [13]. The available literature has mostly relied on HE which encrypts the data in a way that applying mathematical function on the encrypted message and then decrypts it, generates the same results as preforming the operations on the plain message. For example, Barni *et al.* report a secure protocol of FingerCode based authentication [14] by leveraging HE [4]. The matching proceeds by computing the Euclidean distance of two FingerCode vectors. Even with protocol-level improvements, their implementation is still impractical for real applications due to the lack of efficiency using HE. By cooperating GC with HE, Blanton *et al.* present a privacy preserving protocol for FingerCode based identification with notable reduction in the overhead [5]. 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 minutiae is moved to a new location far from its original position will not be matched properly.

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[10], [15], [16]. 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 [17]. A great improvement in circuit compactness and scalability is achieved by using a sequential circuit description for GC and TinyGarble optimizations.

III. Preliminaries

We use garbled circuits and oblivious transfer as our primary cryptographic tools. These techniques are briefly reviewed here.

A. 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 [18].

B. Yao's GC Protocol

Yao's Garbled circuits is a protocol that enables secure two-party computation [19]. Specifically, two semi-honest parties, Alice and Bob, want to jointly compute a function f on their secret inputs. The function needs to be represented as a combinational Boolean circuit. Alice, the garbler, maps the plain values of each wire to random symmetric keys and generates an encrypted truth table for each gate according to

the possible combinations of the input keys. Alice then sends to Bob the encrypted truth tables and keys corresponding to her input values. Bob, the evaluator, obliviously receives the keys corresponding to his inputs through OT. Bob evaluates the circuit gate by gate using these keys and encrypted truth tables until he evaluates all gates. Finally, Alice provides a mapping from the encrypted output to plain output.

IV. Bozorth algorithm and modifications

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 [20], [21], [22]. 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, National Institute of Standards and Technology (NIST) has developed a rotation and translation invariant fingerprint matching algorithm, called Bozorth matcher.

The official description of Bozorth matcher is given in [23]. 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:

1) Constructing intra-fingerprint minutia-pair(comparison) tables.
2) 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.
3) Traversing the inter-fingerprint compatibility table to build a web/cluster and accumulate a match score.

A. 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 1, 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 $\{\Delta(\theta_{ij,c}, \theta_{ij,s}), i_c, j_c, i_s, j_s\}$, where $\Delta(\theta_{ij,c}, \theta_{ij,s})$ 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.

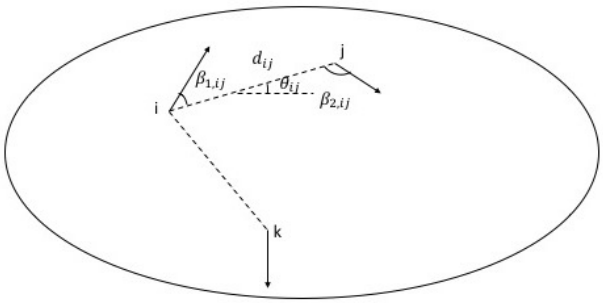

Fig. 1: Minutia-pair and minutia-triplet.

B. Our Adaptations

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 1, we define a 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, $(\Delta(\Delta(\theta_{ij,c}, \theta_{ij,s})) < t)$. 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 incomplete compatibility table. In the following section, we propose privacy-preserving protocol for performing fingerprint authentication based on the customized Bozorth algorithm.

V. Privacy-preserving Fingerprint Authentication

In this section, we introduce a secure realization for a 1-to-1 fingerprint matching algorithm. In order to achieve an

Protocol1 Privacy-preserving minutiae-based fingerprint authentication

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

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

3: **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 OT and computes a garbled incomplete compatibility table, where the second column(the index of the starting point of a minutia-pair derived from C) is encrypted by first applying a linear transformation $(ax + b)$, then XOR the transformed index with a private key K generated by the server. The encrypted incomplete compatibility table is released to C.
3) C decrypts the encrypted incomplete 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.

efficient implementation, we have designed a pre-computation stage for both parties computing their minutia-pair comparison tables, and an offline computation stage on the client side to perform the final stage of our matching algorithm, thus greatly reducing the amount of computation enrolled in GC. Besides, for security consideration, the intermediate results are encrypted by server before sending to client.

A. 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 minutia-pair comparison 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 $Y = \{(x_1', y_1', \alpha_1'), ..., (x_n', y_n', \alpha_n')\}$ and a private key 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 $X = \{(x_1, y_1, \alpha_1), ..., (x_m, y_m, \alpha_m)\}$ 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 Protocol1.

B. Security Argument

In this section, we briefly present a security argument for our scheme, where 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 file 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 OT, 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 GC 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 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 linear transformation during GC evaluation. Thus, owning the encrypted index, the client is unable to locate the corresponding minutia-pair in its own minutia-pair comparison table, not to mention inferring the server's minutiae information. We thereby demonstrate the partially encrypted incomplete compatibility table is of no use in recovering the other party's minutiae information.

VI. Implementation

In this section, we present the implementation of the sensitive parts (compatibility table construction) of our customized Bozorth matcher based on Yao's GC protocol. 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 [17].

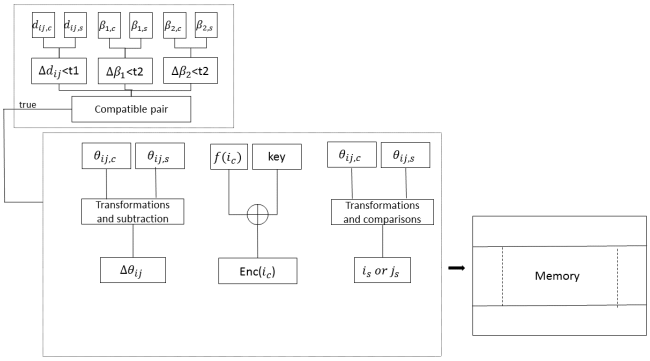

Fig. 2: Customized Bozorth matcher architecture.

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 [17].

Figure 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 [17].

VII. 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.

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). The images are of the size 296x560 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 clear that the number of features involved in the algorithm evaluation determines the detection rate. In our case, the number of minutia-pairs 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 I). 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 TinyGarble framework [17]. The metrics are stated as follows[6]:

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

$$CS = 24 \times q, \tag{1}$$

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:

$$T = \#\text{ of non-XOR} \times T_{\text{non-XOR}} + \#\text{ of XOR} \times T_{\text{XOR}}, \tag{2}$$

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:

$$T_{total} = \#\text{ of rounds} \times T, \tag{3}$$

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.

The circuit size and timing performance for different values of l are shown in Table II. 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 I shows a GAR at 84.2% with FAR at 0.1%.

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%, where FAR = 0.1% is a typical setting for real fingerprint authentication/identification systems.

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 both sides takes only

FAR \ l	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 I: GAR vs FAR for different number of minutia-pairs

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 II: circuit size and timing evaluation for customized Bozorth matcher

around 20 milliseconds per match. Compared with circuit evaluation stage, the timing for offline computation is trivial.

VIII. Conclusion

This paper 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.

Acknowledgment

This work is partially supported by an Office of Naval Research grant (ONR-R17460), a National Science Foundation grant (CNS-1059416), and a Multidisciplinary University Research Initiative grant (FA9550-14-1-0351/Rice 14-0538) to the ACES lab at Rice University.

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," Nov. 8 2005, uS Patent 6,963,659.

[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*, vol. 1. IEEE, 2000, 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 *et al.*, "Privacy-preserving fingercode authentication," in *Proceedings of the 12th ACM workshop on Multimedia and security*. ACM, 2010, pp. 231–240.

[5] M. Blanton and P. Gasti, *Secure and Efficient Iris and Fingerprint Identification*. Cambridge Scholars Publishing, 1 2015, ch. 9.

[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*. ACM, 1989, pp. 73–85.

[9] D. Malkhi, N. Nisan, B. Pinkas, Y. Sella *et al.*, "Fairplay-secure two-party computation system." in *USENIX Security Symposium*, vol. 4. San Diego, CA, USA, 2004.

[10] W. Henecka, A.-R. Sadeghi, T. Schneider, I. Wehrenberg *et al.*, "Tasty: tool for automating secure two-party computations," in *Proceedings of the 17th ACM conference on Computer and communications security*. ACM, 2010, pp. 451–462.

[11] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Privacy Enhancing Technologies*. Springer, 2009, pp. 235–253.

[12] 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.

[13] 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.

[14] 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.

[15] 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*. ACM, 2012, pp. 772–783.

[16] B. Kreuter, A. Shelat, B. Mood, and K. R. Butler, "Pcf: A portable circuit format for scalable two-party secure computation." in *Usenix Security*, vol. 13, 2013, pp. 321–336.

[17] 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.

[18] M. Naor and B. Pinkas, "Computationally secure oblivious transfer," *Journal of Cryptology*, vol. 18, no. 1, pp. 1–35, 2005.

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

[20] D. Lee, K. Choi, and J. Kim, "A robust fingerprint matching algorithm using local alignment," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 3. IEEE, 2002, pp. 803–806.

[21] X. Luo, J. Tian, and Y. Wu, "A minutiae matching algorithm in fingerprint verification," in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, vol. 4. IEEE, 2000, pp. 833–836.

[22] 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.

[23] C. I. Watson, M. D. Garris, E. Tabassi, C. L. Wilson, R. M. McCabe, and S. Janet, "Users guide to nist fingerprint image software 2 (nfis2)," *National Institute of Standards and Technology*, 2004.