Poster: Secure Computation of Fingerprint Alignment and Matching

Fattaneh Bayatbabolghani and Marina Blanton
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN, USA
Email: {fbayatba,mblanton}@nd.edu

***Abstract*—Secure computation with biometric data is a challenging topic and attention to this area has grown during recent years. This is because biometric data is highly sensitive and requires protection when used in a variety of applications including biometric verification and identification. Fingerprint images are one of the most accurate type of biometry used in these applications and they are highly sensitive due to their ability to uniquely identify the data owner. For that reason, when fingerprint data is used in applications, it is desired to properly protect such data. Typically the computation involves comparing two fingerprint representations in different settings ranging from comparing two databases belonging to different entities to searching for a fingerprint in an external database or outsourcing fingerprint comparisons for computational reasons. In this work, we put forward three secure protocols for fingerprint alignment and matching based on the most precise or efficient algorithms in the biometric literature. To the best of our knowledge, this is the first time fingerprint alignment based on minutia points is considered in a secure computation framework due to the complexity of the algorithms. After designing the necessary building blocks, we build the overall protocols and realize them in the two-party setting using garbled circuit evaluation and in the multi-party setting using secret sharing techniques.**

***Keywords*—Fingerprint alignment, fingerprint matching, secure computation, garbled circuit evaluation, secret sharing.**

I. Introduction

The motivation of this work comes from the need to protect sensitive biometric data when it is used in applications. In a biometric recognition process, biometric modalities such as iris, voice, fingerprint, and DNA are compared for the purposes of verification or identification. One common type of biometric data is fingerprints, which is both accurate and requires protection because it can reveal the identity of its owner. In the most accurate types of fingerprint recognition (typically based on minutia points), there are two main steps: alignment and matching. The purpose of the alignment step is to improve the accuracy of the matching step. To the best of our knowledge, all publications on secure fingerprint recognition focused on the matching step because of the high complexity of the alignment. Therefore, we aim to design an efficient solution to the entire recognition process including both the alignment and matching steps to achieve more reliable results.

In this work, we focus on three algorithms for fingerprint comparisons that include both alignment and matching. The algorithms were selected due to their precision or speed with the goal to provide as efficient security solutions as possible. To be able to create such solution, we first needed to design new secure sub-protocols for computing sine, cosine, and the square root for fixed point representation (suitable for both two-party and multi-party computation). Once we obtain the building blocks and compose them to minimize the overall overhead, we implement the solutions using garbled circuit evaluation techniques in the two-party setting and linear secret sharing in the multi-party setting.

II. Fingerprint Recognition

Fingerprint is an exterior surface of a finger. All features extracted from a fingerprint image result in a unique representation. One of the features most commonly used for fingerprint recognition is minutia points. They are related to fingerprint ridge endings and are typically represented by the following elements: 1) an x-coordinate, 2) a y-coordinate, 3) orientation in radians denoted by θ, and 4) an optional minutia type. Sometimes the relationship of each minutia to a reference point is computed, which is typically the core point representing the center area of a fingerprint.

The selected algorithms take as the input two fingerprints S and T, each represented as a set of minutia points or helper data: a sample set $S = \{s_1, s_2, \ldots, s_n\}$ and a template set $T = \{t_1, t_2, \ldots, t_m\}$, after which they proceeded with alignment and matching. We next briefly introduce the three selected fingerprint recognition algorithms.

A. Algorithm 1: Fingerprint Matching Based on Minutiae Alignment

The first algorithm [3] uses minutia representation of fingerprints S and T and considers all possible pairs (s_i, t_j) as a reference pair based on which the fingerprints are aligned. Each reference pair is considered as a matched pair and then all other points of S are transformed using that pair as part of the alignment process. The algorithm then counts the number matched minutia pairs from T and transformed S to compute the matching score of the reference pair. After trying all possible reference pairs, the algorithm chooses an alignment that maximizes the matching score of the fingerprints and thus increases the likelihood of two fingerprints to be matched. The details of the alignment step are provided in [3] and the matching step in [1].

B. Algorithm 2: Fingerprint Matching Based on Trimmed Iterative Closest Point Algorithm

The second algorithm is based on trimmed iterative closest point (TICP) with its detailed available from [5]. In that

algorithm, the alignment step uses TICP [2] and is explicitly different from the matching step. The algorithm uses helper data representation of fingerprints S and T as described earlier and returns the computed alignment and the corresponding matching score.

C. Algorithm 3: Fingerprint Matching Based on Spectral Minutiae Representation

The last algorithm [4] uses minutia-based representation of fingerprints, but offers greater efficiency than other algorithms that use the same representation because the representation is modified using Fourier transform. Also, feature reduction approaches from [4] are applied to a spectral minutia feature vector to reduce the feature size without losing precision. After feature reduction, S and T are presented as real-valued feature vectors. Then what remains is to compute a two-dimensional correlation coefficient between S and T to determine the similarity score of S and T. Based on the properties of spectral minutia representation, rotation of the image for the purpose of its alignment corresponds to a circular shift of the vector. After performing each rotation, the algorithm computes the corresponding similarity score. The algorithm returns the highest similarity score across all rotations, which corresponds to the best alignment of the two fingerprints and its value shows how similar the fingerprints are.

III. Problem Setup and Security Model

Because of the variety of settings in which fingerprint recognition may be used, we distinguish between different computation setups and the corresponding security settings.

1) There will be circumstances when two entities would like to compare fingerprints they respectively possess, without revealing any information about their data to the other party. This can correspond to the cases when both entities own a fingerprint database and would like to find entries common to both of them or when one entities would like to search a database belonging to a different entity for a specific fingerprint. In these settings, the computation takes the form of secure two-party computation, where the participants can be regarded as semi-honest or fully malicious depending on the application and their trust level.
2) There will also be circumstances when one or more the data owners are computationally limited and would like to securely offload their work to more powerful servers or cloud providers (this applies even if all fingerprints used in comparisons belong to a single entity). Then multi-party settings that allow for input providers to be different from the computational parties apply. In such settings, the computational parties are typically semi-honest, but stronger security models can also be used for added security guarantees.

IV. Building Secure Solutions

To be able to build secure protocols for different fingerprint recognition algorithms, we first need to determine what operations and arithmetic type are used in the algorithms and what is required to implement those components.

Some of the operations used in the computation are elementary and well-studied in the security literature. They, for example, include addition/subtraction, multiplication, comparisons (including equality), and prefix multiplication. Others are more complex (such as division), but still have presence in prior work. What, however, is more interesting is that there are also a number of rather complex operations, which have not been studied before in the context of secure two-party or multi-party computation. For the purposes of this work, we are to build efficient implementation of the following functions using fixed point representation of real numbers. Here notation $[a]$ denotes that the value of a is protected and not known throughout the computation.

- $[y] \leftarrow \mathsf{Sin}([x])$ computes sine $[\mathsf{Sin}(x)]$ and stores the result as $[y]$.
- $[y] \leftarrow \mathsf{Cos}([x])$ computes cosine $[\mathsf{Cos}(x)]$ and stores the result as $[y]$.
- $[y] \leftarrow \mathsf{Sqrt}([x])$ computes the square root of $[x]$ and stores the result as $[y]$.

We begin with exploring the available algorithms for computing these functions (such as Taylor series for sine and cosine) and build protocols that provide the best performance in each of the chosen settings of two-party and multi-party computation. Our optimizations for designing the building blocks as well as the overall protocols focus on the specific costs of the underlying techniques for secure arithmetic and ensure that we achieve efficiency together with precision and provable security guarantees.

V. Performance Evaluation

To evaluate performance of our protocols, we will provide experimental results for both settings described in section III. The implementation will also provide insights on the relative performance gains between the settings for this application. In the two-party case, our implementation is based on garbled circuit evaluation. In the multi-party settings, we will utilize secure computation techniques based on linear threshold secret sharing (such as Shamir secret sharing).

References

[1] M. Blanton and P. Gasti. *Secure and Efficient Iris and Fingerprint Identification*. Chapter 9 in Biometric Security, D. Ngo, A. Teoh, and J. Hu (Editors), Cambridge Scholars Publishing, 2015.

[2] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek. The trimmed iterative closest point algorithm. In *Pattern Recognition*, pages 545–548, 2002.

[3] T. T. Chu and C. T. Chiu. Fingerprint recognition based on minutiae alignment. In *IPPR conference on computer vision, graphics and image processing*, 2014.

[4] Xu. Haiyun, R. N. J. Veldhuis, T. A. M. Kevenaar, and T. A. H. M. Akkermans. A fast minutiae-based fingerprint recognition system. *IEEE Systems journal*, 3(4):418–427, 2009.

[5] K. Nandakumar, A. K. Jain, and S. Pankanti. Fingerprint-based fuzzy vault: Implementation and performance. *IEEE Transactions on Information Forensics and Security*, 2(4):744–757, 2007.