Outsourceable Two-Party Privacy-Preserving Biometric Authentication

Hu Chun
Dept. of Computer Science
Missouri S&T
Rolla, Missouri 65409
chwrc@mst.edu

Yousef Elmehdwi
Dept. of Computer Science
Missouri S&T
Rolla, Missouri 65409
ymez76@mst.edu

Feng Li
Dept. of Computer Science
Missouri S&T
Rolla, Missouri 65409
lftrd@mst.edu

Prabir Bhattacharya
School of Computing Sciences
and Informatics
University of Cincinnati
Cincinnati, Ohio 45221
bhattapr@ucmail.uc.edu

Wei Jiang
Dept. of Computer Science
Missouri S&T
Rolla, Missouri 65409
wjiang@mst.edu

ABSTRACT

Biometric authentication, a key component for many secure protocols and applications, is a process of authenticating a user by matching her biometric data against a biometric database stored at a server managed by an entity. If there is a match, the user can log into her account or obtain the services provided by the entity. Privacy-preserving biometric authentication (PPBA) considers a situation where the biometric data are kept private during the authentication process. That is the user's biometric data record is never disclosed to the entity, and the data stored in the entity's biometric database are never disclosed to the user. Due to the reduction in operational costs and high computing power, it is beneficial for an entity to outsource not only its data but also computations such as biometric authentication process to a cloud. However, due to well-documented security risks faced by a cloud, sensitive data like biometrics should be encrypted first and then outsourced to the cloud. When the biometric data are encrypted and cannot be decrypted by the cloud, the existing PPBA protocols are not applicable. Therefore, in this paper, we propose a two-party PPBA protocol when the biometric data in consideration are fully encrypted and outsourced to a cloud. In the proposed protocol, the security of the biometric data is completely protected since the encrypted biometric data are never decrypted during the authentication process. In addition, we formally analyze the security of the proposed protocol and provide extensive empirical results to show its runtime complexity.

Categories and Subject Descriptors

K.6 [**Management of Computing and Information Systems**]; K.6.5 [**Security and Protection**]: Authentication

General Terms

Security

Keywords

Security; Cloud Computing; Biometric Authentication

1. INTRODUCTION

In recent years, biometric authentication has increasingly gained importance for various application domains. Those systems which include fingerprint-, face- and iris- authentication systems are widely used in enterprise, civilian and law enforcement. Such systems typically consist of an entity whose databases hold biometric records and users/clients who send the entity their biometric recordings for authentication [15]. During the process of biometric authentication, the systems need to identify whether candidate biometric readings from users match the records in the entity's biometric database. There are potential privacy concerns regarding biometric authentication: biometric matching process would cause a leakage of user's biometric data especially running on untrusted servers. Biometric data are usually very sensitive because they could uniquely identify a person. Thus, the leakage of biometric data to malicious parties can lead to a violation of personal privacy. To avoid this privacy concern, privacy-preserving biometric authentication (PPBA) protocols have been developed, e.g., [5, 6, 13, 15, 43]. Under the existing PPBA protocols, a user's biometric data record is never disclosed to the entity, and the entity's biometric database is never exposed to the user.

Due to cost efficiency and operational flexibility of the cloud computing paradigm [1, 8, 30], an entity has the opportunity to outsource its data and their relevant computations to a cloud which can provide on-demand services. To outsource biometric authentication tasks, the entity's biometric database needs to be encrypted before outsourced to a cloud.

This is because it is difficult for the cloud to actually guarantee the confidentiality of sensitive data. The challenge arises due to several well-documented security risks faced by existing cloud service providers [14,22,40,41,48,52]. One such risk for instance, is that when a breach occurs in the cloud, any sensitive data stored in the clear (i.e., not encrypted) can be easily exposed to the attacker.

As mentioned previously, the goal of PPBA protocols is to perform biometric authentication without disclosing the involved biometric data to the participating parties, except for their own data. The input data to the existing PPBA protocols are not encrypted, and it is not straightforward to modify these protocols to solve the outsourced biometric authentication problem because the cloud cannot access the original/non-encrypted biometric database. Different from existing PPBA, an entity's biometric database is encrypted under an outsourceable privacy-preserving biometric authentication (O-PPBA) protocol.

1.1 Problem Definition

Let $D = \{v_1, \ldots, v_n\}$ denote an entity's biometric database with n biometric data records. Each v_i, $1 \leq i \leq n$ is a m-dimension vector representation of a biometric data record. Let $[D] = \{[v_1], \ldots, [v_n]\}$ denote the encryption of D where each $[v_i]$ is encrypted component-wise. Suppose Bob is a user who wants to be authenticated using his biometric image data denoted by u. Here u is represented using the same method as the records in D. An O-PPBA protocol can be defined as follows with $[D]$ and [u] as the inputs:

$$\text{O-PPBA}([D], [u], t) \to b \tag{1}$$

When $b = 1$, there is a biometric data record v_i in D such that the distance between u and v_i is at most t distance apart; otherwise, $b = 0$. t is defined by the underlying biometric authentication system, and it varies from system to system. t also depends on the biometric data used for authentication. How to determine the best value for t is out of the scope of this paper, so we just assume t is publicly know parameter. During an execution of O-PPBA, $[D]$ and $[u]$ are never decrypted to maximize the confidentiality protection of D and u.

We inherit the common structure of the existing PPBA systems which mainly consist of two phases: distance computation, and matching and retrieval [15]. In the distance computation phase, either Euclidean or Hamming distance between the feature vectors of biometric records v_i's in D and the user's biometric data record u are calculated pairwise. After that in matching and retrieval phase, those distances are compared with a pre-defined threshold t to decide whether u matches some v_i with t-distance apart. Some biometric authentication protocols such as face recognition [13] need a feature extraction phase to get the biometric feature vectors; however, in this paper, we assume that feature extraction phase as a pre-computation stage to produce the feature vectors in D and u.

1.2 Our Contributions

In this paper, we propose fully secure O-PPBA protocol. Our protocol meets the following privacy requirements:

- Contents of D or any intermediate results should not be revealed to the cloud.
- Bob's biometric image data u should not be revealed to the cloud.
- The output b should not be revealed to the cloud.
- Data access patterns, such as the biometric data records corresponding to the matching or comparison results with u, should not be revealed to Bob and the cloud.
- After sending his encrypted biometric data to the cloud, Bob never participates in any intermediate computation. Therefore, our protocols incur low computation overhead on Bob.

We emphasize that the intermediate results seen by the cloud in our protocol are either newly generated randomized encryptions or random numbers.

1.3 Overview of the Proposed Protocol

Let P_1 denote a cloud service provider who stores $[D]$ and performs biometric authentication based on $[D]$, $[u]$ and t. When outsourced biometric data $[D]$ are encrypted with a fully homomorphic encryption scheme [17], P_1 can perform any possible computations over the encrypted data. However, fully homomorphic encryptions have yet to be practical due to their extremely high computation costs.

In our proposed O-PPBA protocol, in addition to P_1, we will utilize another independent cloud service provider P_2 and use an additive homomorphic encryption scheme (AH-Enc) to encrypt each v_i and u component-wise to produce $[v_i]$ and $[u]$. There are several AH-Enc schemes, and without loss of generality, we adopt the Paillier cryptosystem [38] for its simple implementation and being semantically secure. $[D]$ is outsourced and stored at P_1, and only P_2 has the corresponding decryption key. Therefore, $[D]$ and $[u]$ cannot be decrypted by P_1 without accessing the decryption key.

To solve the proposed O-PPBA problem, one might suggest using garbled circuits [24] for every secure computation between P_1 and P_2. This is logical for simple tasks (e.g., secure comparison), where such an approach can be quite efficient. However, biometric authentication is complex process consisting of several sub-components. In Section 4.1, we present these sub-components and investigate the most efficient and secure way to implement them. The proposed O-PPBA combines homomorphic encryption and garbled circuit to take advantages offered by each approach. This hybrid approach leads to a more efficient protocol as shown in Section 5.

The rest of this paper is organized as follows: Section 2 overviews the work related image retrieval and existing PPBA protocols, Section 3 presents technical background and threat models, Section 4 presents our proposed O-PPBA protocol, Section 5 provides empirical results showing the run time complexity of the proposed protocol, and we summarize our contributions and future research in Section 6.

2. RELATED WORK

Since our work is directly related to the concept of secure multiparty computation (SMC), in this section, we first provide a high level discussion of SMC. In addition, we provide an overview of content based image retrieval techniques and the existing privacy-preserving biometric identification protocols. The formulation of privacy-preserving biometric identification (PPBI) is slightly different from privacy-preserving biometric authentication (PPBA) which is considered in this paper.

A PPBI protocol generally returns the profile of an person whose biometric data record stored at the server matches the user's input biometric data record. On the other hand, a PPBA protocol only returns a single bit to indicate if there is a match or not. The existing PPBI protocols can be easily modified to satisfy the PPBA problem statement definition. From a technical perspective, the difference between PPBI and PPBA is negligible. Since there is a lack of exiting work directly related to PPBA, here we will mainly present the work related to PPBI and discuss the reasons why the existing PPBI protocols cannot be directly applied to solve the outsourceable PPBA problem.

2.1 Secure Multiparty Computation

Secure multiparty computation (SMC) was first introduced by Yao's Millionaire problem [51] where Alice and Bob want to know who is richer without disclosing their actual wealth to each other. Suppose there are n parties $P_1, \ldots, P_n$ who hold private inputs $a_1, \ldots, a_n$ respectively. An SMC protocol allows $P_1, \ldots, P_n$ to collaboratively compute a function f on inputs $a_1, \ldots, a_n$ without disclosing a_i to P_j where $1 \leq i, j \leq n$ and $i \neq j$. To achieve that, the participating parties have to exchange messages and perform some local computations until all the parties get the desired output. In general, we need to make sure that the messages exchanged should not disclose any information regarding each party's private input.

2.2 Content based Image Retrieval

Biometric authentication or identification is a special case of content based image retrieval (CBIR). In a CBIR system, a desirable image is retrieved from a large image database based on the similarity of some common attributes which are called features. There are two kinds of features: the global and local. Global features usually include color, texture and shape which are well known and studied in image retrieval. Local features are usually used in a localized circumstance which needs to recognize objects more precisely.

The basic way to separate images is to use color. Thus, color is a very important visual feature for image retrieval. In existing literatures many color space models are available. The RGB model is the most commonly used color space model that represents colors by red (R), green (G) and blue (B). In most RGB systems, the values of red, green and blue are depending upon the nature of the primaries in a certain imaging device [46]. Thus, other models such as HSI, HSV, LAB, LUV and YCrCb have been proposed to overcome of RGB shortcomings. When those color spaces models, the color histogram scheme is the most effective way to represent a image since color histograms are invariant to orientation and scale [9,31]. However, color histogram can hardly characterize the spatial structures of images; as a result, other descriptors have been proposed such as compact color moments, the color coherence vector and color correlograms [23].

Texture is another important characteristics of an image. Various algorithms have been designed for texture analysis such as gray level co-occurrence matrices [39], the Tamura texture feature [21], and Gabor filtering [7]. Co-occurrence matrix is a popular representation of texture feature of an image which is constructed based upon the orientation and distance between image pixels [39]. Tamura texture feature is another popular representation which is based upon the human mental study in visual perception of texture [21]. Gabor filters are very effective that applying signal processing and wavelet transform methods in texture analysis [7]. The wavelet transform is used for image classification based on multi-resolution decomposition of images.

In addition, shape features are also playing a critical role in CBIR because human beings can recognize objects solely based on shapes. Main method for shape features is the use of moment invariants [47]. Usually shape feature extraction requires the use of image segmentation, where an image needs to be divided into different regions.

Beside of those global feature-based algorithms (color, texture, shape), local feature-based algorithms are more precise but time-consuming. Several types of local descriptors have been proposed in the literature [3, 28, 32, 42, 49]. One very effective technique is termed as scale-invariant feature transfer (SIFT) [32] which extracts features that do not change when exposed to a set of image transformations. Speed-Up Robust Features (SURF) [3] is another local feature based technique which is faster than other descriptors including SIFT. In addition, SIFT has its advantages in rotation and illumination changes against SURF [33].

2.3 Privacy-Preserving Biometric Authentication / Identification

To the best our knowledge, Erkin et al. [13] proposed the first privacy-preserving biometric face recognition protocol based on the standard Eigenfaces recognition algorithm. The protocol is a secure two-party computation protocol where one party (client) wants to learn whether its candidate biometric reading matches for one or more records in the other party's (server's) database without disclosing any information except the final result. The protocol computes Euclidean distances between face image feature vectors from client and server's face image database, and it returns the profile information associated with the facial image data record that has the smallest distance to the client's input biometric data. At the end, the client only knows the profile information without knowing other biometric data and their associated profiles stored at the server, and the server knows nothing, even regarding the profile information returned to the client. The data exchanged during an execution of the protocol are encrypted by an additive homomorphic encryption scheme, but the data stored at the sever are not encrypted. Therefore, the server knows the contents of its biometric database.

Sadeghi et al. [43] developed a hybrid privacy-preserving face recognition protocol which improved the efficiency of Erkin's work. This protocol follows the problem setting of Erkin's work and also uses additive homomorphic encryption (the Paillier cryptosystem) to securely compute Euclidean distances. On the other hand, Garbled circuits [24, 50] are used for finding the minimum distance. The authors also proposed a method of packing multiple values together into a single ciphertext before blinding. This can save communication cost significantly.

Huang et al. developed an efficient privacy-preserving biometric identification protocol for fingerprint recognition [15]. The basic algorithm for fingerprint identification is based on $FingerCode$ [27]. In the main system, the server and a client jointly perform the protocol to retrieve the identity record from the server's database whose fingerprint best matches the client's input fingerprint reading. The protocol provides the same security guarantee as the previously mentioned protocols. In addition, similar to Sadeghi's work, this protocol also combines additive homomorphic encryption with garbled circuits. However, the protocol improves efficiency for both distance-computing phase and matching phase compared to Sadeghi's work. The protocol separates retrieval step from matching phase, and it uses the by-product of evaluating the garbled circuit in the matching phase to perform the oblivious retrieval efficiently. In the retrieval phase, the protocol utilizes a backtracking tree to obliviously and efficiently recover the profile corresponding to the closest matching vector.

Different from calculating Euclidean distance, Blanton and Gasti [6] developed a privacy-preserving identification protocol for iris codes based on Hamming distance. In their protocol, the client who possesses an iris reading would also like to learn whether his or her reading matches one or more records in an iris database managed by the server. The contents of the server's database are never disclosed to the client, but the client learns the comparison results between his or her input iris record and every record in the server's database. The implementation of the protocol also uses both additive homomorphic encryption and garbled circuits, and it reduces the complexity of the circuits used for comparison based on an optimization that permits XOR gates to be evaluated for free. The proposed techniques can be applied in protocols where $FingerCode$ and Euclidean distances are adopted to improve the performance compared to the existing Euclidean distance based solutions.

SCiFI [37] is a realization of a privacy-preserving face identification system which uses a component-based face identification technique which building a binary index into a vocabulary representation. The adopted image representation technique is robust against different viewing conditions, such as illumination, occlusions, and changes in appearance (like wearing glasses). The protocol utilizes Hamming distance to measure image similarity, and its implementation were based on additive homomorphic encryption and oblivious transfer [34].

Although the aforementioned privacy-preserving biometric identification / authentication protocols protect the confidentiality of both server and client's biometric data, our problem setting is quite different from these protocols. In our problem domain, the biometric database is encrypted and outsourced to a server (or a cloud), and the server does not have the key to decrypt the data. Thus, the server does not know anything regarding the original biometric database. Since the previously proposed solutions require the server to perform computations on the original biometric data, these solutions cannot be applied to solve this outsourced biometric authentication problem.

Recently, Blanton and Aliasgari [5] developed a secure approach to outsourcing computations of matching iris biometric data records. The setting of their work is very similar to ours. Two protocols were proposed for either single server-setting or multiple-server setting. In their single-server setting, the protocol uses predicate encryption scheme [29,45] that allows the server to perform non-interactive computations. The predicate encryption scheme is not as secure as additive homomorphic encryption. As a result, our proposed protocol offers much better confidentiality protection of the biometric data. Under the multiple-server setting, the protocol adopts a secret sharing scheme (e.g., Shamir [44]) to "encrypt" the outsourced biometric database. However, the protocol requires at least three independent servers to perform the intermediate computations. Our proposed protocol requires only two servers, so it is more practical.

In addition, the protocol mainly focuses on calculating Hamming distance, whereas our proposed protocol can compute Euclidean and Hamming distances. Since both distances are commonly used to image retrieve according to different kinds of biometric data, our protocol offers more generality. Another different is that at the end of protocol execution, the server can know which encrypted biometric data record matches the client's input. This access pattern leakage can violate the security guarantee of the underlying encryption scheme [26]. Because our protocol does not disclose this information to any participating party, our protocol offers the same security protection as the encryption scheme used to encrypt the outsourced biometric data.

3. SECURITY DEFINITIONS AND ADVERSARY MODELS

In this section, we first discuss the encryption scheme used to build the proposed protocols. Since the notion of security varies from different application domains, we also present the security threat or adversary models that best match the proposal application domain.

3.1 Additive Homomorphic Encryption

The proposal protocol adopts an additive homomorphic probabilistic public key encryption (AH-Enc) scheme as the building block. Let E_{pk} and D_{pr} be the encryption and decryption functions in an AH-Enc scheme with public key pk and private key pr. Without pr, no one can discover x from $E_{pk}(x)$ in polynomial time. Since the encryption and decryption keys are unique in our problem domain, we will adopt the succinct notation $[x]$ to represent an encryption or ciphertext of x. an AH-Enc has the following properties:

- The encryption function is additive homomorphic: $[x_1 + x_2] = [x_1] \times [x_2] \mod N^2$;
- Given a constant c and $[x]$, $[c \cdot x] = [x]^c \mod N^2$;
- The encryption function has semantic security as defined in [20], i.e., a set of ciphertexts do not provide additional information about the plaintext to an adversary with polynomial-bounded computing power. In other words, two encryptions of x differs with very high probability. This is the probabilistic property of the AH-Enc system.

Any AH-Enc system is applicable, but in this paper, we adopt the Paillier encryption scheme [38] for its implementation simplicity. Informally speaking, the public key in the system is (g, N), where N is resulted from multiplication of two large prime numbers and $g \in \mathbb{Z}^*_{N^2}$ is randomly chosen. When computations are performed on ciphertexts, we need to perform a modulo N^2 operation at the end (e.g., $[x_1 + x_2] = [x_1] \times [x_2] \mod N^2$). Because N is unique in the proposed protocols, we will drop or eliminate this modulo N^2 operation whenever the context is clear.

3.2 Threat / Adversary Models

In general, there is a conceptual difference between *privacy* and *security*. However, in this paper, we will not differentiate the two terms. Regarding a distributed protocol, security is generally related to the amount of information leaked during the protocol execution. To maximize security

Table 1: Common Notations

SMC	Secure Multiparty Computation
PPBA	Privacy-Preserving Biometric Authentication
O-PPBA	Outsourceable PPBA
AH-Enc	Additive homomorphic encryption
v_i or u	m-dimension feature vector representation of a biometric data image
$[v_i]$ or $[u]$	A component-wise encryption of v_i, using an AH-Enc scheme
D	A biometric database with n records: $v_1, \dots, v_n$
$[D]$	An encryption of D: $[v_1], \dots, [v_n]$
P_1 or P_2	A cloud service provider

protection, we use the security definitions in the literature of secure multiparty computation (SMC).

SMC-based secure protocols generally assume there basic adversarial models: semi-honest (or honest but curious), malicious and covert [2,18]. Semi-honest adversaries follow the protocol faithfully, but can try to infer the secret information of the other parties from the data they see during the protocol execution. Malicious adversaries may do anything to infer secret information. On the other hand, under the covert adversary model, malicious behaviors can be detected without disclosing private information. Almost all practical SMC-based protocols are secure under the semi-honest model, and using zero-knowledge proofs [19], these protocols can be transformed into protocols secure under the malicious model. Also, it is hard to imagine that two well-established cloud service providers can collude to damage their reputations and customer base. Therefore, we will adopt the semi-honest model. In addition, since the covert model provides a nice balance between the semi-honest model and the malicious model, we will also proposed strategies to make the proposed protocol secure under the covert model. Briefly, the following definition captures the security definition under the semi-honest model.

Definition 1. *Let a_i be the input of party P_i, $\prod_i(\pi)$ be P_i's execution image of the protocol π and b_i be the result computed from π for P_i. π is secure if $\prod_i(\pi)$ can be simulated from $\langle a_i, b_i\rangle$ and distribution of the simulated image is computationally indistinguishable from $\prod_i(\pi)$.*

A formal way to prove the security of a protocol under the semi-honest model is to use the simulation approach [18]. In our security proofs, we will adopt this method.

4. THE PROPOSED TWO-PARTY O-PPBA PROTOCOL

We will follow the same notations as given in Sections 1.1 and 1.3. For clearness, commonly used notations for this paper are given in Table 1. According to the number of participants, an SMC protocol can be classified into either two-party or multi-party protocol. In our problem domain, it is more cost effective to use two cloud service providers, so we will focus on developing a two-party O-PPBA protocol. For a functionality as complex as biometric authentication, the best way to implement an efficient two-party secure protocol is to combine homomorphic encryption approach and garbled circuit. This hybrid approach was adopted in [36] to produce a secure protocol that is more efficient than either homomorphic encryption approach or garbled circuit approach alone. The key challenges to utilize the hybrid approach are:

- Break a complex functionality into a set of simpler sub-components (or sub-functionalities).
- Determine which approach to use to implement each sub-component securely.

Next we will dissect O-PPBA into sub-components and identify the most efficient approach, between homomorphic encryption or garbled circuit, to implement each component.

4.1 Sub-Components of O-PPBA

To implement an O-PPBA protocol, first each biometric image needs to be represented as a feature vector. This pre-processing step can be implemented using the existing image representation techniques as summarized in Section 2.2. For the rest of this paper, we assume that the feature vector of each biometric image is given as part of the input to an O-PPBA protocol.

According to Equation 1 and based on the description given in Section 1.1, an O-PPBA protocol (with input $[D] = \{[v_1], \dots, [v_n]\}$, $[u]$ and t) consists three main sub-components or protocols that need to be performed sequentially. According to the Composition Theorem [18], when a protocol is implemented based on a set of secure primitives, and for the protocol to be secure, the intermediate results produced from these primitives must be in the form of random shares. By random shares, we mean that the actual value is divided into two additive random shares, each of which is independently and uniformly distributed in a group (e.g., $\mathbb{Z}_N = \{0, 1, \dots, N-1\}$). The actual value can be reconstructed by adding the two shares modulo N. Next we describe each sub-component and its outcomes.

(1) *Distance computation*: This component computes the distance between u and each v_i, for $1 \leq i \leq n$. Note that the distance is between the actual feature vectors, but the input should be the encrypted feature vectors $[u]$ and $[v_i]$. During the computation, $[u]$ and $[v_i]$ are never decrypted to preserve the confidentiality of u and v_i. Let d_i denote the distance between u and v_i. The output of this component returns two random shares d_i' and d_i'' for each d_i such that $d_i' + d_i'' \mod N = d_i$.

(2) *Comparing with the threshold*: Once the distances are computed, these distances need to be compared with the threshold t, to find out if u is one of $v_1, \dots, v_n$. Distance less than t indicating u matches some biometric record in D. Again, all these computations are based on encrypted data or random shares. More specifically, the (d_i', d_i'') pairs and the threshold t are the input for this sub-component. Let b_i denote the comparison result between d_i and t. $b_i = 1$ if $d_i \leq t$; otherwise $b_i = 0$. The output of this component returns two random shares b_i' and b_i'' for each b_i such that $b_i' + b_i'' \mod N = b_i$.

(3) *Combining the comparison results*: As long as there is a b_i equal to 1, we can conclude that u matches some v_i in D. Then authentication succeeds. The (b_i', b_i'') are the input for this sub-component. Let b_f denote the final authentication result. If $b_f = 1$, then authentication

succeeds. On the other hand, $b_f = 0$, authentication fails. The output of this component returns two random shares b'_f and b''_f such that $b'_f + b''_f \mod N = b_f$.

Note that, it is possible for the last component to return b_f directly, but b'_f and b''_f are more useful in case b_f serves as an intermediate result for a more complex protocol. Base their need, the participating parties can decide the appropriate output. Without loss of generality, this paper adopts b'_f and b''_f as the final outcome of the proposed O-PPBA protocol.

4.2 Efficient and Secure Implementation of Each Sub-Component

According the descriptions of the sub-components given in the previous section, this section investigates how to implement them efficiently and securely. Depending on the specific functionality or computation, we will identify the most efficient and secure method (either homomorphic encryption based approach or garbled circuit) to implement each sub-component.

4.2.1 Secure Distance Computation

As mentioned previously, since biometric data have multiple types and each type can be represented in various ways, here we will implement secure sub-protocols to compute two most common distance metrics adopted in the existing PPBI work (summarized in Section 2.3): Euclidean distance and Hamming distance. Both metrics have their advantages and disadvantages, and an entity can decide which metric to use for its O-PPBA protocol.

Euclidean Distance.

Suppose v_i and u are m-dimensional vectors, and the content of the individual dimension is represented by $v_i\langle j\rangle$ ($u\langle j\rangle$), where $1 \le j \le m$. The Euclidean distance between v_i and u can be computed as follows:

$$\text{Euclidean\_Dist}(v_i, u) = \sqrt{\sum_{j=1}^{m}(v_i\langle j\rangle - u\langle j\rangle)^2}$$

When comparing two distances, the square root does not make any difference. Thus, in our implementation, we will drop the square root to compute the square of the Euclidean distance between v_i and u:

$$\text{Euclidean\_Dist}^2(v_i, u) = \sum_{j=1}^{m}(v_i\langle j\rangle - u\langle j\rangle)^2 \qquad (2)$$

According to the above equation, the main challenge is to compute $v_i\langle j\rangle^2$, $u\langle j\rangle^2$, and $v_i\langle j\rangle \cdot u\langle j\rangle$ from $[v_i\langle j\rangle]$ and $[u\langle j\rangle]$ (where $[v_i\langle j\rangle]$ and $[u\langle j\rangle]$ are encryptions of $v_i\langle j\rangle$ and $u\langle j\rangle$). The key steps to securely compute Euclidean_Dist$^2(v_i, u)$ are given by Algorithm 1 where computations are only based on either encrypted or randomized data. At step 1(a) of the algorithm, x_j is a randomized difference between $v_i\langle j\rangle$ and $u\langle j\rangle$, and r_j is randomly chosen to hide the actual difference. According to the additive homomorphic property of the encryption scheme, $[x_j]$ can be computed as follows:

$$[v_i\langle j\rangle - u\langle j\rangle + r_j] \leftarrow [v_i\langle j\rangle] \times [u\langle j\rangle]^{N-1} \times [r_j]$$

Since P_2 has the decryption key, it can decrypt $[x_j]$ to get x_j. Then P_2 computes the square of x_j and sends the encrypted results (e.g., $[x_1], \ldots, [x_m]$) to P_1. Because P_1 knows r_j and

Algorithm 1 Secure_Distance$^e([v_i], [u]) \rightarrow (d'_i, d''_i)$

```
Require: $P_1$ has $[v_i]$ and $[u]$; $P_2$ has the decryption key
1: $P_1$:
   (a) $[x_j] \leftarrow [v_i\langle j\rangle - u\langle j\rangle + r_j]$, where $r_j \in_R \mathbb{Z}_N$ and $1 \le j \le m$
   (b) Send $[x_1], \ldots, [x_m]$ to $P_2$
2: $P_2$:
   (a) Perform decryption to get $x_1, \ldots, x_m$
   (b) Compute $[(x_1)^2], \ldots, [(x_m)^2]$, and send them to $P_1$
3: $P_1$:
   (a) Obtain $[(v_i\langle j\rangle - u\langle j\rangle)^2]$ by eliminating $r_j^2$ and $2r_j(v_i\langle j\rangle - u\langle j\rangle)$ from $[(x_j)^2]$, for $1 \le j \le m$
   (b) $[d_i] \leftarrow [sum_{j=1}^{m}(v_i\langle j\rangle - u\langle j\rangle)^2]$
   (c) Compute $[d_i + r]$, where $r \in_R \mathbb{Z}_N$
   (d) Set $d'_i = N - r$ and send $[d_i + r]$ to $P_2$
4: $P_2$:
   (a) Perform decryption on $[d_i + r]$ to get $d_i + r$
   (b) Set $d''_i = d_i + r$
```

$[v_i\langle j\rangle - u\langle j\rangle]$, it can easily compute $[r_j^2]$ and $[2r_j(v_i\langle j\rangle - u\langle j\rangle)]$. Since $[(x_j)^2] = [(v_i\langle j\rangle - u\langle j\rangle)^2 + 2r_j(v_i\langle j\rangle - u\langle j\rangle) + r_j^2]$, P_1 can obtain $[(v_i\langle j\rangle - u\langle j\rangle)^2]$ by:

$$[(v_i\langle j\rangle - u\langle j\rangle)^2] \leftarrow [(x_j)^2] \times ([v_i\langle j\rangle - u\langle j\rangle]^{2r_j} \times [r_j^2])^{N-1}$$

From here, $[d_i]$ can be directly derived:

$$[d_i] \leftarrow [(v_i\langle 1\rangle - u\langle 1\rangle)^2] \times \cdots \times [(v_i\langle m\rangle - u\langle m\rangle)^2]$$

The rest of the steps are straightforward.

Hamming Distance.

When v_i and u are binary vectors, Hamming distance can be used to determine how close v_i and u are. Hamming distance can be computed as follows:

$$\text{Hamming\_Dist}(v_i, u) = m - \sum_{j=1}^{m} v_i\langle j\rangle \cdot u\langle j\rangle \qquad (3)$$

According to the above equation, the main challenge is to compute $v_i\langle j\rangle \cdot u\langle j\rangle$ from $[v_i\langle j\rangle]$ and $[u\langle j\rangle]$. The key steps to securely compute Hamming_Dist(v_i, u) are given by Algorithm 2 where computations are only based on either encrypted or randomized data. At step 1(a) of the algorithm, x_{j_1} and x_{j_2} are randomizations of $v_i\langle j\rangle$ and $u\langle j\rangle$ respectively, and r_{j_1} and r_{j_2} are randomly chosen to hide the actual values of $v_i\langle j\rangle$ and $u\langle j\rangle$. According to the additive homomorphic property of the encryption scheme, $[x_{j_1}]$ and $[x_{j_2}]$ can be computed as follows:

$$[v_i\langle j\rangle + r_{j_1}] \leftarrow [v_i\langle j\rangle] \times [r_{j_1}]$$

$$[u\langle j\rangle + r_{j_2}] \leftarrow [u\langle j\rangle] \times [r_{j_2}]$$

Algorithm 2 Secure_Distance$^h([v_i], [u]) \rightarrow (d'_i, d''_i)$

```
Require: $P_1$ has $[v_i]$ and $[u]$; $P_2$ has the decryption key
1: $P_1$:
    (a) $[x_{j_1}] \leftarrow [v_i\langle j\rangle + r_{j_1}]$ and $[x_{j_2}] \leftarrow [u\langle j\rangle + r_{j_2}]$, where $r_{j_1}, r_{j_2} \in_R \mathbb{Z}_N$ and $1 \le j \le m$
    (b) Send $([x_{1_1}], [x_{1_2}]), \ldots, ([x_{m_1}], [x_{m_2}])$ to $P_2$
2: $P_2$:
    (a) Perform decryption operation to get $(x_{1_1}, x_{1_2}), \ldots, (x_{m_1}, x_{m_2})$
    (b) Compute $[x_{1_1} \cdot x_{1_2}], \ldots, [x_{m_1} \cdot x_{m_2}]$, and send them to $P_1$
3: $P_1$:
    (a) Obtain $[v_i\langle j\rangle \cdot u\langle j\rangle]$ by eliminating $r_{j_1} \cdot r_{j_2}$, $r_{j_2} \cdot v_i\langle j\rangle$ and $r_{j_1} \cdot u\langle j\rangle$ from $[x_{j_1} \cdot x_{j_2}]$, for $1 \le j \le m$
    (b) $[d_i] \leftarrow [m - \sum_{j=1}^m v_i\langle j\rangle \cdot u\langle j\rangle]$
    (c) Compute $[d_i + r]$, where $r \in_R \mathbb{Z}_N$
    (d) Set $d'_i = N - r$ and send $[d_i + r]$ to $P_2$
4: $P_2$:
    (a) Perform decryption on $[d_i + r]$ to get $d_i + r$
    (b) Set $d''_i = d_i + r$
```

Since P_2 has the decryption key, it can decrypt $([x_{j_1}], [x_{j_2}])$ to get (x_{j_1}, x_{j_2}) at step 2(a). Then P_2 computes the multiplication of $x_{j_1} \cdot x_{j_2}$ and sends the encrypted results to P_1. Because P_1 knows r_{j_1}, r_{j_2}, $[v_i\langle j\rangle]$ and $[u\langle j\rangle]$, it can easily compute $[r_{j_1} \cdot r_{j_2}]$, $r_{j_2} \cdot v_i\langle j\rangle$ and $r_{j_1} \cdot u\langle j\rangle$. Since $[x_{j_1} \cdot x_{j_2}] = [v_i\langle j\rangle \cdot u\langle j\rangle + r_{j_1} \cdot r_{j_2} + r_{j_2} \cdot v_i\langle j\rangle + r_{j_1} \cdot u\langle j\rangle]$, at step 3(a), P_1 can obtain $[v_i\langle j\rangle \cdot u\langle j\rangle]$ by:

$$[v_i\langle j\rangle \cdot u\langle j\rangle] \leftarrow [x_{j_1} \cdot x_{j_2}] \times ([r_{j_1} \cdot r_{j_2}] \times [r_{j_2} \cdot v_i\langle j\rangle] \times [r_{j_1} \cdot u\langle j\rangle])^{N-1}$$

From here, $[d_i]$ can be directly derived at step 3(b):

$$[d_i] \leftarrow [m] \times ([v_i\langle 1\rangle \cdot u\langle 1\rangle] \times \cdots \times [v_i\langle m\rangle \cdot u\langle m\rangle])^{N-1}$$

The rest of the steps are fairly obvious. All above computations are based on the additive homomorphic property of the encryption schemes. The two sub-protocols are more efficient than garbled circuit based solutions, and detailed analysis is provided in Section 5. The security analysis of both protocols is presented in Section 4.3.2.

4.2.2 Securely Comparing with the Threshold

To implement the second sub-component securely, we need a secure comparison protocol that takes random shares (i.e., d'_i and d''_i) and a threshold t as input and returns two random shares b'_i and b''_i, such that $b'_i + b''_i \mod N = 1$ if $d_i \le t$; otherwise $b'_i + b''_i \mod N = 0$. The inputs of most existing secure comparison protocols [4,10,11,16,35] are non-encrypted or non-random shares, so they are not directly applicable in our problem domain. On the other hand, garbled circuit has been known for its efficiency to securely evaluate simple functionalities, such as comparison. In addition, garbled circuit only requires one round of communication and can be easily modified to fit our problem domain. As a result, we adopt the fast garbled circuit method [24] to implement this sub-component.

Algorithm 3 O-PPBA$([D], [u], t) \rightarrow b$

```
Require: $P_1$ has $[D]$ and $[u]$, $P_2$ has the decryption key, and $t$ is a public parameter
1: $P_1$ and $P_2$ jointly execute Secure_Distance($[v_i], [u]$), for $i = 1$ to $n$
2: $P_1$ and $P_2$ jointly execute Secure_Comparison($d'_i, d''_i, t$), for $i = 1$ to $n$
3: $P_1$: $\alpha \leftarrow \sum_{i=1}^n b'_i \mod N$
4: $P_2$: $\beta \leftarrow \sum_{i=1}^n b''_i \mod N$
5: $P_1$ and $P_2$ jointly execute Secure_Comparison($\alpha, \beta, 0$)
6: $b \leftarrow b'_f + b''_f \mod N$
```

Technical details on how to produce and evaluate a garbled circuit are well documented in [24]. In this paper, we will skip these technical details. Instead, we assume that Secure_Comparison$(d'_i, d''_i, t) \rightarrow (b'_i, b''_i)$ is the protocol to implement the sub-component and it is constructed using a garbled circuit, where d'_i and d''_i are private input values from P_1 and P_2 respectively, and t is publicly known parameter. The protocol returns b'_i to P_1 and b''_i to P_2.

4.2.3 Securely Combining the Comparison Results

As discussed in Section 4.1, the third sub-component / functionality is to identify the existence of a comparison result that is equal to 1. To maximize security guarantee, the individual comparison results cannot be disclosed; otherwise, the first two sub-components would be sufficient. Now the challenge is to find out whether there is a comparison result of 1 without disclosing which encrypted biometric record $[v_i]$ matches $[u]$. Since a server only knows if two encrypted biometric data records match, we may think that disclosing the matching result is harmless. However, according to [26], the matching result must not be leaked to preserve the security guarantee of the underlying encryption scheme.

To prevent the matching or comparison results from disclosing to P_1 and P_2, the design of our protocol is based on the following observations:

Observation 1. *Let $b_1, \ldots, b_n$ be the n actual comparison results from comparing $d_1, \ldots, d_n$ with the threshold t, where $d_1, \ldots, d_n$ are the distances between u and each of $v_1, \ldots, v_n$. Then $b_f = 1$ if and only if $0 < \sum_{i=1}^n d_i$.*

Observation 2. *Let $\alpha = \sum_{i=1}^n b'_i$ and $\beta = \sum_{i=1}^n b''_i$. Then we have $\alpha + \beta \mod N = \sum_{i=1}^n b_i$.*

From the above observations, it is not hard to see that this sub-component can be implemented using the same secure comparison protocol as discussed in Section 4.2.2. More specifically, Secure_Comparison$(\alpha, \beta, 0) \rightarrow (b'_f, b''_f)$ is the protocol to implement the sub-component and it is constructed using a garbled circuit, where α and β are private input values from P_1 and P_2 respectively, and 0 is a public parameter. The protocol returns b'_f to P_1 and b''_f to P_2 such that $b'_f + b''_f \mod N = b_f$.

4.3 The O-PPBA Protocol

Once we have proper secure implementations of the three sub-components, we can build an O-PPBA protocol directly based on these components. The key steps of the O-PPBA protocol are given in Algorithm 3. Here we assume that a user's encrypted biometric data record $[u]$ has been received by P_1, and it is encrypted using the same public key as the one to encrypt D to produce $[D]$, where $[D]$ is encrypted by the entity who owns the biometric database D.

Step 1 of Algorithm 3 securely computes the distance between u and each of $v_1, \ldots, v_n$. P_1 and P_2 receive $d'_1, \ldots, d'_n$ and $d''_1, \ldots, d''_n$ as their private output respectively. Note that in the protocol, we do not specify which distance metric (either Euclidean or Hamming). Depending on the type of biometric data and the feature vector, the entity who outsourced its biometric authentication to P_1 and P_2 can decide the appropriate metric to use. Secure_Distance^e and Secure_Distance^h in Algorithm 3 are interchangeable without affecting the security and the correctness of O-PPBA.

The output from step 1 serves as the input for step 2. At the end of step 2, P_1 receives $b'_1, \ldots, b'_n$ and P_2 receives $b''_1, \ldots, b''_n$. P_1 and P_2 independently compute α and β which are input for step 5 of the algorithm. Step 5 returns b'_f to P_1 and b''_f to P_2. To obtain the actual authentication result, P_2 can send d''_f to P_1, and b_f can be reconstructed by adding the two random shares modulo N. Steps 2, 3 and 5 of Algorithm 3 correspond to the three sub-component and their implementations investigated in Sections 4.1 and 4.2.

4.3.1 Complexity Analysis

Here we analyze both computation and communication complexities of the proposed O-PPBA protocol. First, we analyze the computation complexity for each sub-protocol. For the Secure_Distance^e protocol, the number of multiplications and encryptions is bounded by $O(m)$, where m is the vector size. To derive this upper bound, we need to examine each step of the protocol. At step 1, the protocol performs $2m$ encryptions and $2m$ multiplications. Here we treat $[u\langle j\rangle]^{N-1}$ as one encryption although it is slightly less expensive than an encryption operation. Step 2(a) performs m decryptions. A decryption has a similar cost as an encryption, so we count m encryptions for step 2(a). Step 2(b) performs m encryptions and m multiplications. Step 3(a) performs $3m$ encryptions and $2m$ multiplications since we count the two exponentiations as two encryptions. Step 3(b) requires $m-1$ multiplications. The computations for the rest of the steps are constant. Therefore, the total number of multiplications and encryptions for this sub-protocol is bounded by $O(m)$. In addition, because an encryption operation is generally several order of magnitude more expensive than a multiplication, we can claim the computation complexity of Secure_Distance^e is bounded by $O(m)$ encryptions. Similarly, we can derive the same asymptotic bound for Secure_Distance^h.

The inputs to the Secure_Comparison protocol are two random shares with size bounded by N, so the initial phase of the garbled circuit needs $O(\log N)$ gates to add the shares. The resulting distance value is much smaller than N is our problem domain. Therefore, the total number of gates in the garbled circuit is bound by $O(\log N)$. Since the fast garbled circuit evaluation method proposed in [24] is every efficient and use symmetric key encryption to garble the circuit. We can expect that the total computation cost of the Secure_Comparison protocol is about performing several homomorphic encryption operations, so $O(m)$ encryptions provide an appropriate upper bound for Secure_Comparison. The O-PPBA protocol executes the Secure_Distance protocol n times and the Secure_Comparison protocol $n+1$ times. As a consequence, the total computation complexity of O-PPBA is bound by $O(mn)$ encryptions.

To analyze the communication complexity, we calculate an asymptotic bound on the number of bits exchanged between P_1 and P_2. Let k denote the number of bits to represent N. Then the communication complexity of Secure_Distance is bound by $O(mk)$. Also, for a sufficiently large m, the communication complexity of Secure_Comparison is bound by $O(mk)$. As a result, the total communication complexity of O-PPBA is bounded by $O(mnk)$ in bits.

4.3.2 Security Analysis and Extensions to Other Adversary Models

The O-PPBA protocol is secure under the semi-honest adversary model of SMC. Since the protocol is a sequential composition of sub-protocols, to prove the security of O-PPBA, we first need to prove the security of each sub-protocol. The security of Secure_Distance^e depends on the size of the encryption key. In general, N needs to be at least 1024 bit long, and then we can say that the Paillier encryption scheme possesses semantic security. Thus, in our proof, we assume N is 1024-bit or larger. Under the semi-honest model, the security guarantee is very easy to proof because the computations are performed on either encrypted data or randomized data. To be more clear, here we analyze step-by-step how the original biometric data records are protected or never disclosed.

At step 1(a) of Algorithm 1, the computations are directly performed on each pair of $[v_i\langle j\rangle]$ and $[u\langle j\rangle]$ and they are never decrypted, no information is leaked regarding v_i and u as long as the encryption scheme is semantically secure. The only place where the original data could be disclosed is at step 2(a) since decryption operations are performed on the intermediate computation results. However, because $x_j = v_i\langle j\rangle - u\langle j\rangle + r_j \mod N$, and r_j is randomly chosen and only known to P_1, x_j is a uniformly distributed in $\mathbb{Z}_N$ from the view point of P_2. Therefore, x_j does not leak any information regarding v_i and u to P_2. The rest of the computations is based on either encrypted or randomized data. As a consequence, if P_1 and P_2 do not collude, no information related to v_i and u is ever disclosed to the two parties. Note that collusion is prohibited under the semi-honest model.

The Secure_Distance^h protocol has the same structure as the Secure_Distance^e protocol, and it is secure under the semi-honest model as well. Secure_Comparison is implemented using a garbled circuit which is also secure under the semi-honest model [24]. Since all these sub-protocols of O-PPBA produce random shares as intermediate results, according to the sequential Composition Theorem [18], the O-PPBA protocol is also secure under the semi-honest model.

Justification of the Use of the Semi-Honest Model.

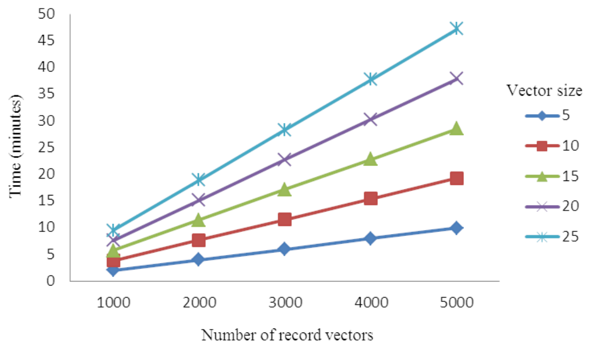
This adversary model assumes that the participating parties follow the prescribed steps of the protocol. The model may not be appropriate for many situations when we do not have any background knowledge about the parties, e.g., P_1 and P_2 in our case. On the other hand, the semi-honest model suits our problem domain very well because we assume P_1 and P_2 are two cloud service providers. Today, cloud service providers are legitimate and well-known companies like Google, Amazon and Microsoft. It is hard to believe any two of these companies collude to discover the original biometric data. Their reputations are invaluable assets and need to be protect at all costs. Therefore, we can safely assume P_1 and P_2 are semi-honest.

Since a user only sends his or her encrypted biometric data $[u]$ and never participate in any intermediate computations, how the user behaves is irrelevant to the security of the protocol. Just like any authentication protocol, a user can try to probe the system using different passwords, which is out of scope of this paper. However, there does exist technique to either prevent or mitigate this probing attacking. These techniques are independent from but their functionalities complement the proposal O-PPBA protocol. Also, the initial set-up of the biometric authentication system is performed by the entity who owns the biometric database D. All these issues are irrelevant to the security of the O-PPBA protocol. Thus, we will not discuss them any further.

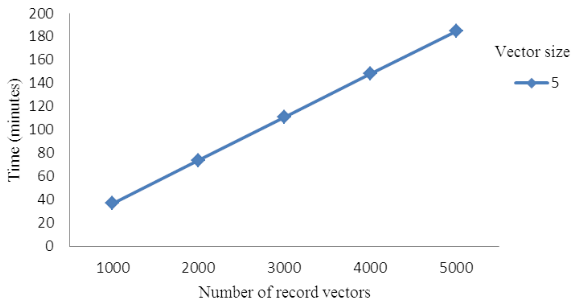
Extension to Other Adversary Models.

In case P_1 and P_2 are not as trustworthy as the existing cloud service providers, we need to extend the O-PPBA protocol to satisfy other adversary models, such as malicious and covert [2] models. Protocols that satisfy the malicious model are generally very expensive and do not have much practical value. This is also evidenced by the fact that almost all practical SMC-protocols are secure under the semi-honest model. In addition, input modification is an open problem in SMC [18]. Before executing an SMC protocol, a participating party can provide wrong input. Since each party's input is private, there is no way to verify the input provided by the party is its actual input. This phenomenon is called input modification, and no solution exists under the general setting of SMC. Because each sub-protocol is non-interactive, any malicious behavior can only lead to a wrong computation result. Under this situation, the malicious behavior can be reduced to input modification. Thus, from a theoretical perspective, to extend the O-PPBA protocol to satisfy the malicious model does not have much value.

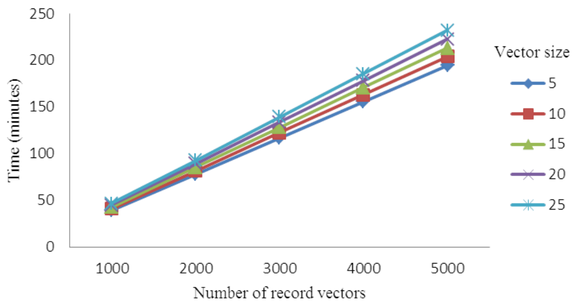
On the other hand, the covert model provides both practical value and added security guarantee. In this model, although the honest behavior of a participating party cannot be enforced, a malicious behavior is detectable. Thus, a protocol secure under the covert model is much more efficient than in the malicious model. Here we highlight the key steps involved to make the proposed O-PPBA protocol satisfy the covert model. We will adopt the dual execution strategy proposed in [25]. First, the entity outsources two encrypted copies $[D]_1$ and $[D]_2$ of D. $[D]_1$ is stored at P_1 and encrypted using the public key generated by P_2 (who has the corresponding decryption key). Similarly, $[D]_2$ is stored at P_2 and encrypted using the public key generated by P_1 (who has the corresponding decryption key). Now the proposed O-PPBA protocol can be executed twice with $([D]_1, u)$ and $([D]_2, u)$ as the respective input. Also, the roles of P_1 and P_2 are switched after the first execution. When the final outputs are inconsistent between the two executions, we know one of the parties behaved maliciously. The complexity of this modified protocol is twice as much as the original O-PPBA protocol. Again, since we assume P_1 and P_2 are cloud service providers, the proposed O-PPBA protocol, secure under the semi-honest model, is sufficient and practical.



(a) Secure squared-Euclidean distance



(b) Secure circuit comparison for $m = 5$



(c) O-PPBA execution time

Figure 1: Time complexities of secure squared-Euclidean distance, secure circuit comparison and O-PPBA for varying values of n, and m

5. PERFORMANCE EVALUATION

In this section, we discuss the performance of the O-PPBA protocol in details under different parameter settings. We look at the performance in term of the execution time. Recall that the proposed protocol is a sequential composition of three sub-protocols. We implemented each sub-protocol and measured their performance on a range of inputs.

We randomly generated synthetic datasets depending on the parameter values in consideration. The advantage of using these synthetic datasets is that we can perform a more elaborated analysis on the computation costs of the proposed protocols under different parameter settings. We randomly generated biometrics feature vectors depending on the parameter values in consideration, such as n- number of record vectors and m-vector size. We encrypted these feature vectors component-wise, using the Paillier cryptosystem [38] with a 1024-bit modulus in our experiments, and the encrypted feature vectors were stored on our machine. Based on this dataset, we performed the O-PPBA protocol. All experiments were done using a Linux machine with an Intel® Xeon® Six-Core^{TM} CPU 3.07 GHz processor and 12GB RAM running Ubuntu 12.04 LTS.

5.1 Performance of O-PPBA

To begin with, we analyze the computation costs of the Secure_Distance^e sub-protocol (Step 1 of Algorithm 3) by varying m and n. We then analyze the computation costs of the secure comparison sub-protocols (Steps 3 and 5 of Algorithm 3) by varying n because m is irrelevant in our secure comparison task.

Secure_Distance^e is implemented using Paillier encryption in C language on top of the GNU multiple precision arithmetic library (https://gmplib.org/). As shown in Figure 1(a), the computation cost grows linearly with n and m. For example, when $m = 5$, the computation time of Secure_Distance^e increases from 1.987 to 9.921 minutes when n is varied from 1000 to 5000.

We build the Secure_Comparison sub-protocol on top of FastGC [24], a Java-based framework that allows users to define boolean circuits. After the circuits are constructed, the framework encrypts the circuits, performs oblivious transfer, and evaluates the garbled/encrypted circuits. The FastGC provides is an 80-bit security level which matches the security level of the homomorphic encryption scheme that we used in our implementation. Since the size of inputs for this stage are fixed by 1024 bits modulus which are random shares of a 1024 bit number, m does not affect the performance for this stage. We evaluated the computation costs of this secure comparison by fixing $m = 5$ and varying values of n. Following from Figure 1(b), the running time of secure circuit comparison varies from 37.45 to 185.013 minutes when n is changed from 1000 to 5000 respectively. Thus, we observed that the running time of secure circuit comparison grows almost linearly with n.

Based on above results, as shown in Figure 1(c), the computation costs of O-PPBA protocol scales approximately linearly with n and m. For example, when $m = 5$, the computation time of O-PPBA increases from 39.032 to 194.934 minutes when n is varied from 1000 to 5000. Based on that, the computation time for O-PPBA is dominated by the time required to compare the encrypted distance vectors. However, when m is sufficiently large (e.g., over 100), the run time complexity of Secure_Distance^e will be greater than Secure_Compare. This is consistent with the upper bound derived in Section 4.3.1.

For $n = 5000$, $m = 25$ and the encryption key size is 1024, the communication cost is roughly 16MB. The time that takes to transmit 16MB of data is significantly less than the computation time. Therefore, this communication complexity could be ignored in the proposed protocol.

5.2 Justification of Using AH-Enc to Implement Secure_Distance

As we claimed that pure garbled circuit based implementation of O-PPBA is not efficient, here we empirically justify the proposed hybrid implementation is more efficient. Since secure multiplication (SM) is the main building block and performance bottle neck for securely computing the distance in both Euclidean and Hamming metrics, we analyze and compare the computation costs of two different implementations of SM: homomorphic encryption approach (SM-HE) and garbled circuit approach (SM-GC). We compared the running times of the two implementations for varying number of bits required to perform multiplication. Based on the results shown in Table 2, we observed that the running time of each implementation is independent from the input size, and the computation cost of SM-GC is significantly higher than that of SM-HE. For example, the computation time of multiplying two 10-bit numbers using SM-GC is 1.752 seconds whereas in SM-HE is 0.02984 second. Therefore, we adopted homomorphic encryption approach to implement Secure_Distance.

Share size	Distance size	SM-GC	SM-HE
1024	10	1.752s	0.02984s
1024	20	1.792s	0.02985s

Table 2: Garbled circuit vs. homomorphic based secure multiplication

5.3 Performance Improvement

If biometric authentication needs to be done in real time, clearly the proposed O-PPBA protocol is not practical even though it is the best two-party protocol we know of. One main advantage of the proposed protocol is that the computations of the sub-components can be parallelized. For example, at steps 1 and 2 of Algorithm 3, each secure distance computation or secure comparison are independent and can be performed at the same time. Since we assume P_1 and P_2 are cloud service providers, the O-PPBA protocol can take advantage of highly parallel computing capability. If P_1 and P_2 have n nodes available to execute O-PPBA, step 1 can be performed within a second, and step 2 can be performed a little bit over 2 seconds. Thus, the total running time would be around 5 seconds. In general, SMC-based privacy-preserving protocols are very expensive. For real-time applications like biometric authentication, using cloud is the only way to make them practical.

6. FUTURE WORK

In this paper, we proposed an outsourceable and privacy-preserving biometric authentication (O-PPBA) protocol. A main difference from the existing privacy-preserving biometric identification protocols is that the input biometric data of O-PPBA are encrypted. In addition, our protocol does not leak data access patterns, so the proposed O-PPBA protocol is more secure than the existing solutions. Furthermore, we adopted a hybrid approach to implement O-PPBA, in order to take advantages of both homomorphic encryption and garbled circuit based approaches.

Although our construction of O-PPBA is the most efficient secure two-party implementation known today, the empirical results clearly show it is still not practical. As a future work, we will implement it using a cloud to drastically improve its run time complexity. Also, we will utilize more than two parties and the secret sharing based secure multiparty computation framework [12] to produce more efficient O-PPBA protocols.

7. ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their constructive feedback. This material is based upon work supported by the Office of Naval Research under award No. N000141110256 and NSF under award No. CNS-1011984.

8. REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. *Communications of the ACM*, 53:50–58, April 2010.

[2] Y. Aumann and Y. Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. *Journal of Cryptology*, April 7 2009.

[3] H. Bay, T. Tuytelaars, and L. Gool. Surf: Speeded up robust features. In A. Leonardis, H. Bischof, and A. Pinz, editors, *Computer Vision âĂŞ ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 404–417. Springer Berlin Heidelberg, 2006.

[4] I. F. Blake and V. Kolesnikov. One-round secure comparison of integers. *Journal of Mathematical Cryptology*, 3(1):37–68, May 2009.

[5] M. Blanton and M. Aliasgari. Secure outsourced computation of iris matching. *Journal of Computer Security*, 20(2):259–305, 2012.

[6] M. Blanton and P. Gasti. Secure and efficient protocols for iris and fingerprint identification. In *Computer Security–ESORICS 2011*, pages 190–209. Springer, 2011.

[7] W. M. B.S. Manjunathi. Texture features for browsing and retrieval of image data. Aug. 1996.

[8] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina. Controlling data in the cloud: outsourcing computation without outsourcing control. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, CCSW '09, pages 85–90. ACM, 2009.

[9] C. W. C.M. Pun. Fast and robust, color feature extraction for content-based image retrieval. 2005.

[10] I. Damgård, M. Geisler, and M. Krøigaard. Efficient and secure comparison for on-line auctions. In *Proceedings of the 12th Australasian conference on Information security and privacy*, pages 416–430. Springer-Verlag, 2007.

[11] I. Damgard, M. Geisler, and M. Kroigard. Homomorphic encryption and secure comparison. *International Journal of Applied Cryptology*, 1(1):22–31, 2008.

[12] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. Smart. Practical covertly secure mpc for dishonest majority âĂŞ or: Breaking the spdz limits. In J. Crampton, S. Jajodia, and K. Mayes, editors, *Computer Security âĂŞ ESORICS 2013*, volume 8134 of *Lecture Notes in Computer Science*, pages 1–18. Springer Berlin Heidelberg, 2013.

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

[14] T. Ermakova, B. Fabian, and R. Zarnekow. Security and privacy system requirements for adopting cloud computing in healthcare data sharing scenarios. 2013.

[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] J. Garay, B. Schoenmakers, and J. Villegas. Practical and secure solutions for integer comparison. In *Proceedings of the 10th international conference on Practice and theory in public-key cryptography*, pages 330–342. Springer-Verlag, 2007.

[17] C. Gentry. Fully homomorphic encryption using ideal lattices. In *Annual ACM Symposium on Theory of Computing*, pages 169–178, Bethesda, MD, USA, 2009.

[18] O. Goldreich. *The Foundations of Cryptography*, volume 2, chapter Encryption Schemes. Cambridge University Press, 2004.

[19] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38:690–728, 1991.

[20] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 291–304, Providence, Rhode Island, U.S.A., May 6-8 1985.

[21] T. Y. H. Tamura, S. Mori. Texture features corresponding to visual perception. June 1978.

[22] K. Hashizume, D. G. Rosado, E. Fernández-Medina, and E. B. Fernandez. An analysis of security issues for cloud computing. *Journal of Internet Services and Applications*, 4(1):1–13, 2013.

[23] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih. Image indexing using color correlograms. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 762–768. IEEE, 1997.

[24] Y. Huang, D. Evans, J. Katz, and L. Malka. Faster secure two-party computation using garbled circuits. In *Proceedings of the 20^{th} USENIX Security Symposium*, page 35, August 2011.

[25] Y. Huang, J. Katz, and D. Evans. Quid-pro-quo-tocols: Strengthening semi-honest protocols with dual execution. In *IEEE Symposium on Security and Privacy*, pages 272–284, May 2012.

[26] M. S. Islam, M. Kuzu, and M. Kantarcioglu. Access pattern disclosure on searchable encryption: Ramification, attack and mitigation. In *Proceedings of the 19^{th} Annual Network & Distributed System Security Symposium (NDSS)*, February 2012.

[27] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti. Filterbank-based fingerprint matching. *Image Processing, IEEE Transactions on*, 9(5):846–859, 2000.

[28] C. K. Milkolajczyk. A performance evaluation of local descriptors. Oct. 2005.

[29] J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *Advances in*

Cryptology–EUROCRYPT 2008, pages 146–162. Springer, 2008.

[30] K. Kumar and Y.-H. Lu. Cloud computing for mobile users: Can offloading computation save energy? *Computer*, 43(4):51 –56, april 2010.

[31] G.-H. Liu and J.-Y. Yang. Content-based image retrieval using color difference histogram. *Pattern Recognition*, 46(1):188 – 198, 2013.

[32] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[33] O. G. Luo Juan. A comparison of sift, pca-sift and surf. 2009.

[34] M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, pages 245–254, Atlanta, Georgia, United States, 1999. ACM Press.

[35] A. E. Nergiz, M. E. Nergiz, T. Pedersen, and C. Clifton. Practical and secure integer comparison and interval check. In *Proceedings of the IEEE Second International Conference on Social Computing*, pages 791–799, 2010.

[36] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *IEEE Symposium on Security and Privacy*, May 2013.

[37] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich. Scifi-a system for secure face identification. In *IEEE Symposium on Security and Privacy*, pages 239–254. IEEE, 2010.

[38] P. Paillier. Public key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology - Eurocrypt '99 Proceedings, LNCS 1592*, pages 223–238, Prague, Czech Republic, May 2-6 1999. Springer-Verlag.

[39] I. D. R.M. Haralick, K. Shangmugam. Textural feature for image classification. Nov. 1973.

[40] J. J. Rodrigues, I. de la Torre, G. FernÃąndez, and M. LÃşpez-Coronado. Analysis of the security and privacy requirements of cloud-based electronic health records systems. *Journal of medical Internet research*, 15(5), 2013.

[41] M. D. Ryan. Cloud computing security: The scientific challenge, and a survey of solutions. *Journal of Systems and Software*, 2013.

[42] J. P. S. Belongie, J. Malik. Shape matching and object recognition using shape contexts. Apr. 2002.

[43] A.-R. Sadeghi, T. Schneider, and I. Wehrenberg. Efficient privacy-preserving face recognition. In *Information, Security and Cryptology–ICISC 2009*, pages 229–244. Springer, 2010.

[44] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612 – 613, November 1979.

[45] E. Shen, E. Shi, and B. Waters. Predicate privacy in encryption systems. In *Theory of Cryptography*, pages 457–473. Springer, 2009.

[46] S. W. S.M. Lee, J.H. Xin. Evaluation of image similarity by histogram intersection. Aug. 2005.

[47] M. B. W. Burger. Principles of digital image processing: core algorithms. 2009.

[48] B. Wang, S. S. Chow, M. Li, and H. Li. Storing shared data on the cloud via security-mediator. In *International Conference on Distributed Computing Systems-ICDCS 2013*, 2013.

[49] R. S. Y. Ke. Pca-sift: a more distinctive representation for local image descriptors. 2004.

[50] A. C. Yao. How to generate and exchange secrets. In *Proceedings of the 27th IEEE Symposium on Foundations of Computer Science*, pages 162–167. IEEE, 1986.

[51] A. C.-C. Yao. How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167. IEEE, 1986.

[52] H.-J. Yu, H.-S. Lai, K.-H. Chen, H.-C. Chou, J.-M. Wu, S. Dorjgochoo, A. Mendjargal, E. Altangerel, Y.-W. Tien, C.-W. Hsueh, et al. A sharable cloud-based pancreaticoduodenectomy collaborative database for physicians: Emphasis on security and clinical rule supporting. *Computer methods and programs in biomedicine*, 2013.