 doi:10.20944/preprints202007.0658.v1

Article

A Privacy-Preserving Fully Homomorphic Encryption and Parallel Computation Based Biometric Data Matching

Ferhat Ozgur Catak [1], Sule Yildirim Yayilgan [1] and Mohamed Abomhara [1]

[1] Department of Information Security and Communication Technology, NTNU Norwegian University of Science and Technology, 2815 Gjøvik, Norway; ferhat.o.catak@ntnu.no (F.O.C.), sule.yildirim@ntnu.no (S.Y.Y), mohamed.abomhara@ntnu.no (M.A.)

* Correspondence: ferhat.o.catak@ntnu.no

‡ These authors contributed equally to this work.

Abstract: One of the most reliable methods of authentication used today is biometric matching. This authentication process, which is done by using biometrics information such as fingerprint, iris, face, etc. is used in many application areas. Authentication at border gates is one of these areas. However, some restrictions have been introduced to storing and using such data, especially with the General Data Protection Regulation (GDPR). The main goal of this work is to find the practical implementation of fully homomorphic encryption-based biometric matching in border controls. In this paper, we propose a biometric authentication system based on hash expansion and fully homomorphic encryption features, considering these restrictions. One of the most significant drawbacks of the homomorphic encryption method is the long execution time. We solved this problem by executing the matching algorithm in parallel manner. The proposed scheme is implemented as proof-of-concept in the SMILE, and its advantages in privacy preservation has been demonstrated.

Keywords: biometric matching; fully homomorphic encryption; privacy-preserving techniques

1. Introduction

Biometrics matching systems are used for the identification of persons by using fingerprints, iris, and palm, etc. The biometrics matching algorithm uses a plain version of the sensitive biometrics data to achieve high-performance matching for accurate identification [1,2]. Today, many organizations, especially border control police, use such identification system to verify travelers [3]. The main drawbacks of these biometrics datasets are privacy and security concerns. All of these methods require the full right to access private and confidential data to create a feature template or data matching model [4].

Moreover, through applying various anonymization techniques to confidential data sets to protect privacy, an adversary can still retrieve the confidential data via several methods [5–7]. Imagine a scenario in which travelers want to approve their identity in border control. In order to prove their identity, the travelers have to share their sensitive biometrics data with the border control police. Due to some vulnerabilities in the identity control system, there is a possibility that the biometric data of travelers may be exposed. In such a case, it is essential to find a new privacy-based matching algorithm for identity matching that can run on confidential biometrics data without retrieving the private data.

In biometrics, a matching algorithm consists of two sequential steps: (1) feature template extraction and storing and (2) new biometric feature template matching with the previously stored biometrics data [8]. The feature template extraction and storing phase of a biometrics matching algorithm creates biometric data representation vector $\mathbf{x}$ in enrolment from a user $\mathcal{U}$. In the data validation stage, a new

 doi:10.20944/preprints202007.0658.v1

biometric data $\mathbf{y}$ is extracted with claimed user identity $\mathcal{U}$, and then the system tries to find the how similar both vectors $\mathbf{x}, \mathbf{y}$ using a distance metrics d are. The traditional biometrics matching algorithms require direct access to plain data, which poses security and privacy risks, as mentioned above.

In this research, we propose a privacy-preserving biometric data matching model building method based on a fully homomorphic encryption scheme that constructs the feature template from encrypted biometrics data for a cloud system.

The contributions of our work are as follows:

- A fully homomorphic encryption scheme based biometrics data storing and matching system is proposed, and thus a privacy-preserving matching model is achieved.
- A cloud system does the computation of the distance metrics. Thus the model is very power efficient for mobile devices.
- We applied a hash expansion based secondary privacy-enhancing technique.
- Parallel execution techniques are applied to reduce the execution time of homomorphic encryption-based matching.

Our paper is organized as follows: in Section 2, we briefly introduce some of the related works for privacy-preserving training. In Section 3, we describe the Microsoft SEAL, fully homomorphic encryption, parallel computation. In Section 4, we describe the proposed secure biometric data storing and data matching method. Section 5 evaluates our proposed model. Section 6 concludes the paper.

2. Related Work

In this section, we review the existing works that focus privacy-preserving biometrics matching methods. We highlight individual differences between our proposed matching model and the current research. Encryption methods can be employed to address the security and privacy concerns in biometric data matching [9,10].

Gunasinghe et al. [11] proposed a machine learning-based classification technique. The authors mainly focus on privacy-preserving and user-centric authentication to overcome the drawbacks of the existing biometrics-based authentication models such as storing and transmitting the biometrics data, sensitive data sharing between service providers. Their approach preserves privacy by using zero-knowledge proof of knowledge and a secret provided by the user. They state that they must improve performance optimization for mobile devices.

Im et al. [12] proposed a face-feature based authentication system for mobile phones. They use Yao's garbled circuit based secure-multi party computation to secure the biometric data. The main difference between the previous approach is the usage of client-server based ResNet face recognition model. According to their experimental results, their solution has an Equal Error Rate (EER) of 3.04% with 1.3 seconds matching duration with two publicly available face datasets. Their approach is more complicated than our approach in terms of biometric matching.We use distance metrics to match the fingerprint biometrics.

In another study, the authors proposed a new system based on biometric data encryption and matching algorithms, and they introduced perturb items in every biometric data [13]. They converted the matrix multiplications to vector-matrix multiplications.

Zhang et al. [14] proposed scalable, efficient,privacy-preserving fingerprint authentication using minutiae representation based on Yao's classic Garbled Circuit (GC) protocol. The optimized implementation achieved in their study requires 210 KB for storage.

In another study, Tian et al. [15] proposed a biometric data based remote user authentication system for a cloud server. Their approach can be applied to the honest-but-curious server in an anonymous and unlinkable manner.

Hahn et al. [16] proposed how an attack enrolls fake biometric data and then manipulates them to recover encrypted an identification request in CloudBI. In their paper, authors define an attack scenario at enrollment stage. In this stage, the attacker randomly generate 8-bit integers and send them

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

to the cloud server to extract the recorded biometrics data. Next, they offered a useful security patch to CloudBI, which is secure against enrollment-level attackers. They introduce more randomness in encrypting the biometric data to find the distance between two biometric data.

Yang et al. [17] proposed several private credential management work models under different trust models between a user and an external party. Their main advantageous parts are feature agnosticism, privacy-enhanced biometric template, and outsourced authentication.

3. Preliminaries

In this section, we briefly introduce preliminary information about the fully homomorphic encryption concept, parallel computation methods, and biometric data matching.

3.1. Fully Homomorphic Encryption

Homomorphic encryption schemes allow arithmetic operations on plaintexts to be performed on their corresponding ciphertexts without exposing the plaintexts when data are shared between two or more individuals as it allows arithmetic operations with ciphertexts such as addition, multiplication. One can say that a public-key encryption scheme such as Paillier is additively homomorphic if, given two encrypted data such as $[\![x]\!]$ and $[\![y]\!]$, there exists a public-key summation operation $\oplus$ such that $[\![x]\!] \oplus [\![y]\!]$ is an encryption of the plaintext of $x + y$. The formal explanation is that an encryption scheme is additively homomorphic if for any private key, public key (key_{priv}, key_{pub}), the plaintext space $\mathcal{P} = \mathbb{Z}_N$ for $x, y \in \mathbb{Z}_N$.

$$Dec\Big(Enc\left(x + y \ mod \ N; key_{pub}\right); key_{priv}\Big) = x + y \tag{1}$$

Also, one can say that a public-key encryption scheme such as RSA is multiplicative homomorphic if, given two encrypted data such as $[\![x]\!]$ and $[\![y]\!]$, there exists a public-key multiplicaltion operation $\otimes$ such that $[\![x]\!] \otimes [\![y]\!]$ is an encryption of the plaintext of $x \times y$. The formal explanation is that an encryption scheme is multiplicative homomorphic if for any private key, public key (key_{priv}, key_{pub}), the plaintext space $\mathcal{P} = \mathbb{Z}_N$ for $x, y \in \mathbb{Z}_N$.

$$Dec\Big(Enc\left(x \times y \ mod \ N; key_{pub}\right); key_{priv}\Big) = x \times y \tag{2}$$

The homomorphic encryption schemes run only on integer numbers. Therefore, the proposed protocols handle only integers, although biometrics matching is typically applied to continuous data. In the case of an input data set with real numbers in the protocol, we need to map floating-point input data vectors into the discrete domain with a scaling function.

Let $ConvertToInteger : \mathbb{R}^m \rightarrow \mathbb{Z}^m$ be the corresponding function that multiplies its floating point number argument by an exponent $(K : 2^K)$ and then rounds it to the nearest integer value and thus supports finite precision. Eq. 3 shows the conversion function:

$$\hat{\mathbf{x}} \leftarrow ConvertInteger(\mathbf{x}) \ \ where \ \mathbf{x} \in \mathbb{R}^m, \hat{\mathbf{x}} \in \mathbb{Z}^m \tag{3}$$

3.2. Parallel execution

The main drawback of the homomorphic encryption algorithms is time complexity. To reduce the execution time of biometric matching, we applied block-stripped decomposition to reduce the biometric matching execution time [18]. In this study, we used parallel computation methods in order to overcome this problem. We used the Euclidean distance to match the two biometric data. Consider

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

the problem of computing the distance of two vectors $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)} \in \mathbb{R}^n$ using the Euclidean distance. The standard Euclidean distance metric is defined as:

$$\begin{aligned} d\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right) &= \sqrt{\sum_{i=1}^{n} \left(\mathbf{x}_i^{(1)} - \mathbf{x}_i^{(2)}\right)^2} \\ &= \sqrt{\left(x_1^{(1)} - x_1^{(2)}\right)^2 + \cdots + \left(x_n^{(1)} - x_n^{(2)}\right)^2} \end{aligned} \tag{4}$$

We can apply Block-Striped Decomposition [19] to find the overall distance between two vectors for parallel execution of the Euclidean distance computation. Our parallel execution solution is to divide the whole distance computation into T smaller computations, where T is the number of CPU cores available on a single computer. Accordingly, the vectors $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ can be partitioned into T sub-vectors. As a result, a possible approach for parallelizing the Euclidean distance calculation is to define the calculation task as the problem of calculating the distance between two vectors. To carry out all the necessary calculations of each subtask, must contain sub-partition of vector $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$. Figure 1 shows the distributed and parallel computation of the distance calculation using several CPU cores. The data fusion will find the `sqrt` of the total value for each vector on each CPU core T.

$$\begin{gathered} \underbrace{\left(x_1^{(1)} - x_1^{(2)}\right)^2 + \left(x_2^{(1)} - x_2^{(2)}\right)^2}_{\text{Processor-1 execution}} \\ + \\ \underbrace{\left(x_3^{(1)} - x_3^{(2)}\right)^2 + \left(x_4^{(1)} - x_4^{(2)}\right)^2}_{\text{Processor-2 execution}} \\ + \\ \cdots \\ + \\ \underbrace{\left(x_{n-1}^{(1)} - x_{n-1}^{(2)}\right)^2 + \left(x_n^{(1)} - x_n^{(2)}\right)^2}_{\text{Processor-T execution}} \end{gathered}$$

Figure 1. The fully homomorphic encryption based Euclidean distance of length n is being computed in parallel manner using T processors.

We can define the expanded form of Euclidean distance as each element of both vectors is subtracted from each other, and the result is squared. Since homomorphic encryption is used in this study, all calculations are done in the encrypted domain. In homomorphic encryption, even if a number cannot be squared, the square of a number can be obtained by with multiplying the number with itself The computations carried out by each processor denoted by d_p as shown in Equation 5.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

$$
\begin{aligned}
d_{P_1}([\![\mathbf{x}^{(1)}_{sub_1}]\!],[\![\mathbf{x}^{(2)}_{sub_1}]\!]) &= \left([\![x^{(1)}_1]\!]-[\![x^{(2)}_1]\!]\right)*\left([\![x^{(1)}_1]\!]-[\![x^{(2)}_1]\!]\right)\\
&+\left([\![x^{(1)}_2]\!]-[\![x^{(2)}_2]\!]\right)*\left([\![x^{(1)}_2]\!]-[\![x^{(2)}_2]\!]\right)\\
d_{P_2}([\![\mathbf{x}^{(1)}_{sub_2}]\!],[\![\mathbf{x}^{(2)}_{sub_2}]\!]) &= \left([\![x^{(1)}_3]\!]-[\![x^{(2)}_3]\!]\right)*\left([\![x^{(1)}_3]\!]-[\![x^{(2)}_3]\!]\right)\\
&+\left([\![x^{(1)}_4]\!]-[\![x^{(2)}_4]\!]\right)*\left([\![x^{(1)}_4]\!]-[\![x^{(2)}_4]\!]\right)\\
&\vdots\\
d_{P_T}([\![\mathbf{x}^{(1)}_{sub_T}]\!],[\![\mathbf{x}^{(2)}_{sub_T}]\!]) &= \left([\![x^{(1)}_{n-1}]\!]-[\![x^{(2)}_{n-1}]\!]\right)*\left([\![x^{(1)}_{n-1}]\!]-[\![x^{(2)}_{n-1}]\!]\right)\\
&+\left([\![x^{(1)}_n]\!]-[\![x^{(2)}_n]\!]\right)*\left([\![x^{(1)}_n]\!]-[\![x^{(2)}_n]\!]\right)
\end{aligned}
\tag{5}
$$

In the data fusion stage, all calculations made by each processor are summed.

$$
d([\![\mathbf{x}^{(1)}]\!],[\![\mathbf{x}^{(2)}]\!]) = \sqrt{\sum_{i\in T} d_{p_i}} \tag{6}
$$

4. System model

The main goal of this work is to find the practical implementation of fully homomorphic encryption-based biometric matching in border controls, shown in Figure 5-6. The biometric matching system consists of two phases: (1) enrollment, and (2) matching stage. The identification system performs the database query during the enrollment stage of the model by the acquisition of unique fingerprint images. The fingerprint images are recorded in the cloud biometric database with SHA256 based hash expansion and The Brakerski/Fan-Vercauteren (BFV) based somewhat homomorphic encryption scheme [20]. In the matching stage, the model identifies the person based on the query fingerprint images by matching the distance metrics of fingerprint features of the enrollment images, saved in the biometric template database. Figure 2 shows the proposed privacy-preserving biometric matching system.

The main drawback of the homomorphic encryption algorithms is time complexity. To reduce the execution time of biometric matching, we applied block-stripped decomposition to reduce the biometric matching execution time.

4.1. Security Model

In this work, the aim is to enable a mobile device to authenticate a traveler with a public cloud system to cooperatively match the biometric data without retrieving the traveler's own confidential biometric data. Our primary assumption is that the input biometric data is matched with the user's feature template, which has been stored at the enrollment stage in the cloud. Our other assumption is that both mobile device and cloud system follow our protocol; this is called a semi-honest security model [21,22].

4.2. Enrollment

Fingerprint enhancement techniques are necessary steps for the feature extraction and fingerprint matching process. We applied several fingerprint enhancement techniques, including image rotation and edge detection. The first stage is image rotation representing all fingerprint images in the same direction, then, we implemented the Harris corner based feature extraction for fingerprint detection. Figure 3 shows the Harris corner detection for an example fingerprint. The original fingerprint images are cropped and resized to 30×28 pixels for the actual representation.

The extracted corners are transformed into the feature vectors and applying them to the binarization technique with a threshold value. Our feature vector contains only 0s and 1s to represent a feature template for the fingerprint images, denoted as $\mathbf{x} \in \mathbb{R}^n$.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

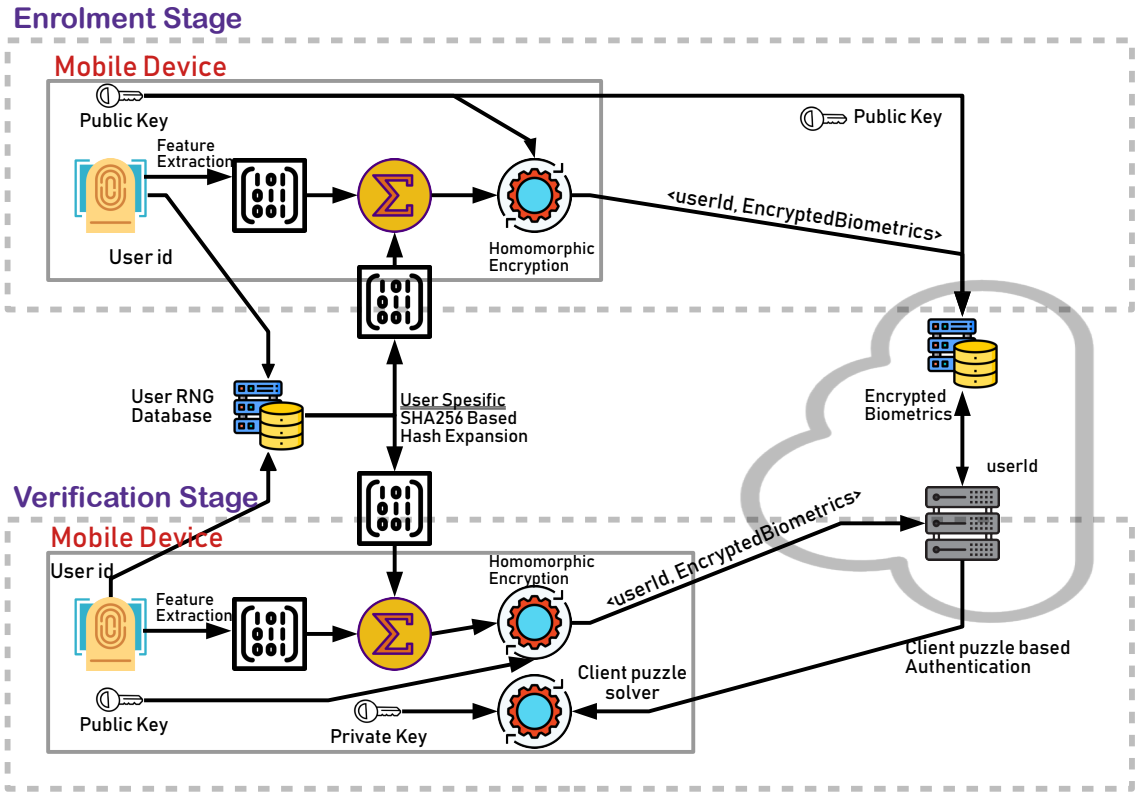

Figure 2. The system model of the proposed privacy-preserving biometric matching system

Figure 5 shows the general architecture of the enrollment stage. We applied the random number generator (RNG) based hash expansion and fully homomorphic encryption schemes to biometrics data in order to enhance privacy.

During the enrollment stage, the user $\mathcal{U}$ who wants to register himself to the border control system provides his biometrics data including fingerprint, face, and iris, from which a feature extracted representation of $\mathbf{x} \in \mathbb{R}^n$ using a mobile device is generated. The RNG generates a unique random number r for the user $\mathcal{U}$, transmits it to the mobile device, and stores it in its internal RNG database. In order to enhance the privacy and to prevent data leakage of private biometrics data of user $\mathcal{U}$, we applied hash expansion technique. A random number for each user in the biometrics control system is created uniquely using the RNG. The RNG generated random number r will be expanded as the same size of feature template $\mathbf{x}$, then the final perturbation vector $\mathbf{p}$ is created. Figure 4 shows the proposed hash expansion technique.

Algorithm 1 explains the detailed steps of the RNG-based random vector generation. The mobile device then adds these two vectors to increase the privacy protection before encryption of the input biometrics data. We can represent this transformation with a function f and initial random key for user $\mathcal{U}$ as;

$$\mathbf{x}_{per} \leftarrow f(\mathbf{x}, r_{\mathcal{U}}) \tag{7}$$

where f is transformation function, $\mathbf{x}$ is feature template and $r_{\mathcal{U}}$ initial random number for user $\mathcal{U}$.

$$\mathbf{x}_{per} = \mathbf{x} + \mathbf{p} \tag{8}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

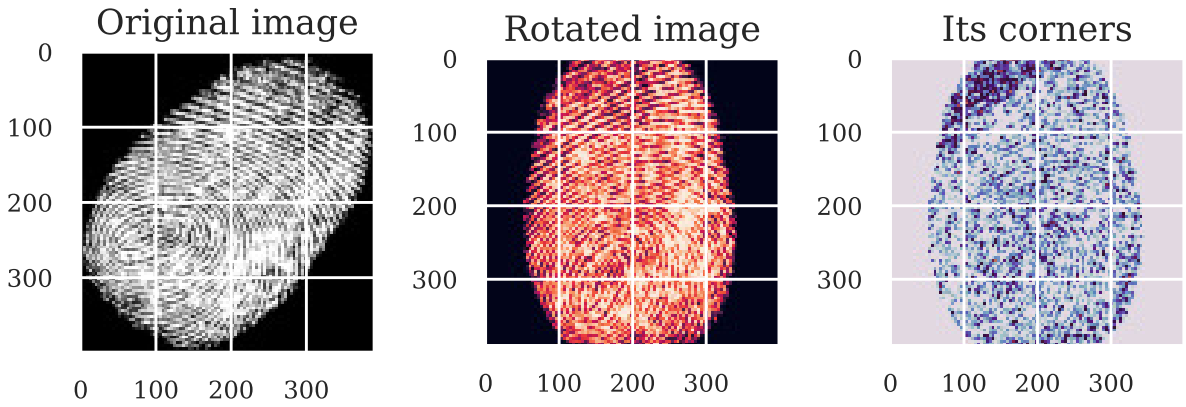

Figure 3. Harris corner based image edge detection for feature extraction of fingerprint images

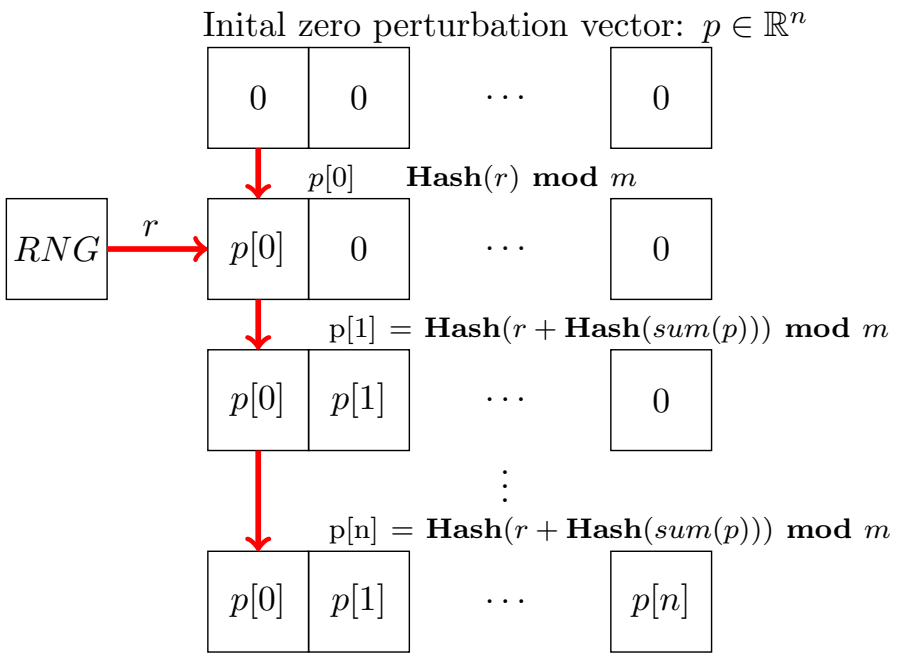

Figure 4. Hash-expansion method

This perturbation technique allows us to calculate the distance (i.e., biometrics template matching) in the transformed version in the same value;

$$d(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = d(\mathbf{x}^{(1)}_{per}, \mathbf{x}^{(2)}_{per}) \tag{9}$$

where d is the distance function. The perturbation enables us to increase the privacy protection of the users' biometrics.

The second step in the enrollment stage is encryption. The encryption stage starts with the key generation. The mobile device creates public/private key pairs and protects its encryption key internally. The encryption key is not accessible to external users or systems and can only be accessed by itself. The mobile device creates public/private key pairs to protect the perturbated version of the feature template vector $\mathbf{x}_{per}$. The perturbated template vector is encrypted ($[\![\mathbf{x}_{per}]\!]$) using the public encryption key and transmitted to the encrypted cloud biometrics database with user id $\mathcal{U}$. The database stores an encrypted template.

Figure 5 shows the enrollment stage of the proposed system. As shown in the figure, there are three components in this stage; the mobile device, RNG, and the encrypted cloud biometrics database.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

Algorithm 1 RNG seed based perturbation vector creation

Input: RNG seed r, vector size n, **mod** p

/ create zero vector rng_gen* $\in \mathbb{R}^n$ **/*
rng_gen = zeros(1,n)
/ the first item is the SHA256 of r */*
rng_gen[0] = **Hash**(r) **mod** p
LOOP Process
for $(i = 1; i < n; i++)$ **do**
  rand_total = 0.0 {*/* temporary variable for previous items addition */*}
  for $(j = 0; j < i; j++)$ **do**
    rand_total += rng_gen[j] **mod** p
    / temporary variable for previous items addition */*
  end for
  rng_gen[i] = **Hash**(r + **Hash**($rand\_total$)) **mod** p
  / Add r and rand_total, then get SHA256 value */*
end for
return *rng_gen*

4.2.1. Secrecy of perturbation

If an attacker has access to n random numbers, then it is impracticable to produce the next $(n+1)$th random number. If the random number generator is protected, then it is impossible to get the following number. The attacker has to have the initial number of $r_{\mathcal{U}}$, which is assigned and saved in the generator. Moreover, the hashing function prevents the attacker from extracting the previous random number. All hash methods are a one-way function. It is impossible to obtain the original data from the hash value.

4.3. Matching

During the matching stage, the mobile device executes the following mechanism: (1) user's biometrics data is extracted into representation vector $\mathbf{y}$, (2) initial random number $r_{\mathcal{U}}$ for user $\mathcal{U}$ that is stored in the enrollment stage is obtained from random number generator, (3) the perturbation vector $\mathbf{p}_{\mathcal{U}}$ are generated using Algorithm 1, (4) the biometrics template $\mathbf{y}$ is perturbated using perturbation vector $p_{\mathcal{U}}$, (5) perturbated biometrics data vector $\mathbf{y}_{per}$ is encrypted using device's own public encrypted key key_{pub}, (6) the obtained encrypted and perturbated biometrics data $[\![\mathbf{y}_{per}]\!]$ with claimed user identity $\mathcal{U}$ and key_{pub} are sent to cloud biometrics database.

The cloud biometrics database (server) calculates the distance score of the both biometrics data $d([\![\mathbf{x}_{per}]\!], [\![\mathbf{y}_{per}]\!], key_{pub})$ using the Euclidean distance ($\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$). The result of this calculation is in the encrypted domain, and the cloud server cannot get the plain version of the distance value. The decryption key required for biometric matching is only available on the mobile client. In this study, the server does not trust the mobile client in the same way. Therefore, instead of sending distance metrics directly to the mobile client, a client puzzle is added to the distance metrics which is shown in Figure 7. Then the server transmits it to the mobile client to protect the distance from manipulation.

In order to build a client puzzle, two random number r_0 and r_1 are created and encrypted using pub_{key}, and the distance metrics are transformed using Equation 10.

$$[\![puzzle]\!] = [\![distance]\!] * [\![r_0]\!] + [\![r_1]\!] \tag{10}$$

The server transmits the encrypted puzzle value to the mobile device. The mobile device decrypts the puzzle value and transmits it into the cloud server. The cloud server gets the matching score using Equation 11.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

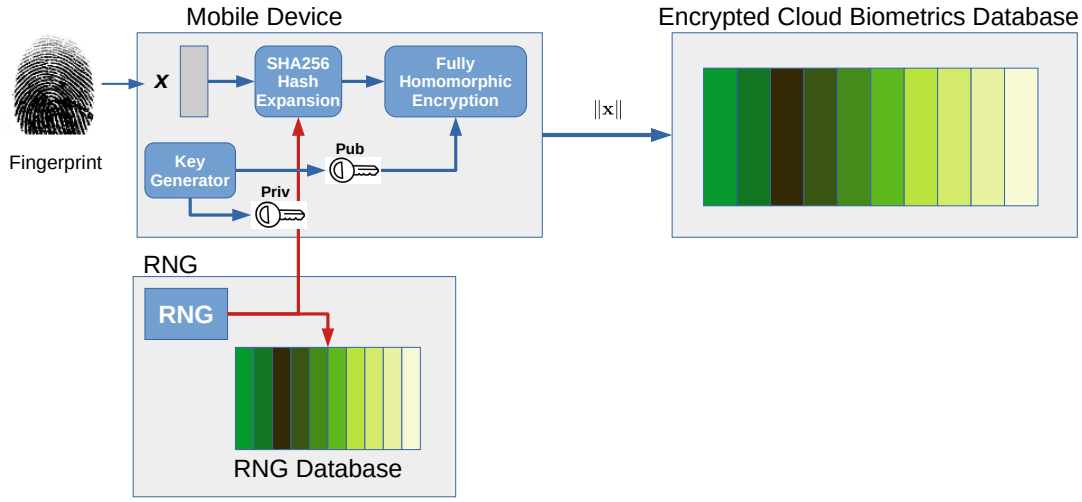

Figure 5. In enrollment stage, the mobile device expands the biometric feature vector **x**, encrypt the hashed feature template with the Key_{pub} and transmits the vector to the encrypted cloud database with the user id c.

$$match = 1 - \frac{puzzle - r_0}{r_1} \tag{11}$$

Figure 6 shows the sequence diagram of the matching phase.

5. Results

The main goal of this work is to explore the practical implementation of a fully homomorphic encryption scheme based biometric data matching for border controls. This process should be completed in less than 2 seconds. In the experimental results section, we will show the execution results in both sequential and parallel implementation of the proposed method. As explained before, the biggest drawback of all fully homomorphic encryption schemes is execution time. We applied the image reduction method for sequential and parallel implementations of the Euclidean distance calculation to increase the time performance of the proposed method.

To evaluate the experimental results of an encrypted biometric matching system, we have implemented our model using Python language.

5.1. Data preprocessing

In this work, we used the Sokoto Coventry Fingerprint Dataset (SOCOFing) biometric fingerprint database to show our work efficiency [23]. Figure 8 shows the example fingerprints images. Each fingerprint picture is recorded in 96×103 dimension and grayscale format. These fingerprint images have been converted from matrix format to vector format ($\mathbb{R}^{96\times103} \rightarrow \mathbb{R}^{9888}$), and the final feature vector for each fingerprint images contains 6776 features after fingerprint enhancement techniques applied.

The high-dimensional dataset is one of the highest computation costs in biometric data matching processes. Although parallel calculation methods are used in this study, total computation time can be reduced by applying size reduction methods. We resize the fingerprint images to reduce the dimensions.

Although there are some other dimensionality reduction techniques such as principal component analysis (PCA), we just resized all fingerprint images then convert them into a compressed vector

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

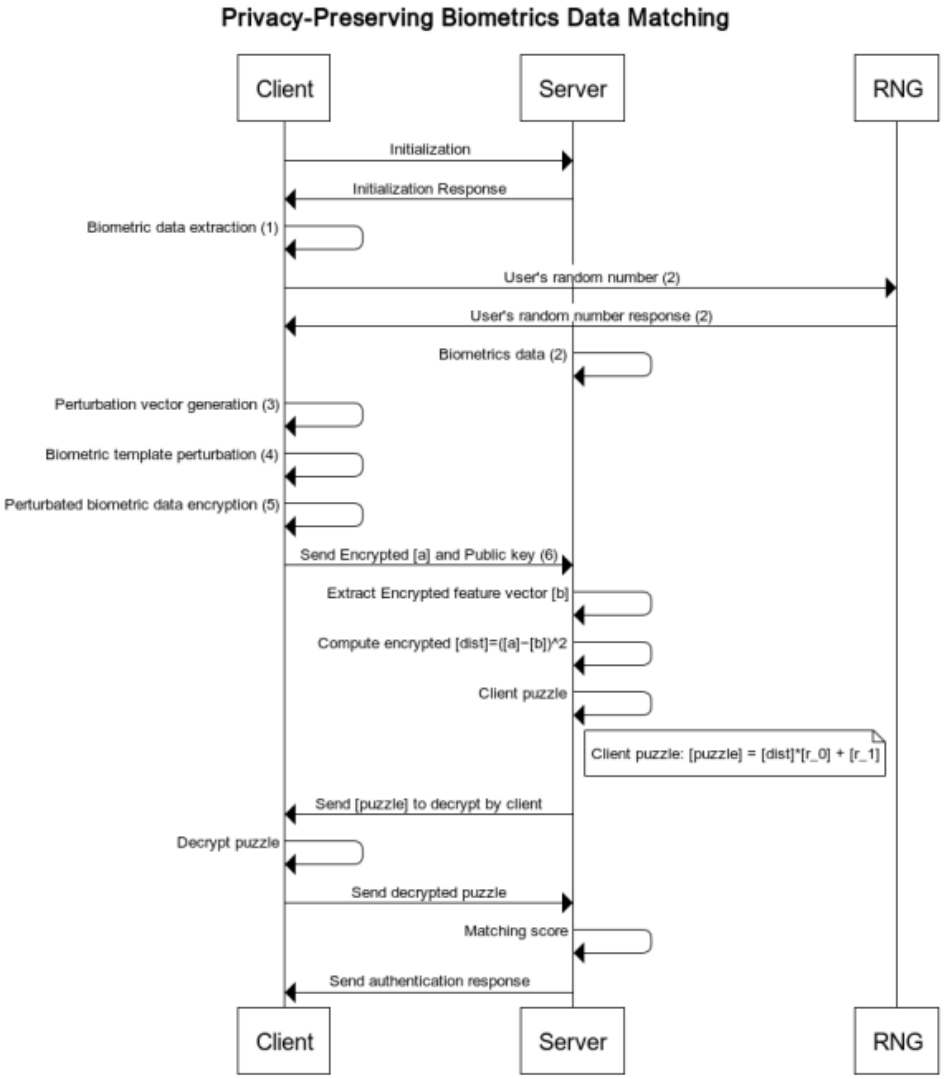

Figure 6. Sequence diagram of the privacy-preserving biometrics matching

representation. The Harris corner detection method is applied [24] to find the corners. The function runs the Harris corner detector on the image. For each pixel (x, y), it calculates a 2×2 gradient covariance matrix $M^{(x,y)}$ over a `blockSize` $\times$ `blockSize` neighborhood. Then, it computes the following characteristic:

$$\texttt{dst}(x, y) = \det M^{(x,y)} - k \cdot \left(\mathrm{tr} M^{(x,y)}\right)^2 \tag{12}$$

Corners in the image can be found as the local maxima of this response map [1].

After these operations are applied, the matrix size of each fingerprint image file is 30×28. The size of the vector created after flattening the matrix is 840. Thus, the original vector length decreases from 9888 to 840. As a result, the newly created vector's is 1 in 12 of the initial vector, reduces the size of the encrypted data, and consequently shorten the calculation time.

We applied the BFV based implementation of Microsoft SEAL library to homomorphically encrypted representation, ($[\![\mathbf{x}]\!]$), of our feature template of the fingerprint image $\mathbf{x}$. The Microsoft SEAL library deals with the plaintexts that are abstract. According to the SEAL implementation of

[1] https://docs.opencv.org/2.4/modules/imgproc/doc/feature_detection.html?highlight=cornerharris

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

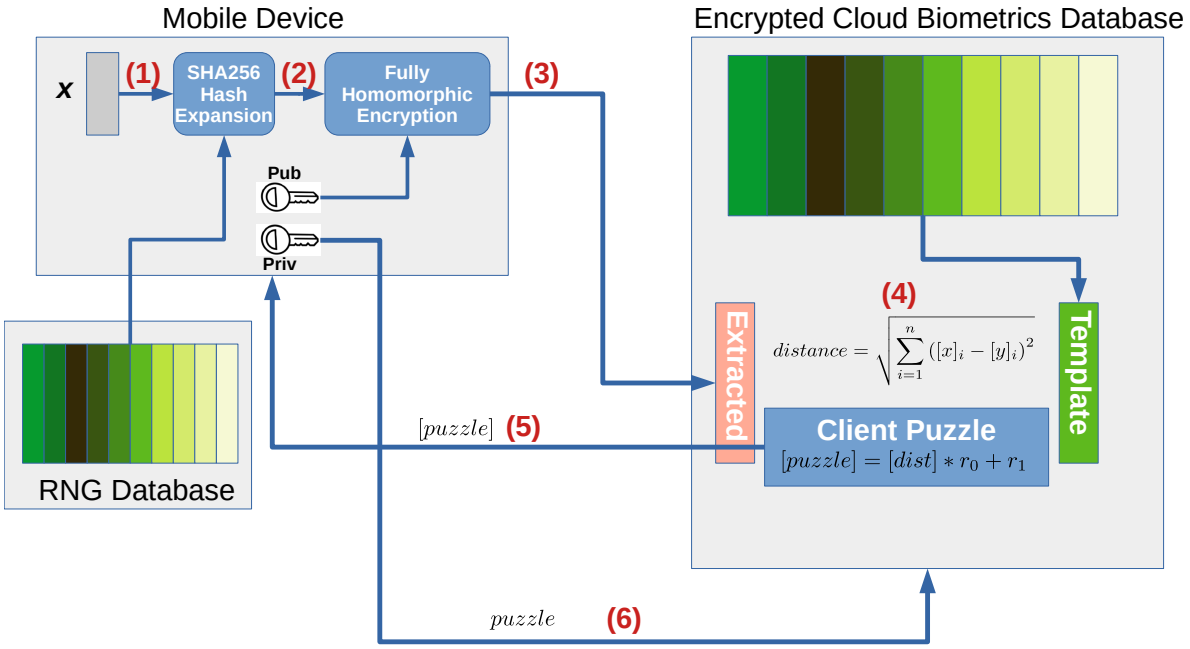

Figure 7. The overall privacy-preserving biometrics data matching process. Accordingly, (1) mobile device extracts user's biometrics data feature, (2) the feature vector is expanded using RNG database, (3) expanded vector is encrypted and transmitted to the cloud server, (4) the server gets user's template vector that is stored in enrollment stage and calculates the distance in the encrypted domain, (5) server creates an encrypted client puzzle and send it to the mobile device, (6) the mobile device decrypts and sends the plain value to the server.

BFV, the plaintext space is a polynomial ring over a prime number ring. Thus, plaintext values are polynomials which have coefficients with integer value modulo and a prime number. The library has some critical parameters; plaintext modulus (p), and polynomial modulus (m). The parameter p is the prime number for the coefficients. m is the degree of the irreducible polynomial $x^m + 1$.

Each ciphertext has a connected noise resource, and with each arithmetic operation, the noise resources shorten. Noise is defined to be the amount that must be rounded away correctly for decryption to obtain. A small fraction is not very beneficial to work with. If n is the size of the noise in a ciphertext, we define the noise budget of the ciphertext to be $log2(2n)$. With this definition, every ciphertext has a positive noise budget, and every operation spends a portion of this budget. Once the noise budget approaches 0, the ciphertext becomes undecryptable [25]. The noise influence of addition arithmetic operation is much less than the one of multiplications arithmetic operation. Also, spending noise resources is not a great approach. Thus, spending noise resources makes ciphertexts indecipherable.

5.2. Sequential Fully Homomorphic Encryption

In sequential matching, the system calculates the matching score without splitting the feature vector. Instead, the system calculates the Euclidean distance metric using all sub-vectors on just one CPU to match two biometrics data input. We want to show the time execution improvement on the proposed parallel matching protocol with the sequential approach. Other critical parameters that affect the execution time are polynomial coefficient modulus and security level equivalent in AES (128,192 or 256). Figure 9 shows the execution time of two biometrics data matching process in seconds. The execution duration increases exponentially in time, with the increase in both parameters. The execution time is increasing with the increasing polynomial coefficient modulus and AES security level as expected. As can be seen from the figure, we observed that the value of plaintext modulus, p does

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

Figure 8. Sokoto Coventry Fingerprint Dataset examples.

not change the execution time. The most determining parameter of execution time in the encrypted domain is the polynomial modulus, m.

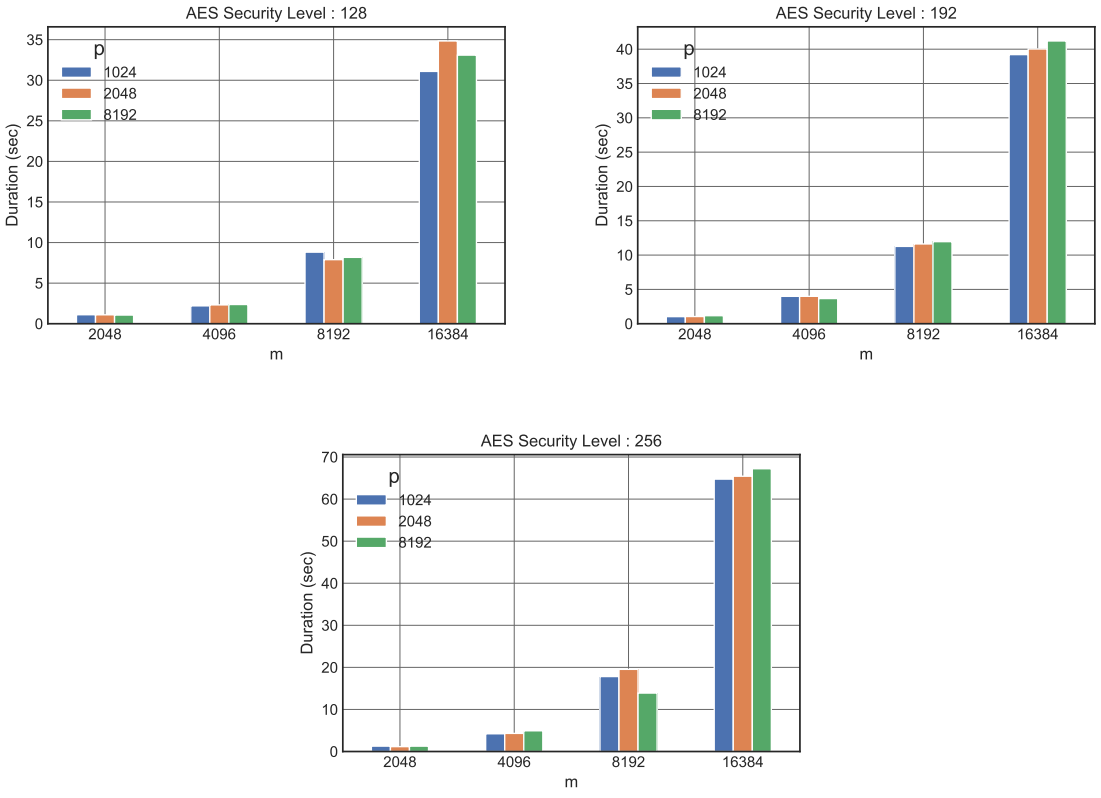

Figure 9. Biometric data matching execution time in seconds for polynomial coefficient modulus and security level equivalent in AES

Table 1 shows the execution time of the sequential privacy-preserving biometric matching model with different m, p, and AES security levels parameters. It is obvious that the first parameter, which has the highest effect on time, is polynomial modulus m, and the second parameter is the AES-Security level. For example, if m is 2048, the change of p value does not cause a significant change in execution time of AES-256 (1.156197 - 1.274226 secs) or AES-192 (1.032180 - 1.169023 secs) or AES-128 (1.065900 - 1.106724 secs). On the other hand, if the parameter m is changed from 2048 to 4096, the execution time of the security levels are quadrupled. The AES security levels again affect the calculation time. For

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

example, if m is 16384, p is 8192, execution times are 67.213099 for AES-256, 41.198113 for AES-192 and 33.093486 for AES-128. As it is seen, if the security level increases, the execution time increases.

Table 1. Sequential privacy-preserving biometrics matching method execution time with differet m, p and AES security levels.

		Duration		
m	**p**	**AES-256**	**AES-192**	**AES-128**
2048	1024	1.274226	**1.032180**	1.106724
2048	2048	1.156197	1.046678	1.097087
2048	8192	1.269185	1.169023	1.065900
4096	1024	4.212071	3.990288	2.199798
4096	2048	4.305351	3.991871	2.319643
4096	8192	4.921499	3.665943	2.366751
8192	1024	17.816331	11.266307	8.815502
8192	2048	19.527792	11.623311	7.893302
8192	8192	13.887665	11.956026	8.177011
16384	1024	64.744956	39.209209	31.088445
16384	2048	65.436099	40.030898	34.840504
16384	8192	**67.213099**	41.198113	33.093486

5.3. Parallel Fully Homomorphic Encryption

Figure 10 shows the speed up on the AES security level over the polynomial coefficient modulus on the biometrics dataset. To asses the effectiveness of the parallel matching algorithm, the time is measured with varying modulus size. As can be seen from the figure, the system achieved performance improvement in matching time. The system achieves a non-linear speed up as the number of polynomial coefficients modulus increases. For example, if the m = 2048, p = 1024 and AES-256 parameters in Table 1 are used, the execution time is 1.274226, while the execution time with the same parameters in Table 2 is 0.126655. Likewise, in the case of m = 16384, p = 8192 and AES-256, the execution time decreases from 67.213099 to 6.849738.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 27 July 2020 doi:10.20944/preprints202007.0658.v1

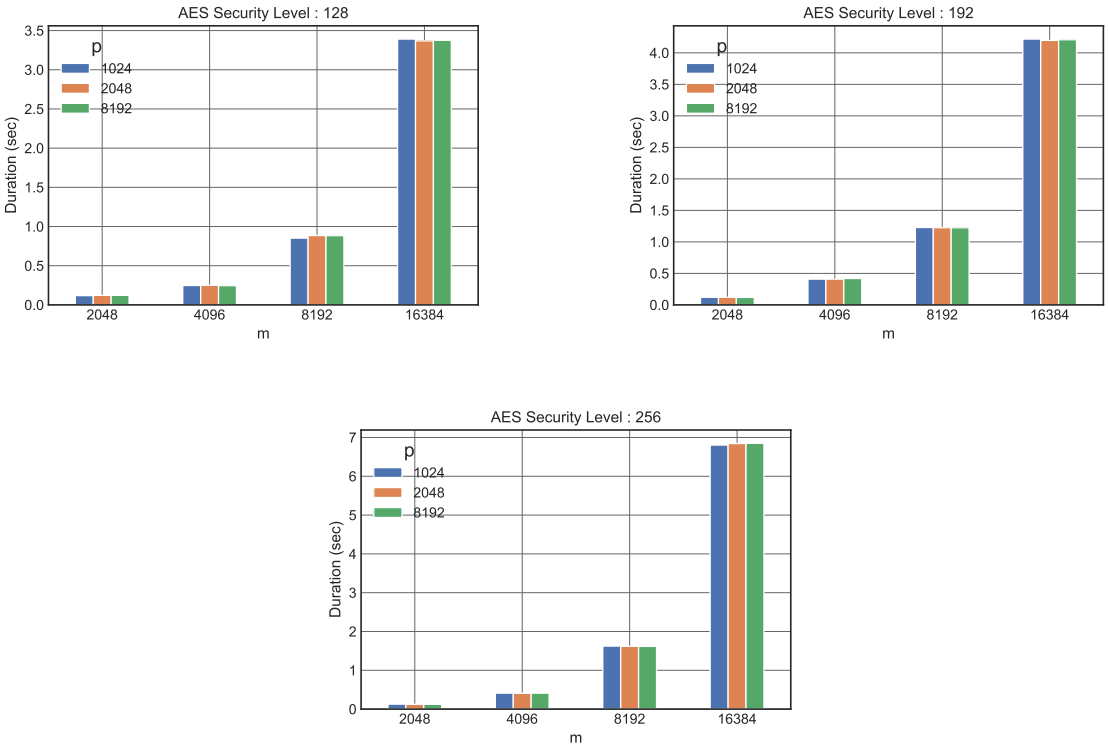

Figure 10. Biometric data matching execution time in seconds for polynomial coefficient modulus and security level equivalent in AES

According to Table 2, as in the sequential method, it is seen that the first parameter, which has the highest effect on working time, is polynomial modulus m, and the second parameter is AES-Security level.

Table 2. Parallel privacy-preserving biometrics matching method execution time with differet m, p and AES security levels.

		Duration		
m	**p**	**AES-256**	**AES-192**	**AES-128**
2048	1024	0.126655	0.120625	**0.118007**
	2048	0.122696	0.120993	0.121632
	8192	0.123279	0.118576	0.121733
4096	1024	0.410431	0.408464	0.247266
	2048	0.406968	0.408694	0.250110
	8192	0.409857	0.418568	0.244966
8192	1024	1.623653	1.228226	0.851412
	2048	1.617480	1.224950	0.885041
	8192	1.615464	1.223757	0.883087
16384	1024	6.803933	4.218115	3.391082
	2048	6.842727	4.197215	3.368394
	8192	**6.849738**	4.207780	3.375809

6. Conclusion

In this study, we proposed a fingerprint-based matching system with hash expansion and homomorphic encryption for privacy-enhancing of sensitive biometric data for cloud storage. Our proposed model extracts the fingerprint images using a mobile device and then enhance the fingerprint image using rotation and corner finding to extract the feature template. The system also provides a hash expansion technique with an RNG database that has a dedicated record for each system user.

After that, the system encrypts and store the sensitive biometric data in the cloud database. We also applied parallel execution techniques to reduce the matching process execution time, which is calculated in the encrypted domain.

Our results suggest that using parallel computation together with hash expansion and homomorphic encryption is a promising solution for the border control systems. Therefore, our solution scales gracefully with the size of the data. Moreover, many of the ideas presented here can be used to match other biometric data, for example, iris or face.

The suggested matching system's deficiency is that the model consumes time during matching of fingerprint in the encrypted domain. The matching of travelers in real-time identification using large fingerprint images needs more processing capability, computing resources, and storage.

Author Contributions: Conceptualization was conceived by F.O.C., methodology, implemetation, validation was done by F.O.C., S.Y.Y and MA The manusript's writing, review and editing by F.O.C., S.Y.Y and M.A.

Funding: "This research received no external funding"

Acknowledgments: In this section you can acknowledge any support given which is not covered by the author contribution or funding sections. This may include administrative and technical support, or donations in kind (e.g., materials used for experiments).

Conflicts of Interest: "The authors declare no conflict of interest."

References

1. Moon, S.; Lee, Y. An Efficient Encrypted Floating-Point Representation Using HEAAN and TFHE. *Security and Communication Networks* **2020**, *2020*.
2. Chatterjee, R.; Riazi, M.S.; Chowdhury, T.; Marasco, E.; Koushanfar, F.; Juels, A. Multisketches: Practical Secure Sketches Using Off-the-Shelf Biometric Matching Algorithms. Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security; Association for Computing Machinery: New York, NY, USA, 2019; CCS '19, p. 1171–1186. doi:10.1145/3319535.3363208.
3. Nguyen, H.M.; Rattani, A.; Derakhshani, R. Biometrics Fusion with Applications in Passenger Re-authentication for Automated Border Control Systems. 2019 IEEE International Symposium on Technologies for Homeland Security (HST), 2019, pp. 1–7.
4. Yang, W.; Wang, S.; Zheng, G.; Yang, J.; Valli, C. A Privacy-Preserving Lightweight Biometric System for Internet of Things Security. *IEEE Communications Magazine* **2019**, *57*, 84–89.
5. Murthy, S.; Abu Bakar, A.; Abdul Rahim, F.; Ramli, R. A Comparative Study of Data Anonymization Techniques. 2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS), 2019, pp. 306–309.
6. Çatak, F.Ö.; Mustacoglu, A.F. CPP-ELM: cryptographically privacy-preserving extreme learning machine for cloud systems. *International Journal of Computational Intelligence Systems* **2018**, *11*, 33–44.
7. Catak, F.O.; Aydin, I.; Elezaj, O.; Yildirim-Yayilgan, S. Practical Implementation of Privacy Preserving Clustering Methods Using a Partially Homomorphic Encryption Algorithm. *Electronics* **2020**, *9*, 229. doi:10.3390/electronics9020229.
8. Kumari, P.; Seeja, K. Periocular biometrics: A survey. *Journal of King Saud University - Computer and Information Sciences* **2019**. doi:https://doi.org/10.1016/j.jksuci.2019.06.003.
9. Kikuchi, H.; Nagai, K.; Ogata, W.; Nishigaki, M. Privacy-preserving similarity evaluation and application to remote biometrics authentication. *Soft Computing* **2010**, *14*, 529–536.
10. Li, X.; Li, J.; Huang, F. A secure cloud storage system supporting privacy-preserving fuzzy deduplication. *Soft Computing* **2016**, *20*, 1437–1448.
11. Gunasinghe, H.; Bertino, E. PrivBioMTAuth: Privacy Preserving Biometrics-Based and User Centric Protocol for User Authentication From Mobile Phones. *IEEE Transactions on Information Forensics and Security* **2018**, *13*, 1042–1057.
12. Im, J.; Jeon, S.; Lee, M. Practical Privacy-Preserving Face Authentication for Smartphones Secure Against Malicious Clients. *IEEE Transactions on Information Forensics and Security* **2020**, *15*, 2386–2401.

13. Zhang, C.; Zhu, L.; Xu, C. PTBI: An efficient privacy-preserving biometric identification based on perturbed term in the cloud. *Information Sciences* **2017**, *409-410*, 56 – 67. doi:https://doi.org/10.1016/j.ins.2017.05.006.
14. Zhang, Y.; Koushanfar, F. Robust privacy-preserving fingerprint authentication. 2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), 2016, pp. 1–6.
15. Tian, Y.; Li, Y.; Liu, X.; Deng, R.H.; Sengupta, B. Privacy-Preserving Biometric-Based Remote User Authentication. *Journal of Internet Technology* **2019**, *20*, 2265–2276.
16. Hahn, C.; Shin, H.; Hur, J. Cloud-based biometrics processing for privacy-preserving identification. 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), 2017, pp. 595–600.
17. Yang, B.; Li, G. Managing Private Credentials by Privacy-Preserving Biometrics. Emerging Technologies for Authorization and Authentication; Saracino, A.; Mori, P., Eds.; Springer International Publishing: Cham, 2018; pp. 47–55.
18. Aguilar-Melchor, C.; Fau, S.; Fontaine, C.; Gogniat, G.; Sirdey, R. Recent Advances in Homomorphic Encryption: A Possible Future for Signal Processing in the Encrypted Domain. *IEEE Signal Processing Magazine* **2013**, *30*, 108–117.
19. Dutta, S.; Cadambe, V.; Grover, P. Short-Dot: Computing Large Linear Transforms Distributedly Using Coded Short Dot Products. In *Advances in Neural Information Processing Systems 29*; Lee, D.D.; Sugiyama, M.; Luxburg, U.V.; Guyon, I.; Garnett, R., Eds.; Curran Associates, Inc., 2016; pp. 2100–2108.
20. Fan, J.; Vercauteren, F. Somewhat Practical Fully Homomorphic Encryption. Cryptology ePrint Archive, Report 2012/144, 2012. https://eprint.iacr.org/2012/144.
21. Bellare, M. *Advances in Cryptology-CRYPTO 2000: 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000. Proceedings*; Springer Science & Business Media, 2000.
22. Yang, Z.; Zhong, S.; Wright, R.N. Privacy-preserving classification of customer data without loss of accuracy. Proceedings of the 2005 SIAM International Conference on Data Mining. SIAM, 2005, pp. 92–102.
23. Shehu, Y.I.; Ruiz-Garcia, A.; Palade, V.; James, A. Sokoto Coventry Fingerprint Dataset, 2018, [arXiv:cs.CV/1807.10609].
24. Canny, J. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **1986**, *PAMI-8*, 679–698.
25. Chen, H.; Laine, K.; Player, R. Simple Encrypted Arithmetic Library - SEAL v2.1. Financial Cryptography and Data Security; Brenner, M.; Rohloff, K.; Bonneau, J.; Miller, A.; Ryan, P.Y.; Teague, V.; Bracciali, A.; Sala, M.; Pintore, F.; Jakobsson, M., Eds.; Springer International Publishing: Cham, 2017; pp. 3–18.

Sample Availability: Samples of the compounds ..... are available from the authors.