Comparative Study on Securing Biometrics Data

Brigitte Liu and Melonie Hardy

***Abstract*— The use of biometrics for identity verification and access control has remarkably increased in the recent years; however, technologies to ensure secure use of biometrics are still presently being developed and steadily improved upon. Biometrics data, which represents a unique identity of an individual, is difficult to replace when compromised. This project carries out a comparative study between several representative techniques that have been proposed for secure biometric matching. Two main techniques being examined include homomorphic encryption and cryptographic protocols and transformation using random projections. Some investigation is also made to error correction coding technique. The comparisons are done in terms of biometric matching accuracy, computational complexity, communication bandwidth required, and security strength offered.**

***Index Terms* — Biometric matching, Error correction coding, Homomorphic encryption, Random projections, Secure biometrics**

I. Introduction

Biometrics identifies a person based on his or her physiological or behavioral characteristics. Examples of biometrics include fingerprint, face, iris, and speech among many others. Compared to password based identity authentication that can be easily forgotten, lost, or stolen, biometrics are considered to be a better reflection of a person's identity because they are based on some inherent physiological or behavioral characteristics of that person. As such, biometrics cannot be forgotten, and have much lower chances to be lost or stolen than a password.

Because a biometric is a true indication of the identity of a person, it should be protected to prevent identity fraud. A biometric, if stolen in plain text, is difficult to replace because an individual cannot change his/her biometrics. The only solution is to store a different biometric from the same individual, however, there are a limited number of biometrics that can be used to identify a person. Biometric systems typically involve communication between a client and a server, thus there will be several points in the overall system that can potentially be attacked and should be properly secured [15], [16].

Section II of this report reviews the overall biometric system. Section III of the report gives a system overview, implementation techniques, and test results from each of the mentioned secure biometric methods. The following section will provide security strengths, trade-off and comparative results between the three methods. In the last section, we provide a conclusion to the paper as well as on-going and possible future work.

Manuscript received August 5, 2011. This material is based upon work supported by the National Science Foundation under Grant No. 1063035.

Brigitte Liu is an undergraduate senior with the Department of Computer Science, City College of New York, New York 10031 USA (e-mail: bliu01@ccny.cuny.edu)

Melonie Hardy is an undergraduate senior at Shorter University, Georgia 30165 USA (e-mail: melonie.hardy@hawks.shorter.edu)

II. Overall System & Points of Attacks

The overall system includes the client passing through a sensor that will capture the client's unique biometric feature. Many secure biometric techniques have been proposed in the literature and can be applied to different biometric modalities. Given the ease of capturing and popularity, we use face as the biometric modality studied in this paper. In order to use these faces with the various methods employed, the faces needed to be transformed by some means into feature vectors. We chose to use the eigenface method originally proposed by Turk and Pentland in [9]. We procured an open source code written by Delac, et al. that was used in [10] which was based on the Turk-Pentland eigenface method. This code was then modified to allow use of the AT&T Face Database [17] that was used for testing. Additionally, we modified the code so that the size for the feature vectors can be determined by the user for use in testing the methods to achieve different trade-off between computational/communication complexity and matching accuracy.

During verification, as shown in fig.1, the client extracts feature vector from the individual's face as query, the query is properly protected and sent to the server. The server then verifies the existence of the user is in the system. If the user exists in the system, the server will then provide the corresponding ID to the client. Generally, the three steps that are used in the overall biometrics matching protocol are: projection onto the face space, distance computation between the probe vector and each of the vectors in the database, and lastly, a minimum distance finding protocol. Eventually, the minimum distance is then compared to a threshold number; if the minimum is less than the threshold, it is assumed that a match has been found. However, if the minimum is larger than the threshold, the server will return a "no match" result to the client.

Given that there is more than one party involved in the overall protocol, it is important to secure the multi-party communication and computation. Prior works [1], [3], [5] have suggested several methods that address these security problems. The methods that are being studied typically assume a semi-honest adversary who would follow the protocol agreed by the two parties but can use gathered data to learn as much information about the other party as possible. Some of these side information may include an instance where the server finds out which ID the probe vector corresponds to, or

the client learning some of the features that were stored in the database, or other instances that may lead to either client and server learning more information than they are supposed to.

Like any systems, biometrics matching protocols also posed some problems. There are several points of attacks that revolve throughout the overall system, however, for the purposes of this project, only the points of attacks that deal directly with the client and the server are mentioned.

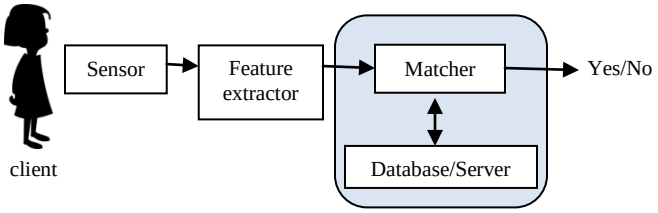


Fig. 1 The high level overview of the overall biometrics recognition and verification system. The purple area features the points of attacks (communication between client and server) that the three methods being studied will address.

The communications between the client and server is an integral part of the overall biometrics system. One of the earliest points of attack deals with the matching algorithm. If an outside adversary (which we call an attacker) were to tamper and alter the matching algorithm, the communication between the client and server is compromised. The next point of attack refers to our attacker directly attempting to steal from the database. This means that such an attacker may try to acquire one or more features that were previously stored in the database. Lastly, a point of attack occurs when an attacker attempts to tamper the communication between client and server. There are several ways that this could happen: an attacker may tamper with the communication protocol that may prove fatal to the end result or he/she may try to gain any information that is shared during the communication protocol.

It is apparent that if such an attacker attempts to target any of the points of attacks that were mentioned above, the overall biometric system is compromised. Consider a real-life situation where a high ranking official from the Department of Defense is trying to ID a high-profile crime boss and they require the assistance of the FBI's database administrator. Due to the nature of the highly classified case, the DOD official refuses to reveal the original picture of the suspect to the FBI's database admin. Similarly, the database admin does not want to reveal any other faces that reside in the database as it could result in a fatal privacy leak. In a case like this, a secure communication method between the two parties is highly necessary. The goal of this project is to compare three methods that address these issues and discuss their trade-offs. The first method, homomorphic encryption will utilize various cryptographic tools and properties to secure the communication between a client and a server. The second technique uses random projections which allows large dimension of data to be compressed and the values concealed by multiplication of the original data by a matrix of independent realizations of random variables. The last method, error correction coding is a coding technique that can correct small errors and inconsistencies in binary data using binary check matrices.

III. Secure Biometrics Techniques

A. Homomorphic Encryption and cryptographic protocol

A.1. System overview

The purpose of encryption is to hide the original message (which we call plaintext) into a cipher text such that the only party that may decrypt the cipher text to obtain the original message is the party that the message is intended for. In literature, the process of deciphering is named decryption There are two major types of encryption techniques: one is symmetric encryption where encryption and decryption use the same key shared between two parties. The other is asymmetric encryption, where one party encrypts a message using the public key from another party, who holds the private key which can decrypt the message encrypted by its public key. In traditional encryptions, the usual scheme is to encrypt a message, send it to the appropriate party and that party decrypts to retrieve the original message. However, there is no way to perform any computation over the encrypted data.

Homomorphic Encryption is distinctive in the way that it allows for operations and computations to be done on encrypted data which include addition and multiplication. This property is very important in many secure computation applications. In this project, a central cryptographic technique used is the partially homomorphic cryptosystem that is proposed by Paillier [2]. Given m = our plaintext, n = p*q where p and q are large prime numbers, a base g = n + 1 [2], and r is a number chosen randomly for each encryption; the cryptosystem performs encryption as shown in (1) and decryption as shown in (2). The result of encryption is denoted as c which is our cipher text. The public key will be a pair of n and g whereas the private key is the pair of λ and u.

$$c = g^m * r^n \mod n^2 \qquad (1)$$

$$m = \frac{L(c^\lambda * \mod n^2)}{L(g^\lambda * \mod n^2)} \mod n \qquad (2)$$

$$L(u) = (u - 1)/n \text{ and } \lambda = \text{lcm}(p - 1, q - 1)$$

For the purposes of the report, an encryption of a value is denoted with a bracket. For example for a plaintext a, the encryption of a is denoted as [a]. Additionally, the cryptosystem allows for addition between two encrypted values and multiplication between an encrypted value and a plaintext; given encryptions of [a] and [b], [a+b] is equivalent to [a]*[b] and [ab] is equivalent to $[a]^b$ respectively [1]. Another trick that will be utilized for added security is a method called re-randomization which allows a cipher text to be "publicly changed into another one without affecting the plaintext" [2] which is denoted by (3). We call the re-randomized text c'.

$$c' = c * g^{nr} \bmod n^2 \qquad (3)$$

The scenario that we use is a situation where we have our client Alice and our server Bob who holds the database in plaintext feature vectors shown in Fig.2. Alice has a face feature vector and she wants to determine if the face matches with any of the faces in the database. However, Alice does not wish to reveal the face she holds to Bob and similarly, Bob does not want to reveal any data from his database. Alice may only know some basic parameters of the database such as its size [1]. Lastly, Bob will provide Alice with the resulting ID in its encrypted form such that even he himself is unable to see the ID that Alice's face is associated to. This system allows both Alice and Bob to encrypt using the Alice's public key but only Alice is allowed to decrypt using her private key.

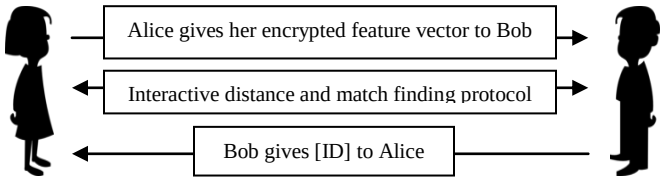


Fig. 2 Alice and Bob does an interactive distance and match finding protocols in the encrypted domain. By carrying out such protocol, security of the all the plaintext features is ensured.

The complete protocol is described in [1] in details, however, some of the protocols were revised for this project. Distance computation is the first step that must be processed. Alice's encrypted vector is denoted as [Ω'] and Bob's feature vector [Ω]. Consider a database of size M and feature vector of size k, this means that Bob has M feature vectors and the distance between Ω' and Ω_1, Ω_2, … Ω_M must be computed. The distance is computed using the squared Euclidean distance method (4).

$$D(\Omega, \Omega') = \| \Omega - \Omega' \|^2 \qquad (4)$$

$$= (\omega_1 - \omega_1')^2 + \ldots + (\omega_k - \omega_k')^2$$

$$= \sum_{i=1}^{k} \omega_i^2 + \sum_{i=1}^{k} (-2\omega_i \omega_i') + \sum (\omega_i')^2$$

In the equation above, we can denote the first operand of the addition S1, the second S2, and the third S3. Using the homomorphic property of addition, we can convert the equation to (5) such that it will be computed in the encrypted domain. Since Bob has the first operand in plaintext, he can go

$$[D(\Omega, \Omega')] = [S1]*[S2]*[S3] \qquad (5)$$

ahead and compute S1 and then encrypt it to [S1] and similarly for the third operand. However, for the third operand, Alice is the one that must compute S3, encrypt and then sends [S3] to Bob. For the second operand, since Bob has ω_i in plaintext, by using the homomorphic property, Bob can compute $[\omega_i']^{(-2\,\omega i)}$ instead which will yield us [S2].

After this protocol, an array of encrypted distances is available to use. The only step left to do is find the minimum distance among the array of encrypted distance values. For this, we must consider a recursive algorithm that compares two encrypted numbers in each iteration. A security measure is added by re-randomizing the resulting minimum distance from each iteration. This will result in one final answer, the encrypted minimum distance.

The first thing to consider is a comparison protocol between two encrypted values. Such a comparison has to be done in several steps. Given two encrypted distances [a] , [b] and a bit length L, we would like to find a truth value z_L shown in (6) where $z_L = 1$ if and only if $a < b$ and $z_L = 0$ if and only if $a > b$. When Bob gets an encryption $[z_L]$, the [min(a,b)] can easily be computed using an application of the homomorphic properties and an interactive protocol between the two parties where min(a,b) = (a < b) (a – b) + b.

$$[z_L] = ([z]*[z \bmod 2^L])^{2^{-L}} \qquad (6)$$

The only part that is missing is the computation for z mod 2^L which can be done by executing a protocol with Alice. The series of equation below will summarize the steps necessary to obtain z mod 2^L. Bob must compute (7) but he is missing the operands necessary to finish his computation: [z'] and [λ]. Therefore, he computes (9) first where r_1 is a random value and he then sends Alice [d]. Alice can then decrypt and compute d mod 2^L and sends the encryption to Bob. Since Bob has the value of r_1, he can now compute (8). Unfortunately, there is still one more step to be done: the computation on [λ].

$$[z \bmod 2^L] = [z']*([\lambda]^{2^L}) \qquad (7)$$

$$[z'] = [d \bmod 2^L]*[r_1 \bmod 2^L]^{-1} \qquad (8)$$

$$[d] = [z]*[r_1] \qquad (9)$$

The purpose of computing λ is because it is a binary value indicating whether $(r_1 \bmod 2^L) > (d \bmod 2^L)$. To summarize the situation at this point: Alice holds the value of $(d \bmod 2^L)$, Bob holds the value of $(r_1 \bmod 2^L)$, and Bob must get the comparison result [λ] between their two private inputs.

Let us denote $(d \bmod 2^L) = d'$ and $(r_1 \bmod 2^L) = r'$. Recall that earlier, we specified the bit length to L. Alice must then convert her d' into its binary representation and encrypt each bit and sends $[d'_{L-1}]$, … , $[d'_0]$ to Bob. Meanwhile Bob has the value of r' and he too, must convert his value to binary and encrypt each of his bits. He also chooses a uniformly random s, s = -1 or s = 1. With these inputs, he can finally compute (10) using the help of (11). Note that since d' and r' are binary values, there is essentially no need to square them.

$$[c_i] = [d'_i]*[-r'_i]*[s]*\left(\prod_{j=i+1}^{L-1} [w_j]\right)^3 \qquad (10)$$

$$[wj] = [d' \text{ XOR } r'] = [d'_j{}^2]*[d'_j{}^2]^{-2r_j} *[r'_j{}^2] \quad (11)$$

$$[e_i] = [c_i]^{r2_i} \quad (12)$$

Once Bob has the value of $[c_i]$, he masks this value with a random r_2 and sends (12) to Alice. On the other hand, Alice can decrypt each of the $[e_i]$ and check if any of them is a zero. If so, she can encrypt a value of [1] = [λ'] or if such is not the case, she encrypts a value of [0] = [λ'] and sends [λ'] to Bob. Since Bob has the value of s from earlier computation, he can interpret the meaning of [λ']. Is s = -1, he will compute $[\lambda] = [1]*[\lambda']^{-1}$ while is s = 1, the will just set [λ] = [λ']. With all the inputs available to Bob, he can now compute $[z_l]$ and find the [min(a,b)].

A.2. Implementation

For the Purposes of the Paillier Cryptosystem, the open source code procured by Kun Liu in [8] was used for encryption and decryption. The re-randomization function as well as the rest of the system was written in details to follow the protocols from [1]. The overall system was written in java and consists of several classes. The client and the server were separated into two different classes to portray the communication between the two parties while still using one machine. The feature vectors that were used for testing were received from a file and are used as the database. The database holds the faces of 40 different users. Each user originally has 10 images each with each image slightly different from each other (e.g. different facial expressions). For a database size of 200, we set each user to have 5 faces each, whereas for database size 400, each user holds 10 faces.

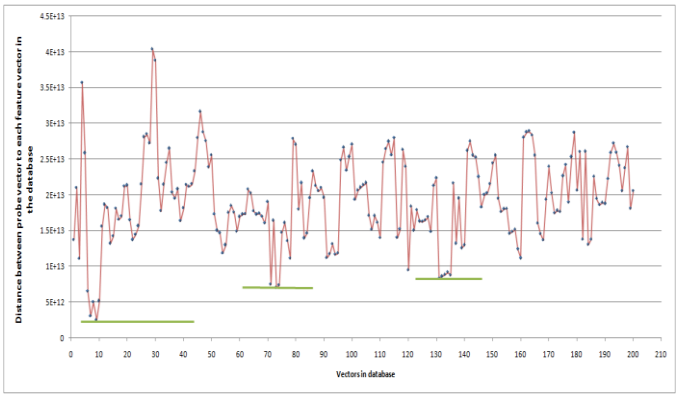
Since the process of encryption and decryption requires the operand to be integer, quantization must first be done to the feature vector matrices. The quantization process was completed in Matlab and thus the file used as the database consisted of a matrix of integers.

Additionally, since the end result is essentially an [ID], we pair each feature vector in the database and thus also the [distance] between the feature vector to the query with its corresponding [ID]. However, no computation is done to [ID].

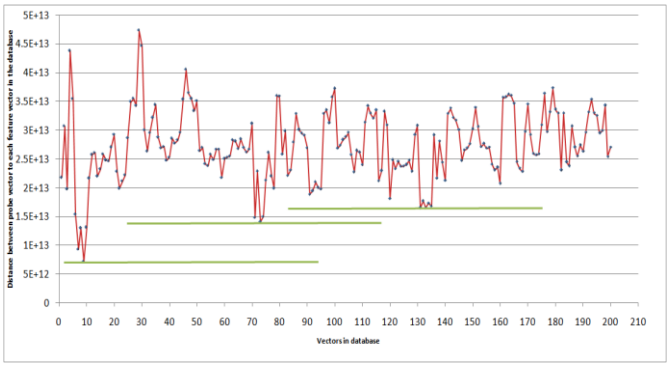
For testing, we implemented the plaintext computation to serve as the base case for all comparative results.

A.3. Results

In order to test the system, we used 200 queries consisting of faces that are not in the database. We also observe the computational results of varying vector sizes : 10, 25, 50, 100, and 150 (the varying vector sizes are the result of the eigenface algorithm mentioned in section II of this paper).



(a)



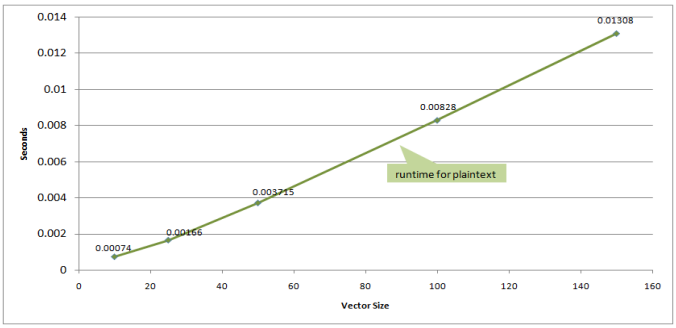
(b)

Fig. 3 Distance computation between the query vector and each of the vectors in the database. Graph (a) uses vectors of size 25 and (b) uses vectors of size 150. The probe vector used is of user 2 and thus the minimum should lie between vectors 5 – 10 in the database. As is shown in both graphs, the minimum distance indeed lies between the mentioned vectors which will result in the correct ID that the probe is matched to. he green line in each graphs showcased the minimum line with the next two minimum distances above it. Graph (a) shows that the minimum distance is close to the next minimum which may impact the accuracy of the system whereas (b) shows the minimum lines are much farther apart. This is because a longer feature vector captures more information about the face, and thus more distinctive for matching.

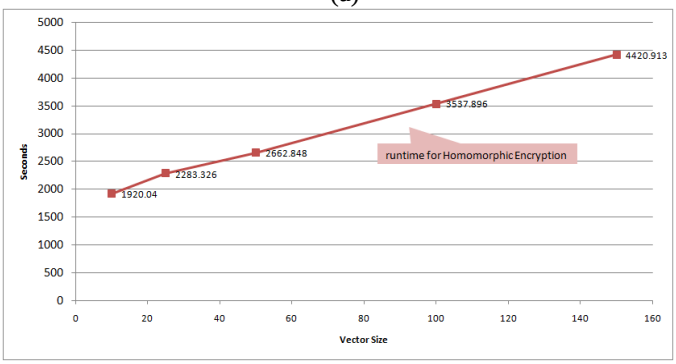
The first testing that was done is distance computation between the query vector with each of the vectors in the database. This is done to verify that the minimum distance has the same ID as the query feature vector. One example is shown in fig.3 and for the rest of the testing, various query vectors were employed.

One of the trends that were perceived is the runtime difference between the plaintext computation and the homomorphic encryption system as is shown in Fig 4.

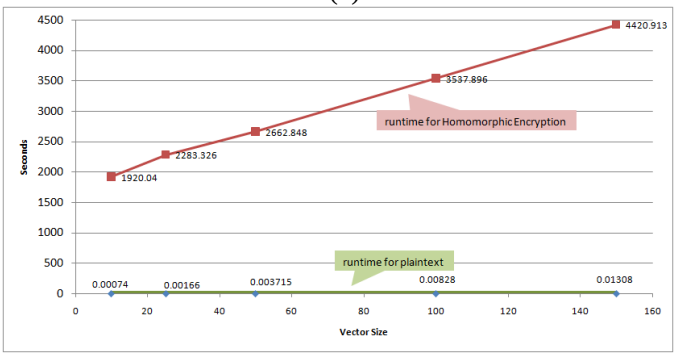
Both plaintext and homomorphic encryption have computational complexity linear to the vector size, however, the computation time is much higher (around ~300,000 longer than plaintext) for homomorphic encryption.



(a)



(b)



(c)

Fig. 4 Vector size v. Runtime (seconds). Graph (a) shows the runtime for plaintext computation. The runtime for even the largest vector size is still below 0.015 of a second, compared to (b) which shows runtime for homomorphic encryption, the same vector size takes ~77 minutes to complete. Graph (c) shows the runtime difference between plaintext computation and homomorphic encryption based on varying vector size.

Next, we tested the matching accuracy of the system based on varying database sizes which is displayed in fig. 5. Since homomorphic encryption preserves the accuracy of the plaintext after quantization (unlike the rest of the methods being studied), therefore, all the testing for matching accuracy was done in the plaintext domain.

It is observed that because the same person has more faces in the database, the chance of correct matching is increased. For database size 399 and on the largest vector size, the matching accuracy already approaches that of 98.5%. In the later section, a comparison against the plaintext matching accuracy (without quantization) is provided as a base line for comparison.

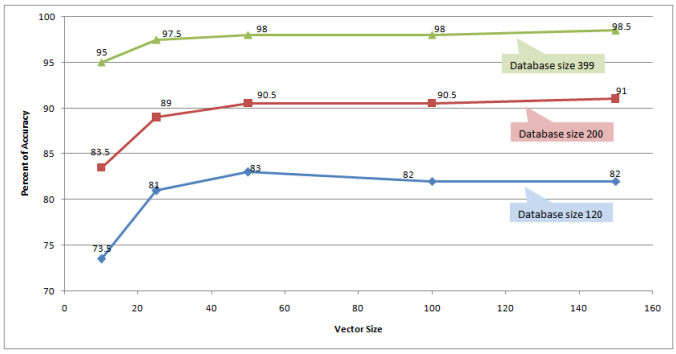


Fig. 5 Vector size v. Matching Accuracy (%). All 40 users are still in the database. The blue line signifies database of size 120 which consists of each user having 3 faces in the database. The red line signifies a database of size 200 which consists of each user having 5 faces in the database and lastly for database size 399, each user has 10 faces in the database. However, for each probe vector for database 399, in order to avoid a 0 case, the vector that will give an exact match (a minimum distance of 0) is removed.

B. Random Projections

B.1. System Overview

Random projection is a method by which a set of data can be embedded from a higher dimensional space to a lower dimensional space while approximately maintaining the distances between the data points.

This concept is based on the Johnson-Lindenstrauss Lemma as described in [4]: Given $\epsilon \in (0,1)$. For every set S of #(*S*) points in $\mathbb{R}^N$, if n is a positive integer such that $n > n_0 = O(^{\ln(\#(S)}/_{\epsilon^2})$, there exists a Lipschitz mapping $f : \mathbb{R}^N \rightarrow \mathbb{R}^n$ such that:

$$(1-\epsilon)\,\|u-v\|^2 \le \|f(u)-f(v)\|^2 \le (1+\epsilon)\,\|u-v\|^2$$

for all u,v $\in S$.

The core computation of random projections is depicted in (13). where Φ is a random *n* x N matrix of elements that are independent realizations of a random distribution such as a Gaussian or Bernoulli distribution. In the equation, g is an N

$$y = \Phi g \qquad (13)$$

dimension vector and y is an *n*-length vector that approximately preserves the pair wise distance of the data points of g.

In a biometric system, random projection would be performed at the enrollment stage and the verification stage. In this way, the original feature vector of a user would not be stored in the database or transmitted during verification so as to prevent adversary attacks. Without knowing the random projection matrix, it can be very difficult for an attacker to recover the original feature vector from the projected feature vectors.

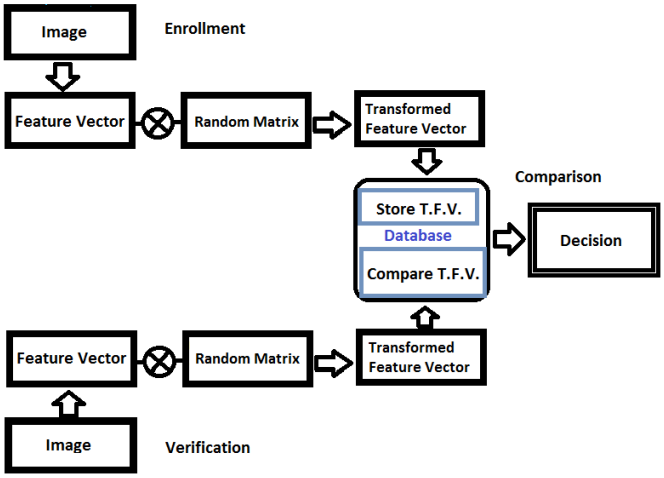


Fig. 6 High level overview of the random projections system. The system consists of two levels: enrollment and verification. In both stages, a feature vector in derived from an image and then transformed by a random matrix. In enrollment, the transformed feature vector is stored in the database for later comparison and in verification the transformed feature vector is compared with the transformed feature vectors within the database. After comparison, a decision is reached as to whether or not the user exists within the database.

B.2. Implementation

This implementation was coded in Matlab and is similar to the proposed scheme in [3]. During the enrollment stage of the implementation, half of the available faces in the database were subjected to the eigenface method mentioned previously to create a database of size two hundred. Each feature vector of the database was then subjected to random projection by using a random matrix composed of independent realizations of a Gaussian distribution. In practice, either one matrix can be created and utilized on all of the available feature vectors or one matrix can be created for each user. The experiment later found that using one matrix for each user provided the system with a better accuracy rate, therefore that method was used in the final implementation. The transformed feature vectors were compiled into a matrix to form the database for comparison and the random matrices were stored in binary files for later use in the verification process. An array of identification numbers was also created to facilitate the matching process in verification.

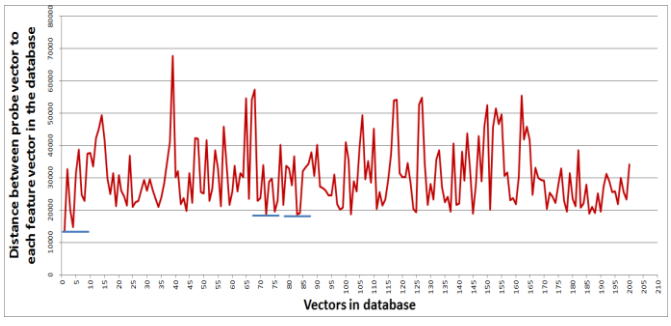


Fig. 7 Distance computation for a single query in a random projection system using one matrix for each user. The probe vector comes from user one, therefore the expected range that the minimum should fall in is 1-5. As is shown, this is true and the next two minimums have a fairly large spread from the actual minimum.

In the verification stage of the implementation, one face image at a time was brought into the system and a feature vector is extracted from the face image using the eigenface method. An identity for the face image would be proposed based on which user number was provided when selecting the image from the available images. The proposed user number would be utilized to retrieve the random matrix for that particular user that was created and used in the enrollment stage. Random projection would then be performed on the feature vector. The transformed feature vector would then be compared to every transformed feature vector in the database that was created in enrollment using the Euclidean distance computation between each vector in the database and the transformed feature vector. The minimum distance and minimum index was selected based on the calculated distances. The minimum index was used to obtain the identification number associated with the minimum distance. From there, the originally proposed user number was compared to the selected identification number to determine if it was a match. This process was run two hundred times to test every available image that was not used to create the database in the enrollment stage.

B.3. Results

Final result calculations utilized one random matrix for each user due to this method showing a significantly higher accuracy rate. Vector sizes of 10, 25, 50, 100, 150 were tested on the basis of enrollment time, verification time, accuracy, and bandwidth usage. Enrollment times ranged from 6.7 seconds to 13.7 seconds based on vector size.

Verification time ranged from194 milliseconds to 233 milliseconds. In the preliminary testing stage, the accuracy rate when using one matrix for all users was shown to range from 64% - 73%. In the final testing stage, the accuracy rate when using one matrix for each user was shown to range from 83% - 85% with a database size of 200, which indicates five pictures each for 40 users.

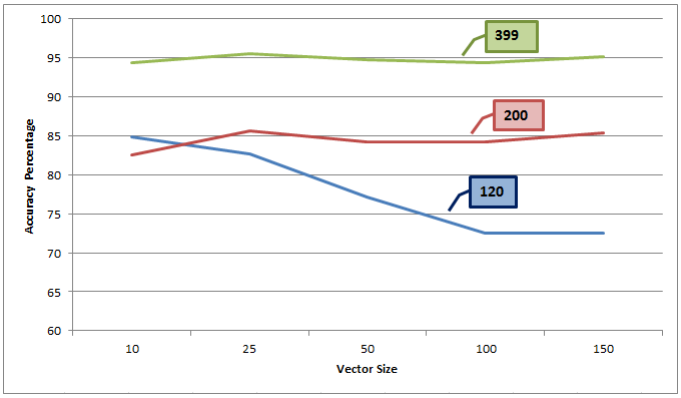


Fig. 8 Comparison of accuracy versus database size using various vector sizes during verification. As is shown, a larger database size yields higher accuracy performance due to a larger number of training images for the eigenface method in enrollment.

When the database size was increased to its maximum possible size of ten pictures for each of 40 users and one

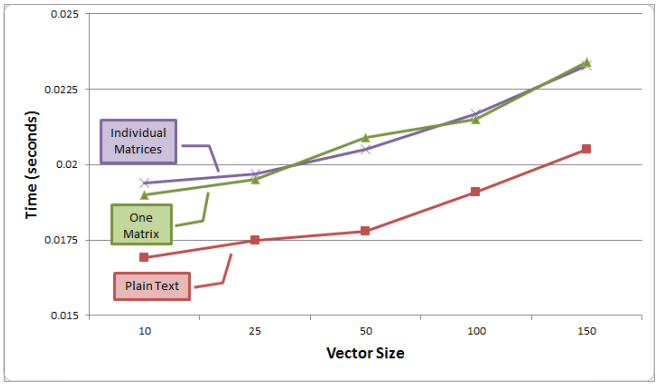


Fig. 9 Comparison of time for a single query lookup using various vector sizes. Plain text computation is clearly the most expedient process, although using one random matrix for the system or one random matrix for each user is not far behind. The time difference would be unperceivable by the human brain.

feature vector at a time was removed to be used at the query vector the accuracy rate increased to between 94% and 95%. This indicates that as the number of images per user increases, the accuracy rate for the method also increases.

From these results it can be observed that the random projection method is an expedient method with reasonable accuracy. The random projection method is useful against an adversary in the fact that it randomizes the elements of a feature vector to mast the original measurements both at the enrollment and verification stage. For an attacker to be able to gain access to a system, they would need to have either an image or feature vector of a user and the correct random matrix for that user. Although these may be possible to obtain, if it is found that a user's random matrix has been stolen a new random matrix can be generated to protect the user's biometric. This is also a reason to use a separate random matrix for each user because a separate random matrix for each user would ensure that if one user's random matrix is compromised, every other user's random matrix will not be affected. For practical use, it is advised that a separate random matrix be used for each user and a new random matrix be issued regularly to prevent successful adversary attacks.

C. Error Correction Code

C.1. System Overview

Error correction coding is a technique designed to enable reliable transmission of information over noisy communication channels. It can be used to correct minor deviations between enrollment and probe biometrics so as to provide a more suitable match during verification. During enrollment in a system employing error correction code the feature vector representing a user's biometric is encoded using (13) where H is a binary matrix and **s** is the syndrome of the message x.

$$s = f_{sec}(x) = Hx \qquad (14)$$

In most systems, a hash function is also employed that creates a hashed version of the enrollment feature vector. The syndromes for the users and the hashed biometric feature vectors are stored in the database for later use during enrollment.

In the verification stage, a user will provide a biometric that may not be exactly the same as the enrollment template. Syndrome bits will be used to attempt to correct small discrepancies between the probe and stored biometric feature vector. A new feature vector will be outputted from the syndrome decoding and then hashed as in enrollment. To be considered a legitimate user, the hashed version of the probe biometric feature vector must perfectly match the hashed biometric stored in the database.

C.2. Implementation

This implementation was performed in Matlab with some C code through MEX exchange in Matlab. The code was obtained from [7] and edited to be able to work with the AT&T face database that was used in this project. The code is also similar to the scheme mentioned in [5] This code is designed to work with a vector size of 396.In order for the code to be usable, after performing the eigenface method on the images in the database, each feature vector was then binarized. In the enrollment stage, the code calculates the crossover probability between different binary feature vectors of the same user to determine how likely it is and how much that an incoming feature vector will differ from the stored feature vectors. The code then creates a Low Density Parity Check (LDPC) matrix and encodes each binary feature vector in the database. The syndrome bits for each binary vector are accumulated during this process and stored for later use.

During the verification stage, an image is brought in and the eigenface method is performed to create a feature vector. This feature vector is then binarized to be able to work with the code. The binary feature vector is then decoded in an attempt to find a match in the database for the probe feature vector. The comparison is done by computing the XOR between the binary probe and the stored probe and taking the sum of the resulting array to determine if they are a perfect match. Only a sum of zero is considered to be a match.

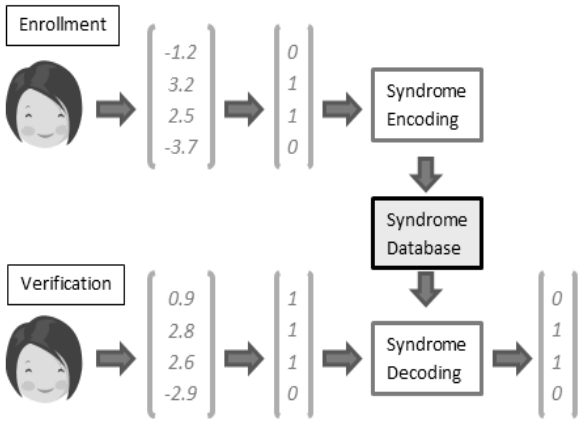


Fig. 10 Overall system of the Error Correction method. There are two steps that must be done: enrollment and verification. In enrollment, the most likely errors are calculated by the syndrome encoding and stored in syndrome bits. In verification, a feature vector is extracted and the stored syndrome bits are used to attempt to reconstruct a viable match to the original feature vector.

C.3. Result

Due to the specifications of the code, only vector sizes of 396 were tested as opposed to previously when any vector size could be tested. Error correction code in the form of this implementation has the ability to protect the stored feature vectors and probe feature vectors because they are a binarized form of the original measurements that mask the true value of the original measurements. The syndromes also protect the integrity of the system because a typical probe biometric will be flawed and need correction by the syndromes to be able to match perfectly to a feature vector in the database, but the syndromes themselves do not reveal any information about the feature vectors.

IV. Comparison Results

A. *Security Trade-offs*

Homomorphic encryption protects all channels of communication between the server and client by continually re-randomizing the encryption on the data. Random projection protects the database and the probe feature vector by transforming the feature vectors. Error correction code protects the database and probe feature vector by masking the original values with binarization and using syndrome bits for decoding that do not reveal anything about the original feature vectors.

Homomorphic encryption can be compromised when an attacker directly targets the database because the database is stored in plaintext. Random projections can be compromised along the lines of communication if a randomly projected feature vector is stolen and resubmitted. Random projection also requires the client or server to hold the specific random matrix for the client's use during verification, which could possibly be lost or stolen. An error correction coding system can be compromised when a binary feature vector of a legitimate user is stolen.

It can be observed that Homomorphic encryption provides the highest security strength because it continuously masks the communication between the server and the client during the matching process. The only apparent weaknesses in the Homomorphic encryption method is that the features stored in the database are in plaintext and the computation complexity requires an exorbitant amount of time and resources. Random projection provides security within the database that Homomorphic encryption does not, but the random projection method can be compromised along the lines of communication if a randomly projected feature vector is stolen. Error correction coding also provides security on the database and probe feature vectors, but security is highly compromised along the lines of communication between client and server should any information is stolen.

B. *Quantitative Comparisons*

On a more quantitative measure, homomorphic encryption also provides the highest accuracy of the methods study with an accuracy rate of 91% on a database size of 200 with a vector size of 150 as compared to an accuracy rate of 85.4% on a database size of 200 with a vector size of 150 with random projection.

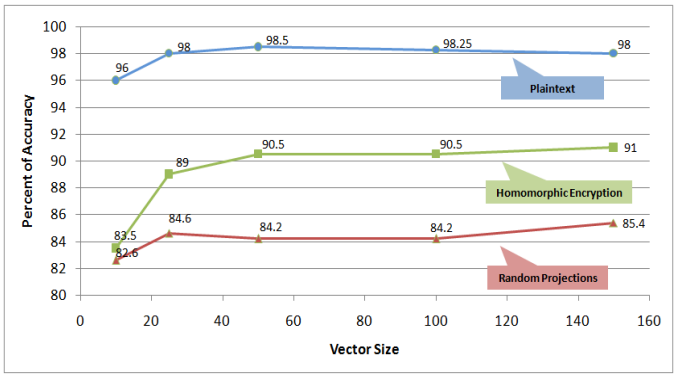


Fig. 11 Vector size v. Matching Accuracy (%). Comparison of matching accuracy of plaintext (without quantization), Homomorphic encryption (which is equivalent to the matching accuracy of the plaintext computation *after* quantization), and random projection methods based on a vector size of 200 which refers to five face image vectors for each of forty users and it is also based on varying vector sizes.

Homomorphic encryption has a high accuracy rate due to the fact that it preserves the actual values of the original feature vectors after quantization. It does not have as high of an accuracy rate as in plain text because of the quantization. The random projection method has a low accuracy rate because the random matrices that are used distort the actual values of the original feature vectors. It should be noted that the accuracy rate of 84.5% is based on using one random matrix for each user. Using one random matrix for every user reduces the accuracy rate by roughly ten percent.

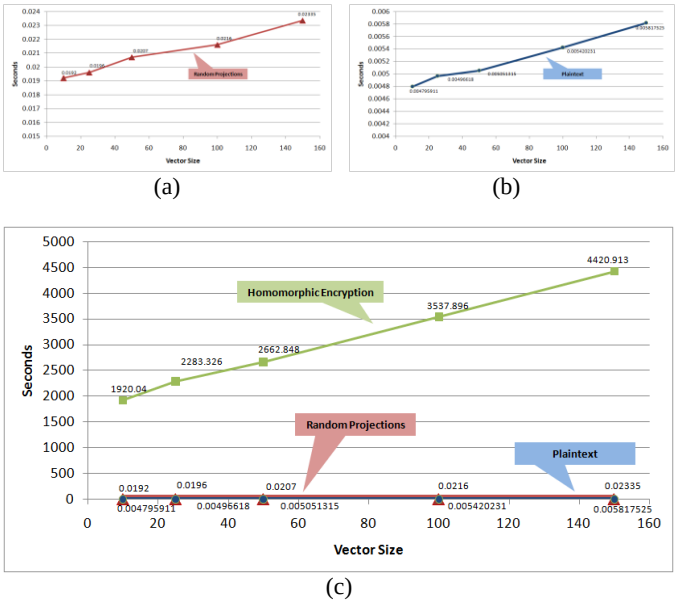


Fig. 12 Vector size v. Runtime (seconds). Graph (a) shows the runtime in random projections and graph (b) shows the runtime using plaintext computation (without quantization). Graph (c) provides a comparison of single query runtime of plaintext (without quantization), Homomorphic encryption, and random projection methods based on a vector size of 200 (five face images for each of forty users) and varying vector sizes. All the graphs show linear increase as the vector size increases.

In respect to runtime in completing one query, the random projection method is the fastest method studied with a runtime of roughly 230 milliseconds with a vector size of 150.

Although the baseline time for plaintext computation is roughly 60 milliseconds for a vector size of 150, the difference between the runtime for plaintext and random projection would not be perceived by the human brain.

Homomorphic encryption on a vector size of 150 has a runtime of roughly seventy-seven minutes to complete a single query. This indicates that Homomorphic encryption takes roughly 4.6 million times longer than the random projection method.

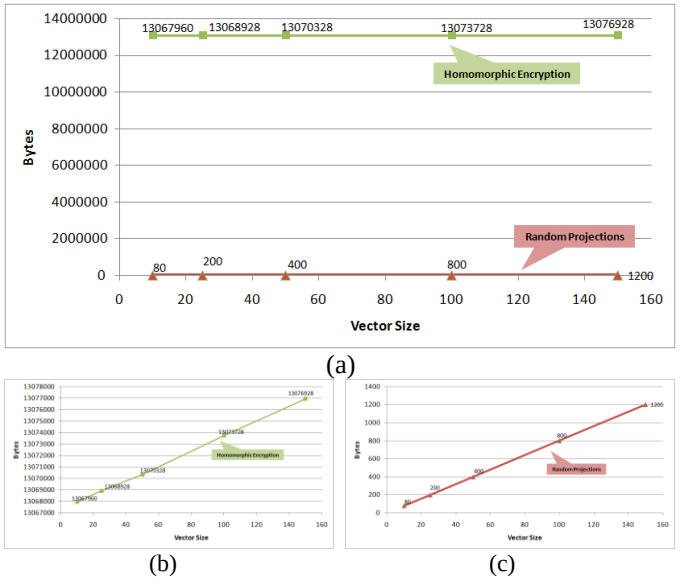


Fig. 13 Vector size v. communication bandwidth (bytes). Graph (b) and (c) displays the communication bandwidth sent between the client and the server of homomorphic encryption and random projections methods respectively. Graph (a) provides a comparison of the two methods shown on the same figure based on a vector size of 200 (five images for each of forty users) and varying vector sizes. Both graphs show a linear behavior in respect to vector size.

In respect to communication bandwidth, it can be seen that the random projection method uses the least amount of bandwidth. The random projection method uses exactly eight bytes for each unit of vector size in the transformed feature vector that will be transmitted from client to server in the verification process. The bytes passed between the client and server in the Homomorphic encryption is over 13 megabytes per query and increases with an increase in vector size. Homomorphic encryption utilizes more communication bandwidth than the random projection method due to the fact that there are a considerable amount of interactive protocols used in the communication between client and server.

V. CONCLUSION AND FUTURE WORK

This project aims to provide a comparative study between the three methods of securing biometrics system that were proposed in various literatures. Overall, it can be concluded that homomorphic encryption is by far, the most secure method of the methods studied due to the secure communication within the encrypted domain at every point of the interactive protocols between the client and server. However, one of the biggest obstacles to its practical deployment is that -off is that it is the method that has the highest computation and communication complexity.

Homomorphic encryption is also the most accurate method studied because it is able to maintain the exact value of the elements of the feature vectors after quantization while they are masked by encryption. While the random projection method may provide somewhat less security and accuracy, it is incredibly fast compared to homomorphic encryption and provides the possibility of cancelable biometrics in the case of a stolen randomly projected feature vector. However, it is observed that accuracy of random projection scheme can be improved by using longer projected feature size. (i.e., given the same original feature size but increasing the size of projected feature, accuracy can be improved).

The last method that was mentioned, error correction coding is currently being implemented. Once implemented successfully, the same testing using the same parameters that were used by the other two methods may be applied.

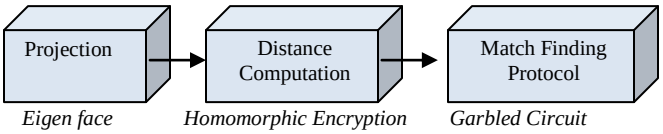


Fig. 14 Overall system of the optimization of the homomorphic encryption method using Yao's garbled circuit. Before actually feeding values into the implemented circuit, there must be preparations done such as conversion of the encrypted distances into garbled values, circuit construction, as well as the oblivious transfer protocol.

On the other hand, despite the high accuracy and high security, the homomorphic encryption method is deemed impractical especially in real-life situations because it is very time and space consuming. Recent literature [11], [12] had proposed an optimization of this method. The method uses cryptographic tool such as oblivious transfer and a minimum finding protocol for secure multi-party computation using the garbled circuit proposed by Yao [14]. The system structure of the optimized system is displayed in Fig. 14. Our future work includes implementing and examining this optimization method on the same grounds that were used on the previous methods studied.

ACKNOWLEDGMENT

We would like to thank our advising mentor, Dr. Min Wu as well as our graduate student Wenjun Lu for their continuous guidance, support and mentorship throughout the course of the project. We also acknowledge our external consults Wei Hong Chuang and Philip Liu. We would also like to thank the MERIT program and the National Science Foundation for providing us with such opportunity.

REFERENCES

[1] Erkin, Z.; Franz, M.; Guajardo, J.; Katsenbeisser, S.; Lagendijk, I.; Tomas, T.; , "Privacy-Preserving Face Recognition," *PETS '09 Proceedings of the 9th International Symposium on Privacy Enhancing Technologies*, 2009.

[2] Pascal Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes", *EUROCRYPT* 1999, pp223-238.

[3] Pillai, J.K.; Patel, V.M.; Chellappa, R.; Ratha, N.K.; , "Sectored Random Projections for Cancelable Iris Biometrics," *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on* , pp.1838-1841

[4] Pillai, J.; Patel, V.; Chellappa, R.; Ratha, N.; , "Secure and Robust Iris Recognition using Random Projections and Sparse Representations," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* , vol.PP, no.99, pp.1.

[5] Sutcu, Y.; Rane, S.; Yedidia, J.S.; Draper, S.C.; Vetro, A.; , "Feature extraction for a Slepian-Wolf biometric system using LDPC codes," *Information Theory, 2008. ISIT 2008. IEEE International Symposium*, pp.2297-2301, 6-11 July 2008

[6] Vetro, A.; Draper, S.; Rane, S.; Yedidia, J., "Securing Biometric Data", *Distributed Source Coding*, ISBN-13: 978-0-12-374485-2 Algorithms and Applications, January 2009.

[7] Varodayan, David. Rate-Adaptive LDPC Accumulate Codes for Distributed Source Coding. n.d. 22 July 2011 <http://ivms.stanford.edu/~varodayan/ldpca.html>.

[8] Liu, Kun. Paillier's Cryptosystem. 13 June 2011. < http://www.cs.umbc.edu/~kunliu1/research/Paillier.html>.

[9] Turk and Pentland, Face Recognition Using Eigenfaces, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3-6 June 1991, Maui, Hawaii, USA, pp. 586-591.

[10] Delac, Grgic, and Grgic, Independent Comparative Study of PCA, ICA, and LDA on the FERET Data Set, *International Journal of Imaging Systems and Technology*, Vol. 15, Issue 5, 2006, pp. 252-260.

[11] A. Sadeghi, T. Schneider, and I. Wehrenberg. Efficient Privacy-Preserving Face Recognition. *In International Conference on Information Security and Cryptology*, 2009.

[12] Y. Huang, L. Malka, D.Evans, J.Katz. Efficient Privacy-Preserving Biometric Identification. In *Proc. of the 17th Annual Network and Distributed System Security Symposium (NDSS)*, 2011

[13] Y. Lindell and B. Pinkas. A Proof of Security of Yao's Protocol for Two-Party Computation. Journal of Cryptology, 22(2), 2009

[14] A. C. Yao. How to Generate and Exchange Secrets. In 27th Symposium on Foundations of Computer Science, 1986

[15] Salil Prabhakar, Sharath Pankanti, and Anil K. Jain. 2003. Biometric Recognition: Security and Privacy Concerns. *IEEE Security and Privacy* 1, 2 (March 2003), 33-42.

[16] Uludag, U.; Pankanti, S.; Prabhakar, S.; Jain, A.K.; , "Biometric cryptosystems: issues and challenges," *Proceedings of the IEEE* , vol.92, no.6, pp. 948- 960, June 2004.

[17] AT&T Laboratories Cambridge. The Database of Faces. 2002. 1 June 2011<http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html >.

Brigitte Liu is a rising senior at the City College of New York Honors Program, and a participant of the 2011 MERIT program. She will receive her Bachelor of Science degree in Computer Science in June 2012.

Melonie Hardy is a rising senior at Shorter University and a participant of the 2011 MERIT program. She will receive her Bachelor degree in Computer Information Systems and mathematics in June 2012.