On the Secure Design of Multi-biometric Template Protection Approaches for Biometric Authentication

By

Surabhi Garg

Under the supervision of Donghoon Chang

Computer Science and Engineering

Indraprastha Institute of Information Technology Delhi

New Delhi– 110020

June, 2021

On the Secure Design of Multi-biometric Template Protection Approaches for Biometric Authentication

By

Surabhi Garg

A Thesis

submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Computer Science and Engineering

Indraprastha Institute of Information Technology Delhi

New Delhi– 110020

June, 2021

Certificate

This is to certify that the thesis titled *On the Secure Design of Multi-biometric Template Protection Approaches for Biometric Authentication* being submitted by *Surabhi Garg* to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of Doctor of Philosophy, is an original research work carried out by her under my supervision. In my opinion, the thesis has reached the standard fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree or diploma.

June, 2021

Dr. Donghoon Chang

Indraprastha Institute of Information Technology Delhi

New Delhi 110020

Abstract

The biometric systems are widely deployed in various sectors for authentication purposes– India's Aadhaar project (a multi-biometric database of above 1.3 billion of India's population) being the prominent example. The multi-biometric systems are preferred these days as they are more reliable and provide an enhanced security level. Generally, the biometric data known as a biometric template is stored on the database server in the raw, unprotected form. Unlike passwords, biometric data, once leaked or stolen, remains compromised forever. It raises serious privacy and security concerns regarding the individual's assets, including but not limited to financial and identity loss. The use of cryptographic primitives such as encrypting or hashing the biometric data for biometric template protection is not always a secure and feasible mechanism. In the context of biometric security, presently, we are witnessing extensive research efforts and, in parallel, standardization activities towards biometric template protection.

Thus, in this thesis, we analyzed, **"how can we ensure that our biometric data is protected on the database servers, particularly in the large scale biometric systems?"**. Specifically, we tried to answer the questions like— What is the trade-off between biometric security and recognition performance? To what extent can we preserve the number of bit errors in the protected biometric templates? How can we ensure secure authentication in the multi-biometric systems, and can we achieve the desired security bound while preserving the recognition performance?

We begin with the design of biometric authentication systems using the biometric cryptosystems or biocryptoystems that are gaining prominence for cryptographic key generation, encryption, and biometric template protection. The fuzzy commitment and fuzzy vault are the most popular state-of-the-art biocryptosystems. However, they are prone to multiple security attacks. Recently proposed multi-biometric biocryptosystems improve security and enhance recognition performance. They perform the fusion of multi-biometric characteristics with either a single biocryptosystem or independently accessed multiple biocryptosystems, which are vulnerable to security attacks. Thus, in our first work, we tried to increase the security bound of existing schemes that is given as $(2^{|K_1|+|K_2|})$ to get the ideal security bound equal to $(2^{|K_1|\times|K_2|})$, where K_1 and K_2 are the underlying security parameters. We propose a multi-biometric fusion framework-BIOFUSE, that combines fuzzy commitment and fuzzy vault using the format-preserving encryption scheme. BIOFUSE makes it improbable for an attacker to get unauthorized access to the system without impersonating all the genuine user's biometric inputs at the same instant.

The biometric cryptosystems require error correcting codes to correct the bit errors present in the biometric templates. However, the error correcting codewords have limited error correcting capability, making it inconvenient and infeasible to use multi-biometric biocryptosystems for the wide scale scenarios. As a part of our second contribution, we focused on the cancelable multi-biometric authentication approach, where the transformation of the original biometric

template to a protected template is done using a secret parameter. In this context, we introduced a novel bit-wise encryption scheme. It transforms the biometric template into a protected template using a secret key generated from another biometric template of the same user. The key is generated using fuzzy extractor. Unlike the existing cancelable schemes, the bit-wise encryption scheme fully preserves the number of bit-errors in the original and the protected template. The results of comparisons with the existing biometric template protection schemes on the various face and iris databases show that the proposed work provides significantly good recognition performance and efficiency while achieving high security.

In the above discussed approaches, the overall system's performance depends on the underlying fuzzy commitment scheme that uses random error correcting codeword. The two major requirements of a fuzzy commitment scheme are (i) the length of the biometric template is equal to the length of codeword generated by the error correcting code, (ii) high error correcting capability. In general, these requirements are satisfied by padding with extra bits on the input biometric template. However, the fixed padding approaches proposed in the literature have a security vulnerability that could disclose the user's biometric data to the attacker, leading to an impersonation attack. We propose a user-specific, random padding scheme that satisfies the requirements mentioned above while preventing the impersonation attack. Our empirical results show that the proposed scheme provides 3 times better recognition performance than the baseline, unprotected systems. Despite the trade-off between the performance accuracy and security, our proposed scheme, compared with existing schemes, provides significantly better recognition performance and efficiency while preserving the overall system's security.

To my Mother,
for her unconditional love and unfaltering belief in me.

Acknowledgements

"Nine-tenths of education is encouragement. – Anatole France

First and foremost, I express my heartfelt gratitude to my esteemed Ph.D. advisor, Dr. Donghoon Chang, for his invaluable guidance and persistently inspiring me to bring out the best me. He encouraged me to learn and gain knowledge throughout my Ph.D. journey. I am much indebted to him for his amazing support and constant motivation throughout my Ph.D. tenure. Not only did he guide me in my research work, but he also influenced me to be a better human being. I am incredibly fortunate to have a friend, philosopher, and guide in the form of my Ph.D. advisor, who pushed me to have a positive outlook towards my work and life. Without his enlightenment, this journey would not have been the same.

I also express my sincere gratitude to Dr. Somitra Sanadhya, who immensely helped me throughout my Ph.D. duration by being available to give a direction at times of professional chaos. His vision, encouragement, and constant support helped me complete my thesis work efficiently.

I appreciate the efforts of my Ph.D. committee members, Dr. A.V. Subramanyam and Dr. Sambuddho Chakravarty, who helped me improve my work by providing their valuable feedback and comments.

I would like to thank my PhD thesis examiners, Prof. Mridul Nandi, Prof. C. Pandu Rangan and Prof. Sourav Mukhopadhyay for their valuable comments and suggestions which helped me improve my dissertation.

I take this opportunity to thank all the members of my Crypto Lab for creating a research-friendly environment. They made my research life at IIIT-Delhi more accommodating and productive. I am grateful to my mates Munawar, Sweta, and Mohona for the inspiring and fruitful collaborations I had with them.

I would especially like to acknowledge the cooperation and support of my friends in IIIT-Delhi, especially Dhriti, Megha, Ishant, Pravin, Devashish, Monika, and Venketesh, for keeping my spirits high at all times. Their remarkable comradeship made my graduate life truly memorable.

I would also like to forward my gratitude to Tata Consultancy Services (TCS), India, for awarding me the prestigious TCS fellowship for my full Ph.D. period. Additionally, I thank Ms. Priti from the IIIT-Delhi administration department for her instrumental role in making my journey more manageable and less stressful.

On a personal front, I owe my heartfelt thanks to my parents, brother, sister, cousins and my

in-laws for their unconditional support, understanding, and encouragement, without which this dissertation would not have been completed. Finally, special thanks to my dearest husband, who understand my priorities and motivated me to focus on my PhD work to achieve my targets.

Above all, I am grateful to the Almighty, who bestowed upon me the opportunities and His blessings for giving me the strength and courage to complete this dissertation.

Research Papers From This Work

1. Cancelable Multi-biometric Approach using Fuzzy Extractor and Novel Bit-wise Encryption.
   Donghoon Chang, **Surabhi Garg***, Munawar Hasan, Sweta Mishra. IEEE Transactions on Information Forensics and Security **TIFS** 2020.

2. BIOFUSE: A Framework For Multi-Biometric Fusion On Biocryptosystem Level.
   Donghoon Chang, **Surabhi Garg***, Mohona Ghosh, Munawar Hasan. Elsevier **Information Sciences** 2021.

3. An Improved Construction of Fuzzy Commitment Scheme for Biometric Authentication.
   Donghoon Chang, **Surabhi Garg***, Munawar Hasan, Sweta Mishra. Elsevier **Computers & Security** 2020. Under Submission.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The traditional means of user authentication, such as passwords and hardware tokens, have serious security concerns that prompt the widespread implementation of biometric authentication systems. Biometric authentication [5] refers to uniquely recognizing a person based on their physiological or behavioral characteristics. Recently, multi-biometric systems have been proposed [6, 7, 8, 9] for user authentication where more than one biometric characteristics are used to authenticate users. These systems are being increasingly used in large-scale applications because of their advantages, such as they provide low error rates by reducing the inter-class similarity, which in turn improves the accuracy and reliability of biometric systems. Further, they provide resilience against spoof attacks [10, 11] since it is difficult to spoof more than one biometric characteristics rather than a single biometric characteristic. The Government of India has implemented the Aadhar project [12] (a multi-biometric database of India's population of over 1.3 billion) for authentication that captures and store multiple traits (face, fingerprints, and iris) of the population of India and issues a unique identification number (UID) to each individual. With the use of biometrics for authentication purposes on a broad scale, the need for biometric data security has become inevitable.

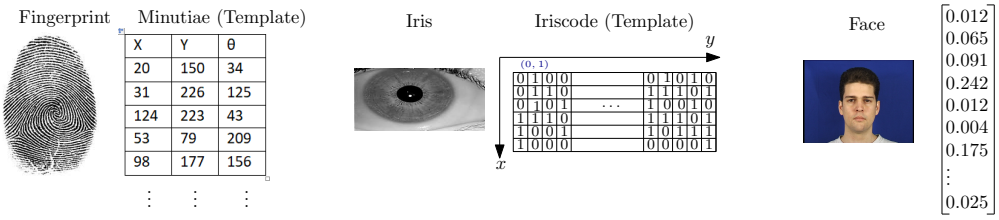


Figure 1.1: Diagram depicting fingerprint and iris templates

1.1 Biometric Authentication System

A general biometric authentication system consists of two phases. During the enrolment phase, the user provides the biometric characteristics from which features are extracted. These features, also known as a biometric template, are shown in Figure 1.1. The template comprises unique features extracted from an individual's biometric characteristics using sensors that capture biometric data in the form of a digital image. For example, a fingerprint template consists of ridge endings and ridge bifurcation points that represent minutiae points [13] denoted by coordinates (X, Y, θ). Similarly, an iris template is a binary string generated from the patterns found in iris texture [14].

During the authentication phase, the comparison is performed between the query and the stored enroled template to know if the two templates match or do not match with each other based on a comparison score.

A typical biometric system consists of four basic modules:

1. a biometric sensor module that captures input biometric characteristics in the form of raw or digital data
2. a feature extraction module that extracts distinguishable, unique features from the digital biometric data
3. a comparator module where the actual comparisons on biometric templates take place between the biometric template stored on a server and the provided query template
4. a database server where the biometric templates are stored during the enrolment phase

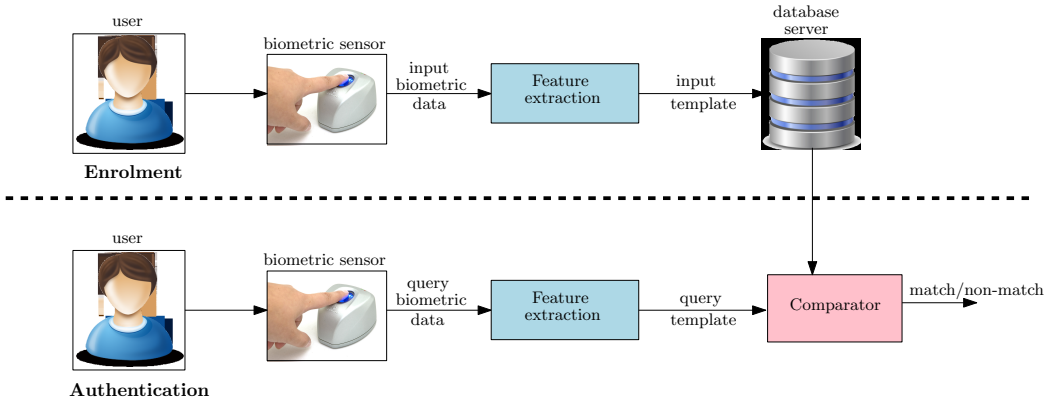


Figure 1.2: Block diagram showing the Biometric system. It consists of two phases- the enrolment and the authentication phase.

A block diagram representing the biometric system is shown in Figure 1.2.

1.1.1 Growing Security Concerns

The advancement of biometrics in the widespread deployment of authentication systems such as India's Aadhaar project emphasizes the need for biometric systems with high recognition performance and the protection of biometric data referred to as biometric templates. Additionally, the biometric systems are highly vulnerable to presentation attacks or spoofing attacks [15]. The study and mitigating of such attacks are beyond the scope of this thesis. Typically, in the biometric systems, the biometric templates are stored on servers in their original, unprotected form during the enrolment or the registration phase. In such a case, the stored, unprotected template could be stolen or modified by an attacker [16, 17], which raises serious security and privacy concerns. Further, it is possible to recover the original biometric samples from the unprotected templates [18, 19, 20]. Unlike passwords, it is impossible to change the biometrics; therefore, once compromised, biometrics are lost forever. European Union (EU) General Data Protection Regulation 2016/679 [21] has classified biometric data as sensitive information with access to such data subject to the right to privacy.

1.1.2 Goals of a Secure Biometric System

Ideally, a secure biometric system follows the following privacy properties, as mentioned in ISO/IEC IS24745:2011 [22]. These properties are stated as follows.

- **Irreversibility or non-invertibility** states that it should be computationally infeasible to obtain the original biometric template from the protected or transformed biometric template.
- **Renewability** states that it should be possible to cancel or revoke the existing compromised protected template and re-issuing a new instance of the protected biometric template.
- **Unlinkability** states that given different templates of the same user, the attacker should not be able to guess whether the two templates are derived from the same user or not.

In addition to these properties, the biometric recognition performance measured in terms of false match rate and false non-match rate should be high.

1.1.3 Biometric Template Protection Approaches

The security concerns and privacy requirements related to a biometric authentication system bring about the need for biometric template protection schemes that could preserve the overall system's security while providing high recognition performance. The simple and naive cryptographic approaches to protect the biometric template could be encryption and hashing. Encryption requires an additional key or a password, which is a security overhead. Further, the encrypted template needs to be decrypted during the authentication phase, which could leak some information about the original biometric template. Regarding the hashing approach, any two samples of the same biometric instance are never the same. Thus bit-errors always exist in the biometric templates, which makes one-way hashing in biometrics infeasible. Thus following conclusions are deduced:

- A system's generated key or user's provided password or a key used for encryption of biometric template is not sufficient for system's security.

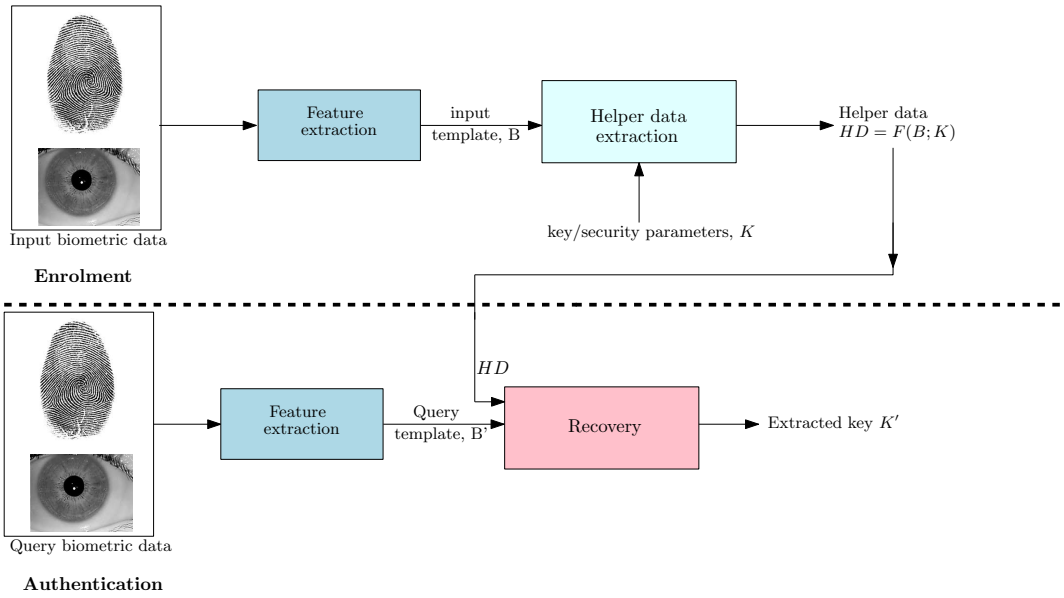


Figure 1.3: Block diagram showing enrollment and authentication phases of biometric cryptosystems. HD denotes the helper data. The key $K = K'$ if the query template matches with the enroled (input) biometric template.

- In the case of loss of the underlying key/password or the protected template itself, re-registration of biometric templates is required, which is not easy in the case of biometrics, unlike passwords.
- A biometric generated key could help in enhancing the security of biometric systems.

Recently, biometric and multi-biometric template protection schemes have been introduced [23, 24, 17, 25] to hide any sensitive information about the original templates. These are broadly categorized as biometric cryptosystems or biocryptosystems, cancelable biometrics, and homomorphic encryption schemes. In biocryptosystems [26, 1, 27, 28, 29], the original template is replaced by the biometric-dependent helper data. Instead of storing the original biometric template on the server database, the helper data is stored as a public value on the server database. Based on this helper data, biocryptosystems are classified into two categories: key-generating biocryptosystems, where the helper data is derived from the biometric template. The cryptographic key is generated from the helper data along with the biometric query template provided during authentication. If the query template and the enroled template match, the same cryptographic key will be generated with high probability. The fuzzy extractor is an example of a key-generating biocryptosystems. Another category includes key-binding systems in which the

helper data is obtained by binding a cryptographic key with the original biometric data. During the authentication phase, the helper data helps recover cryptographic keys without revealing any information about the original biometric template. Like the key-generating biocryptosystems, it ensures that the same key is recovered with a high probability if the comparison is successful. The two popular key-binding biocryptosystems are the fuzzy commitment scheme and the fuzzy vault scheme. The biocryptosystems are formally defined in Chapter 2. A block diagram showing enrolment and authentication phases of biometric cryptosystems is shown in Figure 1.3.

The biocryptosystems use the random error correcting codes to correct the bit-errors present in the biometric templates. The use of error correcting codes limits the applicability of biocryptosystems on a wider scale. A random message of a predefined length generated by a pseudorandom number generator is used to derive the error correcting codes [1, 30]. The fuzzy commitment-based biometric system's security depends on the length of this random message. Usually, in most of the fuzzy commitment based biometric systems [31, 32], the length of the random message is kept small to correct a large number of bit-errors that leads to the possibility of a security vulnerability [33]. Similarly, the fuzzy vault scheme suffers from various security attacks [34] such as brute force attack on the stored vault, correlation attack, blend substitution attack, and key inversion attack. Further limitations are discussed in Chapter 3.

Cancelable biometrics [16, 35, 36] is another biometric template protection approach in which a transformation function dependent on a key or a password is used to transform the original template into a protected template known as a cancelable template. The comparisons between the two templates are performed in the transformed or protected domain. Cancelable biometric schemes aim at non-invertible, alignment free, and revocable templates using a secure parameter such that even if a transformed template is compromised, it can be re-allocated to the same user for a particular application. Thus a secure cancelable biometric template protection scheme satisfies the privacy properties of a biometric system that includes: irreversibility, unlinkability, and renewability. Cancelable biometric approaches are mainly classified into the following categories: non-invertible transforms and biometric salting. These approaches are discussed in detail in Chapter 4. Compared to the unprotected (denoted as baseline) biometric systems, recognition performance degrades in cancelable templates due to noise in the form of bit-errors

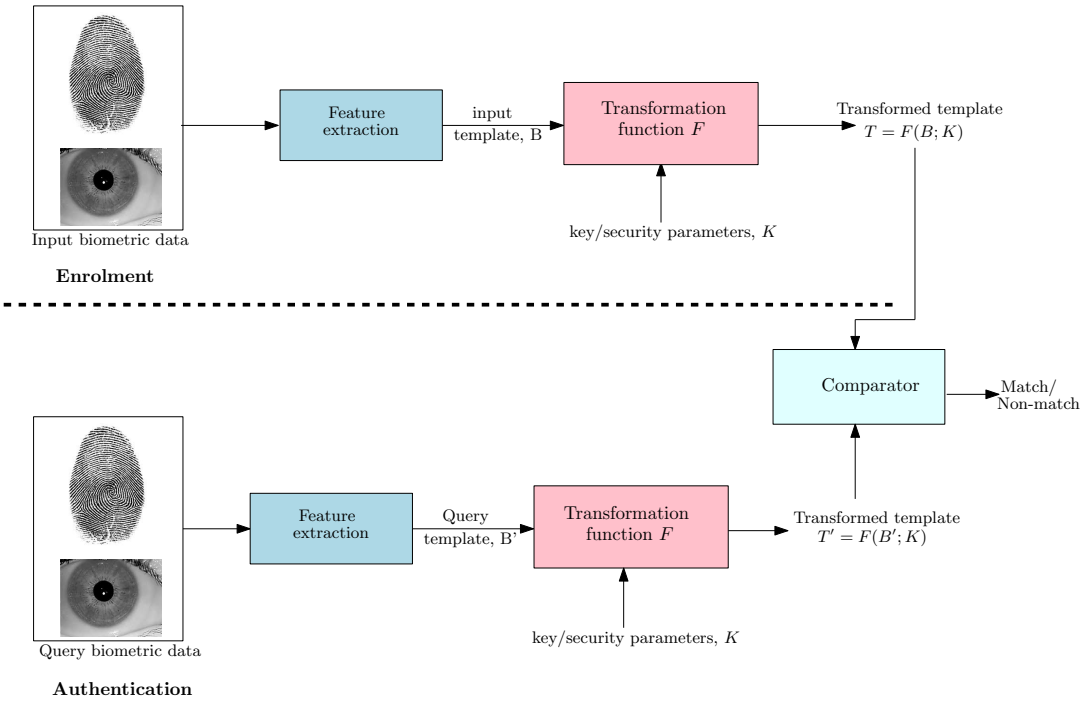


Figure 1.4: Block diagram showing enrolment and authentication phases of cancelable biometrics

added during the transformation. A block diagram showing enrolment and authentication phases of feature transformation systems or cancelable biometrics is shown in Figure 1.4.

In the homomorphic encryption schemes [37, 8, 38], the comparisons for authentication are performed on the encrypted biometric templates. These schemes provide high security but take huge computational time, which restrains such schemes' feasibility in practical applications. Further description of these schemes with their limitations is given in Chapter 4.

1.1.4 Thesis Contribution

The existing biometric template protection schemes transform the biometric templates into a secure transformed form. However, they have multiple security concerns that hinder the acceptability of biometric authentication on a larger scale. Further, the requirements mentioned above of a secure biometric authentication system to transform the biometric data into the secure, non-invertible, unlinkable, and renewable template without any user provided security parameter in the form of a key or a password is still unanswerable. Thus, it is natural to ask– how to ensure that our biometric data is stored safely on the database servers, particularly in the

wide-scale biometric systems? More specifically, how can we provide secure authentication in the multi-biometric systems? In context, the thesis broadly looks into the following research questions.

1. **Can we design a multi-biometric multi-biocryptosystem which could provide secure authentication?**

   In other words, what are the all possible ways of designing the architecture in which the two popular biocryptosystems– fuzzy commitment scheme and fuzzy vault scheme could be combined? What is the ideal security bound of such a system? Does it achieve the ideal bound? We can find answers to these questions in Chapter 3.

2. **Is it possible to design a cancelable multi-biometric scheme that could fully preserve the bit-errors present in the biometric template?**

   To be precise, is it feasible to design a scheme that ensures that the number of bit errors remains preserved during the transformation of the original biometric template to generate the protected biometric template? Further, we also looked upon whether it is possible to perform the transformation without using additional security parameters such as a key or a password? What is the security bound in such a system? We address these concerns in Chapter 4.

3. **To what extent, we can preserve the trade-off between security and performance in biometric cryptosystems?**

   A literature study shows that there always exists a trade-off between security and performance. The biocryptosystems use error correcting codes that correct more bit errors in the biometric template when the security parameter is chosen from a low-security range, such as a parameter with less than 128 bits. Thus, the recognition performance increases if we decrease the security of our system and vice versa. Further, the efficiency (in terms of time taken) depends on implementing the underlying crypto-primitives in the biocryptosystems or cancelable biometric schemes. Can we improve the security of the fuzzy commitment scheme while preserving the performance? Is there the need for any additional key or a password to enhance the security? We address these questions in Chapter 5.

To begin with the motivation of designing a multi-biometric system with secure authentication, we explored all the architectures of multi-biometric biometric cryptosystems, which are possible by the combination of the fuzzy commitment and the fuzzy vault scheme. We found one secure design that provides the ideal security bound of $2^{|K_1|+|K_2|}$. However, we observed that the implementation of biometric cryptosystems has limited applicability in terms of recognition performance. Hence, we designed a cancelable multi-biometric approach that uses a fuzzy extractor (involves a fuzzy commitment scheme with a key generation module) and a novel bit-wise encryption scheme. It completely preserves the bit errors in the protected biometric template while protecting the overall system's security. To efficiently implement the underlying fuzzy commitment scheme in both the mentioned approaches, we work on a random padding approach for the fuzzy commitment scheme so that performance is highly improved while preventing the existing impersonation attacks in similar schemes with fixed (or zero) padding approaches.

Using the empirical results and the mathematical proofs in the thesis, we showed that the proposed work provides significantly good recognition performance and efficiency while it achieves high security. Although we provide a secure multi-biometric authentication system where authentication maps to users' verification, the identification aspect in the biometric systems is still a challenging problem that needs to be solved. Additionally, our proposed solutions can be built over the commercial-off-the-shelf biometric systems to achieve the equivalent performance with high security and broader scalability.

Chapter 2

Preliminaries

We use several concepts and functions in our constructions, as discussed below. We discuss some fundamental concepts that are used throughout our work in the following subsections.

2.1 Notations

B denotes the original biometric template represented in binary format. C denotes the protected template or cancelable template. It could be interchangeably used as a ciphertext. (x, y) are the x and y-coordinates in the biometric template that represent the bit-wise position for a particular bit as $b_{(x,y)}$, K denotes a random, cryptographic key of length k, W denotes the error correcting codeword, H denotes the hash function, HD denotes the helper data generated by the fuzzy extractor. r represents the random number generated from a pseudo-random number generator.

2.2 Metric Space

1. **Hamming Weight Metric:** Given biometric data in the form of a binary string, the hamming weight of binary string is defined as the number of ones in that string. The Hamming distance between two biometric strings B and B' is defined as the number of bit positions in which the two strings differ and is denoted as $d(B, B') = \|B \oplus B'\|$. $\|.\|$

denotes the L0 norm.

2. **Symmetric Difference Metric:** For any two sets consisting of elements x, where the sets represents the biometric template denoted as B and B', the symmetric difference is defined as the $|\{x \in B \cup B' | x \notin B \cap B'\}|$.

2.3 Key Generating Biocryptosystems

The key generating biocryptosystems are introduced in [1]. The fuzzy extractor and secure sketch are key generating biocryptosystems that generates a cryptographic key using the biometric template and the helper data.

2.3.1 Secure Sketch

A secure sketch is given by two procedures [1], sketching procedure, SS and recovery procedure, Rec:

1. Sketching procedure SS takes a n-bit string B and a random error correcting codeword W. It returns a n-bit helper string S known as a secure sketch value.

2. Recovery procedure Rec takes a n-bit string B' and a public, secure sketch value S. The correctness property of secure sketches guarantees that if $\|B \oplus B'\| \leq t$ where t is the maximum number of errors that can be corrected in the bit string by the error correcting codeword, then $Rec(B', SS(B)) = B$, else there is no guarantee for the value of the output.

2.3.2 Fuzzy Extractor

A fuzzy extractor is given by two procedures [1]- Generation, Gen and Reproduction, Rep.

1. Generation Gen procedure consists of sketching procedure of secure sketch and a strong extractor. A strong extractor outputs a highly uniformly distributed, randomized string

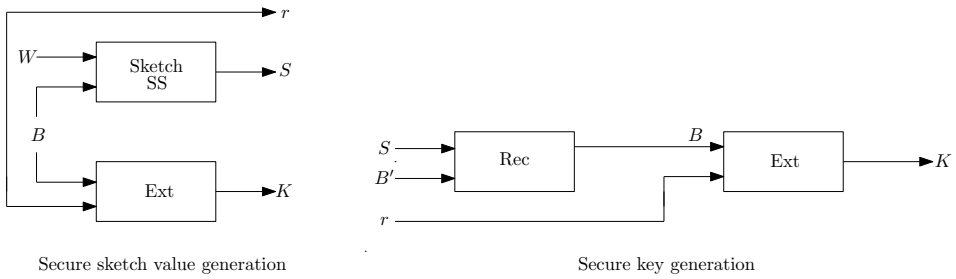


Figure 2.1: Fuzzy extractor construction from secure sketch [1]. Here, SS denotes the sketching function that generates secure sketch value S. Ext denotes the extractor that generates a strong cryptographic key K from the non-uniform input binary string B. W is the random error correcting codeword and r denotes the random number.

independent of the random input string of low entropy. A universal hash function $H : \{0,1\}^n \longrightarrow \{0,1\}^l$ can be taken as an example of a strong extractor. The sketching procedure SS on input $B \in \mathcal{M}$ generates a secure sketch value $S \in \{0,1\}^*$. $\mathcal{M}$ denotes the Hamming weight metric space. The extractor takes the input B and a random number r to output a uniformly random string $K \in \{0,1\}^l$. This K represents a secure key that is used for user authentication. For example, one way of generating the key with $\|$ as the concatenation operator is given as

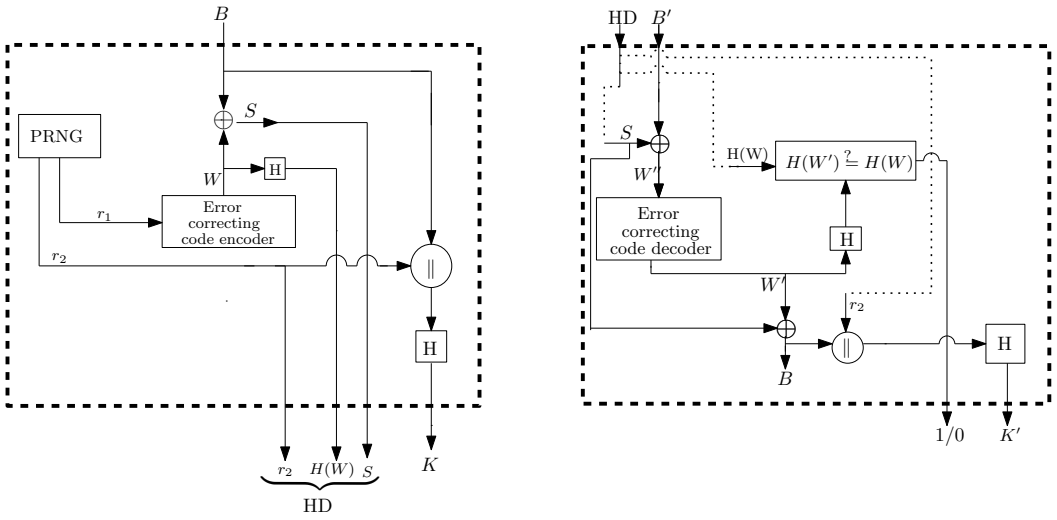
$$K \longleftarrow H(B\|r)$$

2. Reproduction Rep procedure takes S generated from secure sketch and another string $B' \in \mathcal{M}$ as inputs. The correctness property of fuzzy extractors guarantees that if $\|B \oplus B'\| \leq t$ and $(K, S) \longleftarrow Gen(B)$, then $Rep(B', S) = K$ else there is no guarantee about the output. The original input B is recovered, which helps to derive the same key K.

Construction of fuzzy extractor from the secure sketch is shown in Figure 2.1.

2.4 Key-Binding Biocryptosystem

Key-binding systems generate helper data by binding cryptographic key with the biometric template. If a similar biometric template is provided during the authentication phase, the same cryptographic key is recovered using the helper data. The popular constructions are fuzzy vault



$(a) Enrolment : Commitment(B) = (HD, K)$ $(b) Authentication : De-commitment(HD, B') = (1/0, K')$

Figure 2.2: Enrolment and authentication phase of the fuzzy commitment scheme. The enrolment module takes input B and a random codeword W to generate a secure K and helper data HD. The authentication module recovers back the biometric template B with the help of helper data, and biometric template B' and key K' is derived. If the hamming distance between B, B' is below a pre-defined threshold level, the key $K' = K$.

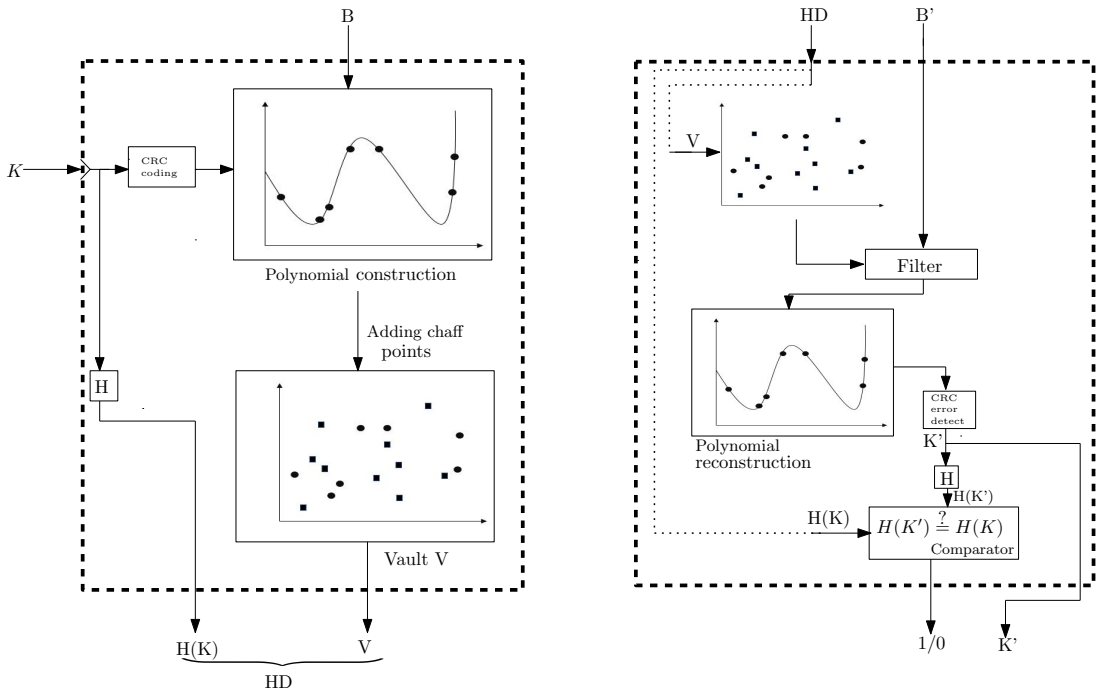
and fuzzy commitment that are discussed in the subsequent subsections.

2.4.1 Fuzzy Commitment Scheme (Hamming weight Metric)

It was introduced by A. Juels and M. Wattenberg in 1999 [27]. The block diagram is shown in Figure 2.2. A fuzzy commitment (FC) is a pair of two functions- commitment function and a de-commitment function with the following properties:

1. A commitment function on input $B \in \{0,1\}^n$, selects a random error correcting code $W \in \{0,1\}^n$ and returns the secure sketch value denoted as $S \in \{0,1\}^n$ such that $S = B \oplus W$. The error correcting code is generated with the help of a random message denoted as r_1. The helper data HD constitutes $(r_2, H(W), S)$, where r_2 is a random message used to generate cryptographic key from B (as a part of fuzzy extractor).

2. The de-commitment function takes a n-bit query template B' and the helper data HD. It computes $(W'' \oplus (B \oplus B'))$. Provided efficient decoding of error correcting codewords, if $\|B \oplus B'\| \leq t$, where t is the maximum number of errors corrected in the bit string, W'' is decoded to the nearest codeword W'. If $(H(W') = H(W))$, the de-commitment is



(a) Enrolment Module

(b) Authentication Module

Figure 2.3: Enrolment and authentication phase of fuzzy vault scheme. The enrolment module that takes input B and a system's provided secret key K to generate a public vault V. The authentication module recovers the key K' with the help of vault and biometric template B'. If the symmetric difference between B, B' is below a pre-defined threshold level, the key $K' = K$.

successful. With the fuzzy extractor's help, the original B is recovered, and the key K is generated, which is the same as the key generated during the enrolment.

2.4.2 Fuzzy Vault Scheme (Symmetric Difference Metric)

The fuzzy vault introduced by Juels and Sudan in 2002 [26] is generated by encoding the biometric features on a polynomial using the key as coefficients of a polynomial. If a similar biometric feature set of the same user overlaps with the original set, the key is decoded using the vault. The block diagram is shown in Figure 2.3.

Let a biometric template B be denoted as an unordered set of l well separated features: $B = \{B_1, B_2, \ldots, B_l\}$. A secret key K is splitted into n parts given as $K = K^0 K^1 K^2 \ldots K^{n-1}$ satisfying $n \leq l$. A CRC code is appended to secret key to get a new secret having $(n+1)$ parts. Using the secret key as coefficients of polynomial denoted by $p(B_i) = K^n B_i^n + K^{n-1} B_i^{n-1} + K^{n-2} B_i^{n-2} + \ldots K^1 B_i^1 + K^0$ for $1 \leq i \leq l$, a genuine set $G = \{(B_i, p(B_i))\}_{i=1}^{l}$ is generated

from each feature in the feature set B. To hide the genuine points from an attacker, random chaff points are added to the vault. A chaff point set $Ch = \{(z_i, u_i)\}_{i=l+1}^{r}$ is generated such that $z_i \notin \{B_i\}$ and $u_i \notin \{p(B_i)\}$. The genuine point set and chaff point set constitute a vault with total r points denoted as $V = G \cup Ch$.

During authentication, if a sufficient number of points (candidates) in the query template overlaps with the biometric feature points in the vault, the polynomial of degree n is reconstructed, and the secret key K' is recovered. The CRC error detection is applied [39] to K', and the authentication step is repeated for a new set of candidates in case of error is detected. If no error is detected, it indicates that $K = K'$ with very high probability.

2.4.3 Error Correcting Codes

Error correcting codes are extensively used in the information theory and coding theory to provide reliable transmission of digital data over the unreliable communication channel. An application of error correcting codes could be seen in the area of biometrics where the multiple biometric samples of a particular individual are similar but not identical (in terms of the bits). Noise, in the form of bit errors is present in these multiple samples due to various atmospheric factors (lightness, etc.), pressure, head rotations, etc. at the time of capturing of biometric samples. Error correcting codes help to correct the bit errors present in the biometric samples.

Error correcting codes map k-symbols to n-symbols such that

$$\sum^{\delta} \longrightarrow \sum^{n}$$

, given the mapping is an injective function.

Here,

- $|\sum|$ is the cardinality of alphabet set. In our work, we consider $|\sum| = 2$ which depicts a binary code.
- δ is the input message length

- n is the length of output, such that $k < n$

Each error correcting code is a set of unique, random codewords denoted by W. One of the prominent examples of error correcting codes used in the biometrics is BCH Codes.

2.4.4 BCH Codes

Bose-Chaudhuri-Hocquenghem codes are random error-correcting cyclic codes constructed using polynomials over a finite field (also called Galois field). For any positive integers $(q \geq 3)$ and $(t < 2^{(q-1)})$, there exists a t-error-correcting BCH code generated with the random, secret message of length δ with the following parameters [40, 41]:

Block length: $n = 2^q - 1$

Number of parity-check bits: $n - \delta \leq qt$

Minimum distance: $d_{min} \geq 2t + 1$

Each error correcting codeword present in the set has the mentioned configuration properties. The codeword W is denoted as $(n, \delta, t)-$ BCH codeword. The block length gives the size of the error correcting codeword. The parity-check bits are used to recover the original, transmitted codeword from a received codeword with some error(s). Minimum distance gives the minimum distance between any two codewords, such that each codeword corrects a maximum of t error bits.

2.4.5 Format Preserving Encryption

Format preserving encryption (FPE) algorithms form a class of cryptographic encryption schemes that protect the format of data after encryption, i.e., the encrypted data has the same format and length as the original, unencrypted data. Black and Rogaway [42] made the first systematic study of for format preserving encryption approach. Following are the encryption and decryption functions used in the format preserving encryption scheme (FF1 being used in our work).

- Encryption function, FP.Enc: takes inputs as a key $K \in \mathcal{K}$ where $\mathcal{K}$ defines the key-space, a random value known as tweak T and the plaintext M to output the ciphertext C given as

$$C \longleftarrow FP.Enc(K, T, M)$$

- Decryption function, FP.Dec: takes input as key K, a random value known as tweak T and the ciphertext C to compute the plaintext M given as

$$M \longleftarrow FP.Dec(K, T, C)$$

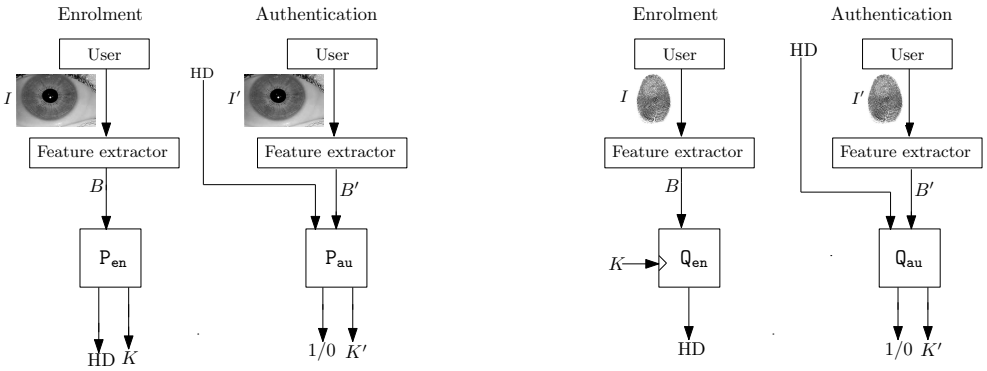
The special publication of NIST SP800-38G [43] specifies three modes of operation for format-preserving algorithms: FF1, FF2, and FF3. These modes employ an unbalanced Feistel structure and use the AES-128 algorithm as the internal round function. FF1 and FF2 invoke the AES-128 algorithm at least 11 times, and FF3 invokes it eight times, thus leading to a high number of AES-128 invocations [44].

Chapter 3

BIOFUSE: A Framework For Multi-Biometric Fusion On Biocryptosystem Level

Biometric cryptosystems or biocryptoystems are gaining prominence for cryptographic key generation, encryption, and biometric template protection [17, 45, 46]. However, the most popular state-of-the-art biocryptosystems- fuzzy commitment and fuzzy vault are prone to multiple security attacks [31, 32, 34]. The fuzzy commitment scheme uses random error correcting codewords. An attack has been proposed [33, 30] on the codewords where if a few bits of a codeword or biometric template are revealed, it can reveal the whole biometric template to the attacker. Further, the fuzzy vault scheme is vulnerable to a brute force attack possible on the public vault [47, 39]. Additionally, it suffers from other security attacks [34] such as correlation attack, blend substitution attack, key inversion attack, etc. These are discussed in detail in Section 3.4.

Lately, multi-biometric biocryptosystem based authentication systems have been proposed in [48, 6, 46, 49], where multiple biometric templates are given as inputs to the biocryptosystem(s). These schemes require combination or fusion of multiple biometric characteristics, which is generally done either on the feature level, score level, or decision level. Multibiometric



(a) Fuzzy Commitment (FC)

(b) Fuzzy vault (FV)

Figure 3.1: General overview of enrolment and authentication phases in biocryptosystems. (a) A module $\mathtt{P}_{\mathtt{en}}$ of FC scheme takes biometric template B extracted from biometric characteristics I. It generates a secret, cryptographic key K and a helper data HD. During authentication using another sample B' and HD, key K' is generated from module $\mathtt{P}_{\mathtt{au}}$. (b) Module $\mathtt{Q}_{\mathtt{en}}$ of FV scheme takes biometric template B and a system's generated key K to give helper data HD. The helper data helps to recover K' from B' using module $\mathtt{Q}_{\mathtt{au}}$. The internal workings of these modules are shown in Figure 2.2 and 2.3.

biocryptosystems improve the accuracy during authentication [50] as well as they improve resilience to spoof attacks since it is difficult to spoof multiple biometric characteristics of the user simultaneously.

The existing multibiometric biocryptosystems are implemented in two ways. A single biocryptosystem is implemented where multibiometric characteristics are given as inputs with an existing fusion approach [49]. Such systems are prone to several existing drawbacks of the state-of-the-art biocryptosystems, i.e., fuzzy commitment and fuzzy vault. Another approach is the implementation of multiple biocryptosystems [51], in which one or more biometric characteristic is given as input(s) to each biocryptosystem involved. Each of these biocryptosystems can be accessed independently of each other. In the latter case, the major disadvantage is that even if one of the underlying biocryptosystems is compromised by an attacker, the security of the whole system breaks. In other words, an attacker can attack both (in case if 2 biocryptosystems are involved) the biocryptosystems independent of each other. In the worst case, when the attacker does not have any information about the input biometric templates, the ideal security bound of a system with multi-biocryptosystems (2 in our case) involved is computed in terms of the number of trials needed to perform brute force attack.

Given K_1 and K_2 as the respective security parameters of the two underlying biocryptosystems, the ideal or desired security bound in terms of the number of trials is equal to $2^{|K_1|} \times 2^{|K_2|}$

(or $2^{|K_1|+|K_2|}$). In the latter approach, the exhaustive search attack (in the worst case) on the security parameters- K_1 and K_2, leads to overall security bound equal to $2^{|K_1|} + 2^{|K_2|}$ or $2^{min(|K_1|,|K_2|)}$ which is much lower than the desired security bound. Further limitations are stated in Section 3.1. Thus, there is a need for a secure design of multibiometric biocryptosystem which could meet the desired security level. In this regard, we focus on the following research questions.

- What are all possible ways of designing the architecture in which the fuzzy commitment scheme and fuzzy vault scheme could be combined?
  - Is the ideal security bound of $2^{|K_1|+|K_2|}$ achieved in the proposed architecture?
- Is there a trade-off between the security and the recognition performance?

To answer the aforementioned questions, we apply an exhaustive search to explore all the possible ways to combine the two biocryptosystems- fuzzy vault and fuzzy commitment with the fusion of multi-biometrics at a biocryptosystem level. Biocryptosystem level denotes that the fusion of biometric characteristics is done by combining the two biocryptosystems such that accessing any of them independently of others is not a feasible option. We found a total of 84 combinations in which a fuzzy commitment scheme can be combined with a fuzzy vault scheme. For simplicity, we considered only the most basic ways of combination. We take the input and output parameters of fuzzy commitment and fuzzy vault scheme, as shown in Figure 3.1. We then construct all the possible cases by performing XOR, concatenation operation, an encryption function between the chosen two parameters; one parameter from each biocryptosystem.

Following are the observations from Table 3.1:

- Helper data generated from the fuzzy vault or fuzzy commitment scheme or the biometric template that denotes the biometric features in a set form (we denote it as B_{FV}), for an example- minutiae points, cannot be XORed or concatenated with another helper data, key or a biometric template due to the difference in formats. Similarly, other parameters that are of different formats cannot be combined using concatenation or XOR operations. A symbol ✗shows these cases.

Table 3.1: All possible 84 cases in which 2 biocryptosystems can be combined. FC and FV in subscripts represent that the particular parameter belongs to fuzzy commitment or fuzzy vault respectively. $\oplus$ represents XOR operation, $\|$ denotes concatenation and E_K represents encryption by block cipher modes of operation using a key K. ✗and ✓denotes if the case is not possible or possible, respectively. The meanings of symbols a and b are discussed in the section below.

Choice of first biocryptosystem	**Choice of operation**	**Choice of second biocryptosystem**					
		B_{FC}	HD_{FC}	K_{FC}	B_{FV}	K_{FV}	HD_{FV}
B_{FC}	$\oplus$	a	✗	a	✗	✗	✗
	$\|\|$	a	✗	a	✗	✗	✗
HD_{FC}	$\oplus$	✗	✗	✗	✗	✗	✗
	$\|\|$	✗	✗	✗	✗	✗	✗
K_{FC}	$\oplus$	✗	✗	a	✗	✗	✗
	$\|\|$	✗	✗	a	✗	✗	✗
	$E_{K_{FC}}$	b	✓(S_1)	a	b	✗	✓(S_3)
B_{FV}	$\oplus$	✗	✗	✗	✗	✗	✗
	$\|\|$	✗	✗	✗	✗	✗	✗
K_{FV}	$\oplus$	✗	✗	a	✗	✗	✗
	$\|\|$	✗	✗	a	✗	✗	✗
	$E_{K_{FV}}(.)$	b	✓(S_4)	a	b	✗	✓(S_2)
HD_{FV}	$\oplus$	✗	✗	✗	✗	✗	✗
	$\|\|$	✗	✗	✗	✗	✗	✗

- Biometric template in the binary string format (we denote it as B_{FC}), for example- iriscode or face can be concatenated with another similar format template or the key generated from the fuzzy commitment scheme (we denote it as K_{FC}). However, it provides no extra security to the system. Such cases are represented by the symbol a.

- XOR or concatenation operation between the keys that belong to two biocryptosystems or encryption of one key using the other provides no extra security since the transformed key after XOR or encryption has no role in the authentication. Such cases are represented by the symbol a. Note that we consider the key used in the fuzzy vault scheme (we denote it as K_{FV}) as the system's generated internal key; hence it cannot be transformed by any of the operations.

- It is not possible to encrypt any of the biometric template using the keys since encryption on similar but not identical biometric template during authentication would completely change the data, leading to authentication failure. We represent such cases by symbol b.
- The four possible cases (S_i) shown by ✓symbol are described in Section 3.3.

Out of the 84 cases we found, the most feasible way of combining two biocryptosystems is only by encrypting the helper data of one biocryptosystem with the help of a key derived from the other biocryptosystem. Without decryption of the helper data using the same key, we cannot perform authentication. We denote such a biocryptosystem level fusion framework as BIOFUSE.

We further introduce a format-preserving encryption (FPE) scheme [52] as discussed in Chapter 2 for combining the two biocryptosystems. FPE preserves the format of the plaintext and the ciphertext. We discuss the rationale behind the use of the FPE scheme in Section 3.4. Further, the implementation details are provided in Section 3.5.

We perform a thorough security analysis of all the possible combinations in which biometric fusion is performed, taking fuzzy commitment and fuzzy vault schemes into consideration. Table 3.2 summarizes the security of all the 4 possible constructions in both offline and online attack modes. We denote the most secure design among these as S-BIOFUSE (S_3). As shown in Table 3.2, the best attack possible on the proposed framework (particularly, S_3) is the time-memory tradeoff attack in the offline mode with security bound equals to $2^{min(|K_1|,p)+|K_1|} + 2^{min(|K_2|,q)}$ which approximates to $2^{2\times|K_1|}$ or $2^{|K_1|} \times 2^{|K_1|}$, assuming the underlying biocryptosystems are secure.

We provide a detailed analysis of the proposed system's recognition performance and the comparisons with existing multibiometric cryptosystems. The results are shown in Table 3.7, which emphasize the scope of deployment of our proposed architecture in real-time scenarios.

Table 3.2: Summary of security analysis in offline and online mode for all the possible constructions (denoted by S_i) formed by the combination of two biocryptosystems- fuzzy commitment (FC) and fuzzy vault (FV). Security bound is given in terms of the number of trials performed for brute force attack on the system. K_1 and K_2 are the secret keys, and W denotes the error correcting codeword used in fuzzy commitment scheme. H refers to the hash function. Bar $\bar{}$ indicates that a particular biometric template B_1 or B_2 is not known to the attacker. p and q refers to the internal security (measured in bits) of FC and FV respectively. Refer Section 3.3 for the details.

Modes		**Cases S_i, $(1 \leq i \leq 4)$**			
		FC-then-FC (S_1)	**FV-then-FV** (S_2)	**FC-then-FV** (S_3)	**FV-then-FC** (S_4)
Offline	$\bar{B_1}, B_2$	$2^{min(\|K_1\|,p)}$	$2^{min(\|K_1\|,q)}$	$2^{min(\|K_1\|,p)}$	$2^{min(\|K_1\|,q)}$
	$B_1, \bar{B_2}$	$2^{min(\|H(W_2)\|,p)}$	$2^{min(\|K_2\|,q)}$	$2^{min(\|K_2\|,q)}$	$2^{min(\|H(W_2)\|,p)}$
	$\bar{B_1}, \bar{B_2}$[1]	$2^{min(\|K_1\|,p)+\|K_1\|}+$ $2^{min(\|H(W_2)\|,p)}$	$2^{min(\|K_1\|,q)+\|K_1\|}+$ $2^{min(\|K_2\|,q)}$	$2^{min(\|K_1\|,p)+\|K_1\|}+$ $2^{min(\|K_2\|,q)}$	$2^{min(\|K_1\|,q)+\|K_1\|}+$ $2^{min(\|H(W_2)\|,p)}$
Online	$\bar{B_1}, B_2$	$2^{\|H(B_1\|r_1)\|}$	$2^{\|K_1\|}$	$2^{\|H(B_1\|r_1)\|}$	$2^{\|K_1\|}$
	$B_1, \bar{B_2}$	$2^{\|H(W_2)\|}$	$2^{\|K_2\|}$	$2^{\|K_2\|}$	$2^{\|H(W_2\|}$
	$\bar{B_1}, \bar{B_2}$	$2^{\|H(W_2)\|}$	$2^{\|H(K_2)\|}$	$2^{\|H(K_2)\|}$	$2^{\|H(W_2\|}$

3.1 Background and Related Work

Several multibiometric biocryptosystems have been analyzed in [10, 50] for their use in biometric authentication systems. We describe some of the popular multibiometric template protection schemes based on biocryptosystems along with their limitations.

3.1.1 Existing Multi-biometric Fusion Schemes

As stated in ISO/IEC TR 24722 on multimodal and other multibiometric fusion [53], the fusion between different biometric characteristics can be performed at three levels- feature level, score level, and decision level fusion.

In feature-level fusion based biometric system, given the biometric templates extracted from different biometric characteristics, a single template of higher dimensions is generated with more discriminative information than the individual templates. The main challenge is to combine features from other biometric characteristics with different representations. If the features are incompatible or heterogeneous (by distance, similarity), several embedding algorithms are proposed [49] that help construct a combined feature vector. Further, one can use the embedding algorithms in score-level or decision-level fusion approaches. Nandakumar et al. [6] proposed a feature level fusion scheme that derives a single multibiometric feature set from the iris and the

fingerprint templates and secure these features using the fuzzy vault approach. They showed that a vault constructed from a multibiometric feature set has more security and better recognition performance than the vault created by a feature set obtained from a single biometric characteristic. Kanade et al. [54] utilized face and iris templates to generate a single feature set and apply the fuzzy commitment scheme to obtain a high entropy cryptographic key. Nagar et al. [49] provided a practical implementation of a feature-level fusion framework for fuzzy vault and fuzzy commitment schemes that simultaneously protect the multiple templates of a user using a single secure sketch. Feature level fusion using multiple characteristics of a user proves to be significant in providing high privacy as compared to the single characteristics biometric systems since only the fused feature vector is stored on the server database. Further, it requires less storage since only the combined feature vector is stored in the database server. However, it requires additional feature extraction and transformation tools for the heterogeneous features (variable formats based on distance, similarity, etc.).

In the score level fusion, the individual similarity scores obtained from each unimodal systems are normalized and combined to obtain a reliable and accurate score. A brief overview of various ways to achieve score level fusion is given in [55, 56] where a two-level score level fusion approach is proposed to integrate the scores obtained from cancelable templates derived from different biometric characteristics. The authors proposed mean-closure based weighting and rectangular area based weighting technique to obtain the overall fused score. Biocryptosystems can use a similar approach. Score level fusion provides high recognition performance, but it usually provides unpredictable performance as comparison scores of different characteristics may follow a different probability distribution. Another fusion framework is decision level fusion, where the individual accepts/rejects decisions from each unimodal systems are combined to get the final decision. The non-homogeneous features can be used without transformation and can be easily compared with the existing comparators. A decision level fusion is performed on fingerprint-based multi-biometric biocryptosystem [46]. Hash functions are used to protect each fingerprint. The system provides authentication if a certain number of fingerprints out of the total query fingerprints satisfies the threshold criteria. However, using an extra secret key at the second level adds an extra parameter to the overall system, which needs to be secured.

3.1.2 Other Multibiometric Schemes Involving Biocryptosystems

A modular approach is proposed in [51, 57] to construct a multibiometric biocryptosystem where the output of one template protection scheme is fed into another scheme/module that provides the final output. In [57], authors proposed a cancelable secure sketch where the secure sketch is applied on cancelable biometric templates. The secure sketch provides biometric template protection, and the cancelable biometric template prevents correlation attacks and provides renewability property. Cimato et al. [51] proposed a modular approach where two secure sketch schemes are taken as independent modules to secure multibiometric templates. Given two biometric templates, a secure value is constructed from the first template. The key is generated from the first biometric template as input. The key is then XORed with the second biometric template to generate helper data, which is stored along with secure value on the server. The above approach is generalized by Fang et al. [58], where multiple biometric templates are combined in a cascaded manner while deploying the secure sketch framework [1] within the fuzzy commitment scheme. The advantage of the modular approach is that it easily allows the addition of biometric characteristics along with the heterogeneous templates. The limitation is that the modular-level fusion scheme's overall security is defined by the security of secure sketch in the outermost module. A multi-biometrics cancelable approach has been proposed in [59] where a fuzzy extractor is combined with a bit-wise encryption scheme to generate a cancelable template as output. The scheme provides high security based on the assumption that getting access to one biometric template by an attacker is equivalent to getting both biometric templates of that user. The scheme provides high-performance rates for any binarized data. However, it would require pre-transformations for non-binarized input data such as fingerprint template.

3.2 Models and Settings

We present the system's entities and possible attack scenarios.

3.2.1 System Model and Participants

The proposed system consists of users and a server. Two biocryptosystems are involved in the biometric system. The user can be an attacker who tries to get successful authentication by combining the credentials of a genuine user of the system. The user provides two biometric characteristics, iris/face and a fingerprint, one for each biocryptosystem. The helper data generated as outputs from the two biocryptosystems is stored corresponding to each user ID on the database present on the server.

3.2.2 Attack Model

Our proposed scheme is based on the client-server setup. The end-user provides biometric data at the client-side. The storage of the data used for authentication is done on the server. Hence, the server is also responsible for user authentication.

We define the security of all the possible constructions in terms of attack complexity in online and offline modes. In the **online mode**, the attacker tries to authenticate itself in the real-time scenario by impersonating a genuine user's biometric templates. On providing inputs as one or both biometric templates B_1 and B_2, the attacker gets the final output as 0 that indicates an authentication failure, or 1 that means successful authentication. In the **offline mode**, an attacker's goal is to extract some secret information such as biometric templates, secret key, etc. from the simulated system. The attacker provides one or more known inputs- such as key or biometric data and tries to get the desired information. The extracted information is used in the online attack mode to perform successful authentication.

3.2.3 Security Notion

In the fuzzy commitment scheme shown in Figure 2.2, authentication is said to be successful when the hash of error correcting codeword $H(W')$ generated during the authentication is the same as the hash of codeword $H(W)$ stored during enrolment. For the rest of the Chapter, we consider that the random message used to generate codeword is denoted by Δ of length

δ. Further, we use r to denote the random message which helps in generating the secret key from biometric template. If the size δ of the random message used to generate the codeword is small, i.e., $|\delta| < 128$ bits, it would be easy to guess the codeword. Once the codeword is leaked, the original biometric template could be easily revealed, which breaks the fuzzy commitment scheme's security. We denote it as the fuzzy commitment scheme's internal security, and it is given by p bits.

Similarly, in the fuzzy vault scheme shown in Figure 2.3, the secret parameter is a system-generated key K that binds with the input biometric template B to generate a public vault V. Given a vault V constructed with a polynomial of degree n, the attacker can perform brute force to the vault by applying a polynomial decoding algorithm on every combination of $(n+1)$ vault points to find out the one which gives the correct secret key K. It is calculated [47, 39] as number trials (in bits) to filter $(n+1)$ genuine points out of all the v points in the vault. It is given as

$$= \log_2(\binom{v}{n+1} / \binom{G}{n+1})$$

where G is the number of genuine points in the vault. The knowledge of $(n+1)$ genuine points could reveal the secret key K to the attacker. Generally, the number of trials used to get the secret key is less [39], which makes fuzzy vault insecure. We denote it as the fuzzy vault scheme's internal security, and it is given by q bits. We consider the fuzzy commitment scheme's internal security as p bits and fuzzy vault scheme as q bits.

We made a few assumptions to analyze the security of BIOFUSE, which is described in Section 3.3.

The assumptions are:

- It is not possible for the attackers to get biometric characteristics using brute force attack with a probability of $1/2^{|B|}$, since the size of the input biometric template B, is generally large [39, 60].
- The size of error correcting codeword is sufficiently larger than the key generated using a fuzzy commitment scheme, i.e., $|W| >> |K|$. Further, error correcting codewords need to

Table 3.3: Some other notations.

Notation	Description
I	Biometric characteristics given by user as a digital image
B	Biometric features representing the biometric template, where B' denotes another instance of B
W	Error correcting codeword
H	A cryptographic hash function
E	A symmetric key encryption scheme using block cipher modes of operation
V	A fuzzy vault with genuine and chaff points combined
K	A cryptographic key
FC	fuzzy commitment scheme
FV	fuzzy vault scheme
FFE	a format-preserving encryption module
FFD	a format-preserving decryption module
$\mathtt{P}_{\mathtt{en}}$	an instantiate of FC enrolment module
$\mathtt{P}_{\mathtt{au}}$	an instantiate of FC authentication module
$\mathtt{Q}_{\mathtt{en}}$	an instantiate of FV enrolment module
$\mathtt{Q}_{\mathtt{au}}$	an instantiate of FV authentication module
HD	helper data

be of the same size as the size of biometric templates.

- The length of both the keys K_1 and K_2 is ≥ 128 bits to ensure the overall scheme's security. Note that, K_1 and K_2 are the instances of the two keys which could belong to any of the fuzzy commitment or fuzzy vault scheme.

- The helper data denoted as HD, whether encrypted or unencrypted is stored on the database server as public data without any password protection.

3.3 Proposed Work

Table 3.3 shows some necessary notations that are used in this work.

Given the inputs as multibiometric characteristics, we provide all the possible combinations that can be constructed from combining two biocryptosystems- fuzzy commitment scheme and fuzzy vault scheme to construct a biocryptosystem-level fusion framework. Similar to a general biometric authentication system, BIOFUSE consists of two phases:

- **Enrolment phase** takes multibiometric characteristics as input for biocryptosystems and

generates secure, public values, i.e., helper data, stored on the server.

- **Authentication phase** authenticates the user only if both the biometric characteristics provided during authentication are from the genuine user.

The combinations or cases that are possible with two biocryptosystems- fuzzy commitment (FC) and fuzzy vault (FV) schemes can be categorized into 4 constructions:

1. Two fuzzy commitment schemes can be combined, denoted as FC-then-FC,
2. Two fuzzy vault schemes can be combined, denoted as FV-then-FV,
3. A fuzzy commitment scheme can be combined with FV scheme denoted as FC-then-FV
4. A fuzzy vault scheme can be combined with FC scheme denoted as FV-then-FC.

3.3.0.0.1 Security analysis For each of the 4 constructions mentioned, we also evaluate the security of the constructed scheme. We analyze two modes -

- **Online mode:** The attacker interacts with the deployed biometric system. It provides biometric inputs as B_1 and/or B_2 and gets output as $1/0$ that denotes match or no match, respectively.
- **Offline mode:** In the offline mode, instead of interacting with the deployed biometric system, the attacker implement the underlying algorithms and functions of the deployed system in its simulated system, with several parameters. The attacker provides some known values such as key or biometric data to retrieve unknown secret data from the system. The secrets could be used in the online attack mode.

We further consider three threat models:

1. Attacker knows B_2 but B_1 is not known.
2. Attacker knows B_1 but B_2 is not known.
3. Both B_1 and B_2 are unknown to the attacker.

Thus, we do our analysis for all the 4 constructions of BIOFUSE under 2 modes of attack, each with 3 threat models.

In the next subsections, we describe each of the 4 constructions in detail. Each construction is denoted as $\mathcal{S}_i$, where $1 \leq i \leq 4$. For each construction, we first describe its working, followed by its security analysis.

3.3.1 FC-then-FC: $\mathcal{S}_1$

3.3.1.0.1 **Construction** FC-then-FC is constructed by combining two fuzzy commitment schemes, each taking a different instance of the biometric template as input. Fig. 3.2 shows the enrolment and authentication phases.

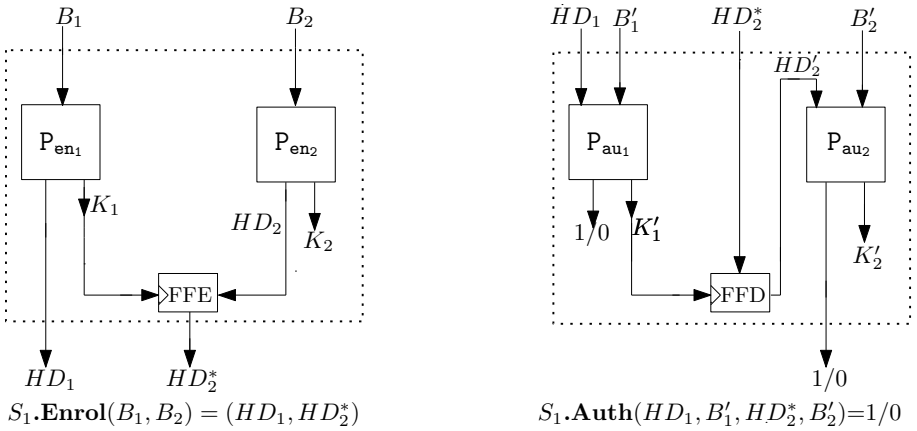


Figure 3.2: Enrolment and authentication phase for the case: FC-then-FC. $\mathtt{P}_{\mathtt{en}_1}$ and $\mathtt{P}_{\mathtt{en}_2}$ represent the two instantiates of enrolment modules for fuzzy commitment scheme. $\mathtt{P}_{\mathtt{au}_1}$ and $\mathtt{P}_{\mathtt{au}_2}$ represent the two instantiates of authentication modules for fuzzy commitment scheme. FFE and FFD denotes the format-preserving encryption and decryption respectively. Refer Figure 2.2 and Figure 2.3 for internal workings of these modules. For other notations, refer Table 3.3.

Enrolment Phase: During the enrolment phase, $\mathtt{P}_{\mathtt{en}_1}$ generates a cryptographic key K_1 from the input biometric template B_1. It also generates a public, helper data denoted as HD_1. From the second input biometric data B_2, another helper data HD_2 is generated with the help of $\mathtt{P}_{\mathtt{en}_2}$ module. The key K_1 encrypts the helper data HD_2 to give an encrypted helper data HD_2^* using format-preserving encryption scheme FFE. The FFE helps in preserving the format of ciphertext and plaintext. The role of FFE is discussed later in Section 3.4. For similar reasons, we use FFE scheme in all our other three constructions as well. The key K_2 does not play any

role in the enrolment phase. The HD_1 and HD_2^* are stored on the server as public values.

Authentication phase: During the authentication phase, $\mathtt{P}_{\mathtt{au}_1}$ helps to recover the key K_1'. If the biometric template B_1' given during authentication is similar to B_1, i.e. $d(B_1, B_1') < t$, where t is a pre-defined threshold and d is the hamming distance between two binary strings, then K_1' is a correct key (i.e. equal to K_1). The FFD takes the key and encrypted helper data as inputs and decrypts the helper data denoted as HD_2'. With the help of second biometric template B_2' and helper data HD_2', hash verification is performed inside the $\mathtt{P}_{\mathtt{au}_2}$ module as

$$H(W_2') = (H(W_2))' \tag{3.1}$$

where $H(W_2')$ represents the hash of codeword W_2' generated internally from P_{au_2} (second fuzzy commitment scheme) in the authentication phase. $(H(W_2))'$ represents the hash of the codeword W_2, which is stored as a part of helper data HD_2 during the enrolment in the encrypted form. If the condition is satisfied, the authentication is successful with output 1 and is shown as

$$\mathcal{S}_1.Auth(HD_1, B_1', HD_2^*, B_2') \longrightarrow 1 \tag{3.2}$$

K_2' generated as output does not play any authentication role.

3.3.1.0.2 Security Analysis We discuss the security analysis of $\mathcal{S}_1$ in 2 different modes: online and offline mode as follows:

CASE A.(Offline mode setup). The attacker could know some of the secret parameters including the key(s), B_1 or B_2 as inputs. Given the public parameters as

$HD_1 = (H(W_1), r_1, S_1)$ and

an encrypted helper data, HD_2^* such that

$FFE(K_1, HD_2) \longrightarrow HD_2^*$ where $HD_2 = (H(W_2), r_2, S_2)$

the attacker can perform several functions to generate the corresponding outputs as,

$\mathcal{S}_1.Enrol(B_1, B_2) \longrightarrow (HD_1, HD_2^*)$ where

$\mathtt{P}_{\mathtt{en}_1}(B_1) \longrightarrow (HD_1, K_1)$,

$\mathtt{P}_{\mathtt{en}_2}(B_2) \longrightarrow (HD_2, K_2)$,

$\mathtt{P}_{\mathtt{au}_1}(HD_1, B_1') \longrightarrow (1/0, K_1')$,

$FFD(K_1', HD_2^*) \longrightarrow HD_2'$,

$\mathtt{P}_{\mathtt{au}_2}(HD_2', B_2') \longrightarrow (1/0, K_2')$

The random numbers r_1, r_2 are the part of helper data and are used for generating or recovering the key during the authentication. The attacker can individually run any or all the above-mentioned functions or algorithms such as FFE, $\mathtt{P}_{\mathtt{en}_1}$, $\mathtt{P}_{\mathtt{en}_2}$, $\mathtt{P}_{\mathtt{au}_1}$, $\mathtt{P}_{\mathtt{au}_2}$ and FFD to get the respective outputs.

Under the offline mode setup, we analyse our first construction under three threat models as follows:

1. **Construction:** $\mathcal{S}_1$

   Mode: Offline

   Threat Model: As attacker knows B_2' where $d(B_2, B_2') < t$, the aim is to recover B_1' or K_1' such that the authentication is successful.
   Since, B_1' is unknown, correct key $K_1' = K_1$ is unknown. The attacker can try to guess K_1' and perform the function FFD to get HD_2' given as,

$$FFD(K_1', HD_2^*) \longrightarrow HD_2' \tag{3.3}$$

   $HD_2' = ((H(W_2))', r_2', S_2')$. Further, the biometric template B_2' is known to the attacker, therefore, using the correct B_2' and all the values of S_2' derived from the above equation, the attacker can perform $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2'. From W_2', the attacker can then compute $H(W_2')$. Given $(H(W_2))'$, it can check the condition given in (3.1). If the condition is satisfied, the authentication is successful for the corresponding guessed key K_1', else a new value of K_1' is guessed and

attack procedure is repeated.

The success probability such that two hash values are equal is $1/|H(W_2)|$ which takes on an average $2^{|H(W_2)|}$ trials. However, the total number of possible values of keys K_1 used to decrypt the helper data to obtain a valid sample space of hash value $(H(W_2))'$ is lower, given as $|K_1| \leq |H(W_2)|$. Therefore, the attacker would choose to guess the values of the key K_1 directly. Hence, the number of trials required to guess a correct key K_1', considering the internal security of fuzzy commitment scheme as p bits= $2^{min(|K_1|,p)}$, where $|.|$ denotes the size of the parameter.

Given correct key K_1' and the known random value r_1 obtained from the public helper data HD_1, the attacker can guess the correct biometric template B_1' with high probability[2], using pre-image attack such that $H(B_1'\|r_1)$ = K_1'. Therefore, the number of trials required to guess B_1', given a correct key K_1' is equal to $2^{|K_1|}$.

2. **Construction:** $\mathcal{S}_1$

   Mode: Offline

   Threat Model: As attacker knows B_1' where $d(B_1, B_1') < t$, the aim is to recover B_2' or K_2' such that the authentication is successful.

   The attacker can generate the key K_1' with the help of known B_1' and helper data HD_1 using the underlying algorithm of P_{au_1} module as

   $\mathtt{P}_{\mathtt{au_1}}(HD_1', B_1') \longrightarrow (1, K_1')$.

   Since, $d(B_1, B_1') < t$, output is 1. The generated K_1' is a correct key and is equal to K_1 which was generated during the enrolment phase. Attacker can then decrypt the given HD_2^* with the help of key K_1' using (3.3) to get $HD_2' = ((H(W_2))', r_2', S_2')$.

   Note that K_2' does not play any role in enrolment or authentication phase. The attacker can try to guess B_2' directly such that $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2'. From W_2', the attacker can compute $H(W_2')$ to checks the condition given in (3.1). Therefore, the number of trials needed to guess a correct B_2' with high probability[2]is equal number of trials needed to guess a correct hash of codeword

[2]considering false match rate (FMR), false non-match rate (FNMR) as negligible

$H(W_2'')$ that will match with the given hash $(H(W_2))'$. It is equal to $2^{min(|H(W_2)|,p)}$ trials, considering the internal security of second fuzzy commitment scheme as p bits.

3. **Construction:** $\mathcal{S}_1$

   Mode: Offline

   Threat Model: As attacker does not know B_1' and B_2', the aim is to recover any or all of the B_1', K_1', K_2' and B_2' such that authentication is successful.

   We use format-preserving encryption[3] in our proposed scheme to encrypt the helper data HD_2. Since B_1' is unknown, the attacker can guess key K_1' and perform the function FFD as shown in (3.3) to get HD_2' from HD_2^*. However, since a format-preserving encryption scheme is used, the format of HD_2' remains the same and is equal to the format of HD_2^*, irrespective of the secret key applied for decryption. Therefore, the attacker cannot guess whether the key K_1' is correct or not by observing the format of HD_2'. The attacker would perform an exhaustive search on all the possible values of key K_1'. For each possible K_1', it will decrypt the helper data HD_2^* to get the decrypted helper data $HD_2' = ((H(W_2))', r_2', S_2')$. Thus the attacker would store all the possible values of $(H(W_2))'$ corresponding to each guessed K_1' in the form of a table.

   The attacker can then guess the value of $H(W_2')$ and check if $H(W_2')$ matches with one of the hashes $(H(W_2))'$ stored in the table to satisfy (3.1). Note that attacker would prefer to guess the hash of codeword directly rather than the codeword since $|W_2| >> |H(W_2)|$. If the guessed $H(W_2')$ does not match any of the table entries; the attacker can guess a new hash value.

   Therefore, the best attack possible when both B_1' and B_2' are unknown is the time-memory trade-off attack given by security bound denoted Time $\times$ Memory. Time includes the number of trials taken for guessing all the possible values of the key given as K_1' considering the internal security of fuzzy commitment scheme as p bits = $2^{min(|K_1|,p)}$. The memory is used to store all the possible decrypted values $(H(W_2))'$ as a part of helper data. It is given as $2^{|K_1|}$. Thus the security would be given as time-memory trade-off attack bound plus the number of trials of $H(W_2)$ that are needed to match the two hash values, considering

[3]The rationale behind the use of format-preserving encryption scheme is given in Section 3.4

internal security of fuzzy commitment scheme as p bits. It is given as $2^{min(|K_1|,p)+|K_1|} + 2^{min(|H(W_2)|,p)}$.

Given a valid K_1' (which validates the condition for successful authentication) and r_1, with high probability[3], the attacker can guess the correct biometric template B_1' using pre-image attack such that

$H(B_1'\|r_1) = K_1'$ in $2^{|K_1|}$ trials.

The attacker can recover B_2' with high probability[3] by guessing B_2' such that $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2' with the condition given in (3.1).

The number of trials needed to guess a correct B_2' with high probability[3] is equal to guessing a correct hash of codeword $H(W_2')$ that will match with the given hash $(H(W_2))'$. It is equal to $2^{|H(W_2)|}$ trials.

CASE B.(Online mode setup). The attacker gives inputs in the form of biometric templates B_1 and B_2 to get the final output as 1 or 0 which indicates whether the authentication is successful or not. Given the public parameters as

$HD_1 = (H(W_1), r_1, S_1)$ and

an encrypted helper data, HD_2^* such that

$FFE(K_1, HD_2) \longrightarrow HD_2^*$ where $HD_2 = (H(W_2), r_2, S_2)$,

the attacker can run only the following 2 functions to generate the corresponding outputs as,

$\mathcal{S}_1.Enrol(B_1, B_2) \longrightarrow (HD_1, HD_2^*)$,

$\mathcal{S}_1.Auth(HD_1, B_1', HD_2^*, B_2') \longrightarrow 1/0$

Under the online mode setup, we analyze our first construction under three threat models as follows:

1. **Construction:** $\mathcal{S}_1$

   Mode: Online

Threat Model: As attacker knows B_2' where $d(B_2, B_2') < t$, the aim is to get successful authentication, i.e. it satisfies (3.2).

Since, B_1' is unknown, K_1' is also unknown. The attacker can input the random biometric templates B_1'. The system would generate corresponding K_1' as an application of the fuzzy commitment scheme given as $H(B_1'\|r_1) \longrightarrow K_1'$ (details of key generation are provided in Section 3.4 in (3.15). In general, the key K_1' is derived after truncation from the hash output $H(B_1\|r_1)$ or it can be of same size as of hash output. Using K_1', the system would decrypt the helper data HD_2^* internally to get decrypted helper data HD_2' using (3.3), where $HD_2' = ((H(W_2))', r_2', S_2')$.

B_2' is known to the attacker; therefore, using the correct B_2' and the derived values of S_2', the system would perform error correcting code decoding by performing $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2'. The system would then verifies the condition given in (3.1). The process is repeated until the authentication is successful.

Since the key K_1' is used to decrypt the helper data, the attacker would need to perform $H(B_1'\|r_1)$ number of trials of B_1' to get all the possible values of key K_1'. Therefore, the number of trials of B_1' required to get successful authentication = $2^{|H(B_1\|r_1)|}$ when key K_1' is derived after truncation from the hash output $H(B_1'\|r_1)$ and is equal to $2^{|K_1|}$ if key $|K_1'|$ = $|H(B_1'\|r_1)|$.

2. **Construction:** $\mathcal{S}_1$

   Mode: Online

   Threat Model: As attacker knows B_1' where $d(B_1, B_1') < t$, the aim is to get successful authentication, i.e. it satisfies (3.2).

   Attacker provides B_1' as input. Using it, the system would generate the key K_1' as $\mathtt{P}_{\mathtt{au}_1}(HD_1', B_1') \longrightarrow (1, K_1')$. Since, $d(B_1, B_1') < t$, K_1' is a correct key with high probability[3] and is equal to K_1 generated during the enrolment phase. The key is used by system to decrypt HD_2^* to get the correct helper data as shown in (3.3). HD_2' is a correctly decrypted helper data and is given as $HD_2' = ((H(W_2))', r_2', S_2')$.

The attacker can guess guess B_2' such that on providing B_2' to the module P_{au_2}, the system would compute $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2'. If the condition given in (3.1) is satisfied, the system will show successful authentication. Thus, the attacker would know if the given B_2' is correct or not.

Therefore, the number of trials of B_2' required for successful authentication is equal to guessing a correct hash of codeword $H(W_2')$ that matches with the given hash $(H(W_2))'$. It sums to $2^{|H(W_2)|}$ trials. Note that K_2' does not play any role in the enrolment or authentication phase.

3. **Construction:** $\mathcal{S}_1$

   Mode: Online

   Threat Model: As the attacker does not know B_1' and B_2', the aim is to get successful authentication, i.e. it satisfies (3.2).

   In the online attack mode, the attacker has no memory. The attacker needs to guess all possible combinations of B_1' and B_2' such that (3.1) is satisfied, which gives output 1 to denote a successful authentication. Therefore, the number of trials that need to be performed to obtain the correct matching of two hash values is equivalent to the pre-image attack complexity, assuming H as a random oracle. Thus the number of trials is equal to $2^{|H(W_2)|}$.

3.3.1.1 FV-then-FV: $\mathcal{S}_2$

3.3.1.1.1 Construction FV-then-FV is constructed by combining two fuzzy vault schemes; each takes a different instance of the biometric template as input. Fig. 3.3 shows the enrolment and authentication phases.

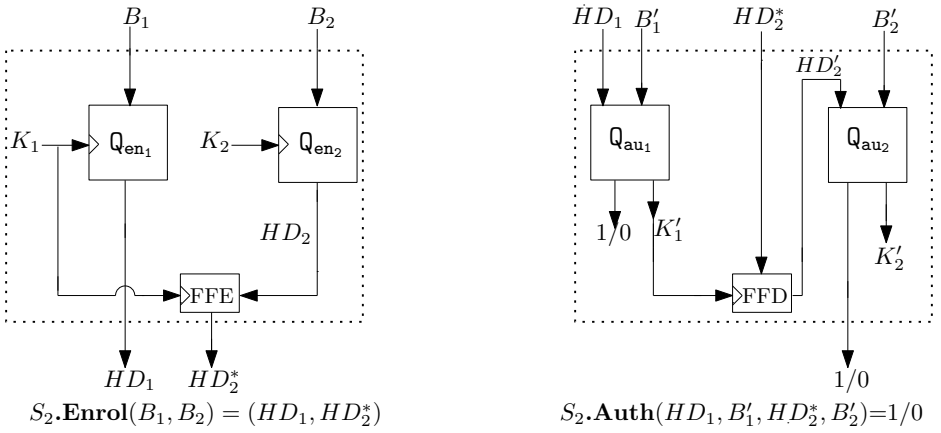


Figure 3.3: Enrolment and authentication phase for the case: FV-then-FV. $\mathtt{Q}_{\mathtt{en}_1}$ and $\mathtt{Q}_{\mathtt{en}_2}$ represent the two instantiates of enrolment modules for fuzzy vault scheme respectively. Similarly, $\mathtt{Q}_{\mathtt{au}_1}$ and $\mathtt{Q}_{\mathtt{au}_2}$ represent the two instantiates of authentication modules for fuzzy vault scheme respectively. FFE and FFD denotes the format-preserving encryption and decryption respectively.

Enrolment Phase: During the enrolment phase, $\mathtt{Q}_{\mathtt{en}_1}$ generates a helper data HD_1 using a random secret key K_1 and the input biometric template B_1. From the second input biometric data B_2 and another secret key K_2, helper data HD_2 is generated with the help of $\mathtt{Q}_{\mathtt{en}_2}$ module. Both the keys are internally generated by the system. The key K_1 encrypts the helper data HD_2 to generate a transformed helper data HD_2^* using format-preserving encryption scheme FFE.

Authentication phase: During the authentication phase, $\mathtt{Q}_{\mathtt{au}_1}$ helps to recover the key K_1'. If the biometric template B_1' given during authentication is similar to B_1, i.e. $|B_1 - B_1'| < \epsilon$, which denotes the symmetric difference between two biometric templates, then K_1' is a correct key (i.e. equal to K_1). The FFD takes the key and encrypted helper data as input and decrypts the helper data denoted as HD_2'. The second biometric template B_2' and helper data HD_2' are given as inputs to $\mathtt{Q}_{\mathtt{au}_2}$. If the symmetric difference between two biometric templates denoted as $|B_2 - B_2'| < \epsilon$, the key K_2' is recovered such that

$$H(K_2') = (H(K_2))' \tag{3.4}$$

where, $H(K_2')$ represents the hash of key K_2' generated from Q_{au_2} (fuzzy vault scheme) in the authentication phase. $(H(K_2))'$ represents the hash of the key K_2, which is stored as a part of helper data HD_2 during the enrolment in the encrypted form. If the above condition is satisfied,

1 is given as the output to show a successful authentication as

$$\mathcal{S}_2.Auth(HD_1, B_1', HD_2^*, B_2') \longrightarrow 1 \tag{3.5}$$

3.3.1.1.2 Security Analysis We discuss the security analysis of $\mathcal{S}_2$ in 2 different modes: online and offline mode as follows:

CASE A.(Offline mode setup). In the mentioned setup, the attacker can know any of the secret parameters including the key(s), B_1 or B_2 as inputs. Given the public parameters as

$HD_1 = (V_1, H(K_1))$, and

an encrypted helper data, HD_2^* such that

$FFE(K_1, HD_2) \longrightarrow HD_2^*$ where $HD_2 = (V_2, H(K_2))$

the attacker can perform several functions to generate the corresponding outputs as,

$\mathcal{S}_2.Enrol(B_1, B_2) \longrightarrow (HD_1, HD_2^*)$ where

$\mathtt{Q}_{\mathtt{en}_1}(B_1, K_1) \longrightarrow HD_1$,

$\mathtt{Q}_{\mathtt{en}_2}(B_2, K_2) \longrightarrow HD_2$,

$\mathtt{Q}_{\mathtt{au}_1}(HD_1, B_1') \longrightarrow (1/0, K_1')$,

$FFD(K_1', HD_2^*) \longrightarrow HD_2'$,

$\mathtt{Q}_{\mathtt{au}_2}(HD_2', B_2') \longrightarrow (1/0, K_2')$

The attacker can individually run any or all the above-mentioned functions or algorithms such as FFE, $\mathtt{Q}_{\mathtt{en}_1}$, $\mathtt{Q}_{\mathtt{en}_2}$, $\mathtt{Q}_{\mathtt{au}_1}$, $\mathtt{Q}_{\mathtt{au}_2}$ and FFD to get the respective outputs.

Under the mentioned setup, we analyse our first construction under three threat models as follows:

1. **Construction:** $\mathcal{S}_2$

   Mode: Offline

Threat Model: As attacker knows B_2' where $|B_2 - B_2'| < \epsilon$, the aim is to recover B_1' or K_1' such that the authentication is successful.

Since, B_1' is unknown, correct key $K_1' = K_1$ is unknown. The attacker can guess a random value of K_1' and can decrypt the encrypted helper data HD_2^* as

$$FFD(K_1', HD_2^*) \longrightarrow HD_2' \tag{3.6}$$

where $HD_2' = V_2' || (H(K_2))'$. Since B_2' is known to the attacker, therefore, using the correct B_2' and the derived value of vault V_2' corresponding to the guessed key K_1', the attacker can derive the key K_2' using polynomial interpolation. Computing $H(K_2')$, the attacker can check the condition as given in (3.4). If the condition is satisfied, the authentication is successful for the corresponding guessed key K_1', else a new value of K_1' is guessed the attack procedure is repeated.

Since the success probability such that two hash values are equal is computed as $1/|H(K_2)|$, on average, the total number of trials required is $2^{|H(K_2)|}$. However, the total number of possible values of keys K_1 used to decrypt the helper data to obtain the valid sample space of hash value $(H(K_2))'$ is smaller, given as $|K_1| \leq |H(K_2)|$. Therefore, the attacker would choose to guess the values of the key K_1 directly. Hence, the number of trials required to guess a correct key K_1', considering the internal security of fuzzy vault scheme as q bits= $2^{min(|K_1|,q)}$, where $|.|$ denotes the size of the parameter.

Given correct key K_1' and V_1', the attacker can recover the correct biometric template B_1' with high probability[3], by separating out the genuine and chaff points.

2. **Construction:** $\mathcal{S}_2$

   Mode: Offline

   Threat Model: As attacker knows B_1' where $|B_1 - B_1'| < \epsilon$, the aim is to recover B_2' or K_2' such that the authentication is successful.

   The attacker can generate the key K_1' with the help of known B_1' and public helper data HD_1 as

   $\mathtt{Q}_{\mathtt{au}_1}(HD_1, B_1') \longrightarrow (1, K_1')$.

Since, $|B_1 - B_1'| < \epsilon$, K_1' is a correct key and is equal to K_1 which is generated during the enrolment phase. Attacker can decrypt the encrypted helper data HD_2^* with the help of the correct key using (3.6) to get a correctly decrypted helper data $HD_2' = (V_2'||(H(K_2))')$. Given $(H(K_2))'$, the attacker can guess all the possible values of key K_2'. It can check if any of the guessed key satisfies (3.4).

Therefore, the number of trials required to guess the correct key K_2', while considering the security of fuzzy vault scheme as q bits = $2^{min(|K_2|,q)}$. Note that the pre-image attack complexity will not hold here when $|K_2| \leq |H(K_2)|$, where H is the given hash function that acts as a random oracle.

With the help of correct key K_2' and decrypted value of vault V_2', B_2' can be recovered with high probability[3] by separating the genuine and chaff points.

3. **Construction:** $\mathcal{S}_2$

   Mode: Offline

   Threat Model: As attacker does not know B_1' and B_2', the aim is to recover any or all of the B_1', K_1', K_2' and B_2' such that authentication is successful.

   We use format-preserving encryption in our proposed scheme to encrypt the helper data HD_2. Since B_1' is unknown, the attacker can guess key K_1' and perform the function FFD as shown in (3.6) to get HD_2' from HD_2^*. However, since a format-preserving encryption scheme is used, the format of HD_2' remains the same and is equal to the format of HD_2^*, irrespective of the secret key applied for decryption. Therefore, the attacker cannot guess whether the key K_1' is correct or not by observing the format of HD_2'. The attacker can perform an exhaustive search on all the possible values of key K_1'. For each possible K_1', it can decrypt the helper data HD_2^* to get the decrypted helper data $HD_2' = V_2'||(H(K_2))'$.

   The attacker can store all the possible values of $(H(K_2))'$ corresponding to each guessed K_1' in the form of a table. It can guess the value of K_2' and can compute $H(K_2')$ to check if $H(K_2')$ matches with one of the hashes stored in the table by satisfying (3.4). If the guessed K_2' for which $H(K_2')$ does not match with any of the table entries, the attacker could guess a new key K_2'.

Therefore, the best attack possible when both B_1' and B_2' are unknown is a time-memory tradeoff attack given by security bound denoted as Time $\times$ Memory. Time includes the number of trials taken for guessing all the possible values of the key given as K_1' considering the internal security of fuzzy vault scheme as q bits = $2^{min(|K_1|,q)}$. The memory is used to store all the possible decrypted values $(H(K_2))'$ as a part of helper data. It is given as $2^{|K_1|}$. Thus the security would be given as time-memory tradeoff attack bound plus the number of trials of K_2 that are needed to match the two hash values, considering internal security of fuzzy vault scheme as q bits. It is given as $2^{min(|K_1|,q)+|K_1|} + 2^{min(|K_2|,q)}$.

With the help of a valid key K_1' and the given vault V_1 (as a part of helper data HD_1), B_1' can be recovered by separating the genuine and chaff points with high probability[3].

With the help of a valid key K_2' and the vault V_2' decrypted by a correct key K_1', B_2' can be recovered by separating the genuine and chaff points with high probability[3].

CASE B.(Online mode setup). In the mentioned setup, the attacker gives inputs in the form of biometric templates B_1 and B_2 to get the final output as 1 or 0 which indicates whether the authentication is successful or not. Given the public parameters as

$HD_1 = V_1 \| H(K_1)$ and

an encrypted helper data, HD_2^* such that

$FFE(K_1, HD_2) \longrightarrow HD_2^*$ where

$HD_2 = V_2 \| H(K_2)$

the attacker can run only the following 2 functions to generate the corresponding outputs as,

$\mathcal{S}_2.Enrol(B_1, B_2) \longrightarrow (HD_1, HD_2^*)$ and

$\mathcal{S}_2.Auth(HD_1, B_1', HD_2^*, B_2') \longrightarrow 1/0$

Under the mentioned setup, we analyze our first construction under three threat models as follows:

1. **Construction:** $\mathcal{S}_2$

Mode: Online

Threat Model: As attacker knows B_2' where $|B_2 - B_2'| < \epsilon$, the aim is to get successful authentication, i.e. it satisfies (3.5).

Since, B_1' is unknown, K_1' is also unknown. The attacker can provide B_1' as input. The system can derive K_1' by applying polynomial interpolation on the B_1' and V_1 obtained as a part of public helper data HD_1. The key is derived as

$\mathtt{Q}_{\mathtt{au}_1}(HD_1, B_1') \longrightarrow (1/0, K_1')$.

Using the derived K_1', the system can decrypt the encrypted helper data internally using (3.6) to get to get the decrypted helper data $HD_2' = V_2' \| (H(K_2))'$. It implies that for different values of guessed B_1', corresponding values of vault V_2' would be derived. B_2' is known to the attacker, therefore, using the correct B_2' and all the possible values of V_2', the system can derive the corresponding key K_2'. It can compute $H(K_2')$ and verify it with $(H(K_2))'$ obtained as a part of helper data. If (3.4) is satisfied, K_2' is a correct key, else the process is repeated with a new value of B_1'.

Note that the attacker can only give B_1' as input to the system and not the key K_1'. Therefore, the number of trials of B_1' needed to get correct key K_1' which further leads to successful authentication is equal to $2^{|K_1|}$, where $|.|$ denotes the size of the parameter. We consider that within $2^{|K_1|}$ trials of B_1', the system would be able to get a correct key K_1' that will decrypt the helper data correctly with high probability[3].

2. **Construction:** $\mathcal{S}_2$

Mode: Online

Threat Model: As attacker knows B_1' where $|B_1 - B_1'| < \epsilon$, the aim is to get successful authentication, i.e. it satisfies (3.5).

Attacker can provide B_1' as input. The system can the generate the key K_1' using the helper data HD_1 and B_1'. Since, $|B_1 - B_1'| < \epsilon$), K_1' is a correct key, equal to K_1. The key can be used to decrypt HD_2^* to get the correct helper data through (3.6) as $HD_2' = (V_2' \| (H(K_2))')$

Attacker can guess the biometric template B_2'. With the help of B_2' and correct value of vault V_2', the key K_2' is generated by system using polynomial interpolation. If B_2' is correct,

with high probability[3], K_2' is correctly generated by the system, satisfying (3.4).

Since $(H(K_2))'$ is always correct, therefore, the number of trials of B_2' required for successful authentication is equal to the sample space of $|K_2| = 2^{|K_2|}$. Note that the pre-image attack complexity will not hold here when $|K_2| \leq |H(K_2)|$, where H is the given hash function that acts as a random oracle. We consider that within $2^{|K_2|}$ trials of B_2', the system would be able to get a correct key K_2' to authenticate the attacker successfully with high probability[3].

3. **Construction:** $\mathcal{S}_2$

   Mode: Online

   Threat Model: As the attacker does not know B_1' and B_2', the aim is to get successful authentication, i.e. it satisfies (3.5).

   In the online attack mode, the attacker has no memory. The attacker would guess all possible combinations of B_1' and B_2' such that (3.4) is satisfied, which gives output 1 to denote a successful authentication. Therefore, the number of trials of B_1' and B_2' that needs to be performed to obtain correct matching of two hash values are equivalent to the pre-image attack complexity, assuming H as a random oracle. Thus the number of trials is equal to $2^{|H(K_2)|}$. Note that the attacker cannot obtain or guess the key K_2 during an online attack and gets only $1/0$ as the output. Therefore, the scope of performing a collision attack on the system by finding two keys, K_2 and K_2' such that their hash matches with each other is negligible.

3.3.1.2 FC-then-FV: $\mathcal{S}_3$

3.3.1.2.1 Construction FC-then-FV is constructed by combining a fuzzy commitment scheme (FC) with a fuzzy vault scheme (FV). Fig. 3.4 shows the enrolment and authentication phases.

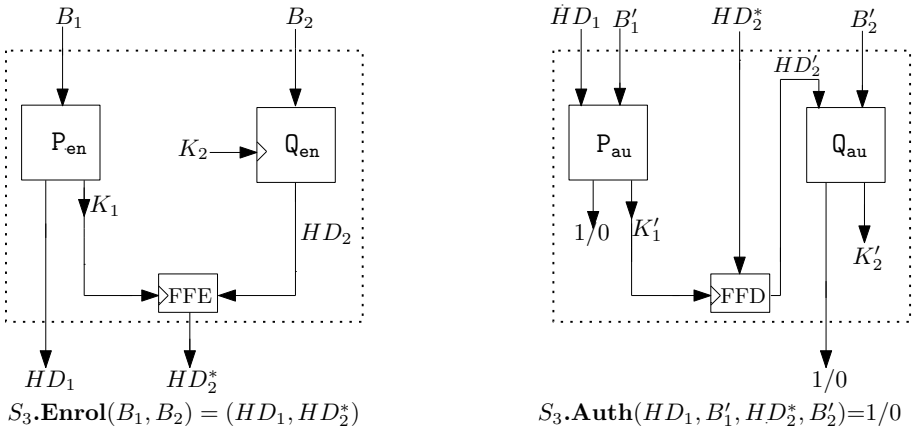


Figure 3.4: Enrolment and authentication phase for the case: FC-and-FV. $\mathtt{P}_{\mathtt{en}}$ and $\mathtt{Q}_{\mathtt{en}}$ represent the enrolment modules for fuzzy commitment and fuzzy vault scheme respectively. Similarly, $\mathtt{P}_{\mathtt{au}}$ and $\mathtt{Q}_{\mathtt{au}}$ represent the authentication modules for fuzzy commitment and fuzzy vault scheme respectively. Refer Fig. 2.2 and 2.3 for details. FFE and FFD denotes the format-preserving encryption and decryption respectively.

Enrolment Phase: During the enrolment phase given an input biometric template B_1, $\mathtt{P}_{\mathtt{en}}$ generates a helper data HD_1 and a cryptographic key K_1. From the second input biometric data B_2, another helper data HD_2 is generated with the help of a random secret key K_2. The key K_1 encrypts the helper data HD_2 to generate a transformed helper data HD_2^* using a format-preserving encryption scheme FFE.

Authentication phase: During the authentication phase, given HD_1 and B_1' as inputs, $\mathtt{P}_{\mathtt{au}}$ helps to recover the key K_1'. If the biometric template B_1' given during authentication is similar to B_1, i.e. $d(B_1, B_1') < t$, where t is a pre-defined threshold and d is the hamming distance between two binary strings, then K_1' is a correct key (i.e. equal to K_1). The FFD takes the key and encrypted helper data as inputs and decrypts the helper data as HD_2'. The second biometric template B_2' and helper data HD_2' are given as inputs to $\mathtt{Q}_{\mathtt{au}}$. If the symmetric difference between two biometric templates denoted as $|B_2 - B_2'| < \epsilon$, the key K_2' is recovered so that

$$H(K_2') = (H(K_2))' \tag{3.7}$$

$H(K_2')$ represents the hash of key K_2' generated from the module $\mathtt{Q}_{\mathtt{au}}$ (fuzzy vault scheme) in the authentication phase. $(H(K_2))'$ denotes the hash of the key K_2 which is stored as a part of helper data HD_2 during the enrolment in the encrypted form. If the above condition is satisfied,

1 is given as output to show a successful authentication denoted as

$$\mathcal{S}_3.Auth(HD_1, B_1', HD_2^*, B_2') \longrightarrow 1 \tag{3.8}$$

3.3.1.2.2 Security Analysis We discuss the security analysis of $\mathcal{S}_3$ in 2 different modes: online and offline mode as follows:

CASE A.(Offline mode setup). In the mentioned setup, the attacker can know any of the secret parameters including the key(s), B_1 or B_2 as inputs. Given the public parameters as

$HD_1 = (H(W_1), r_1, S_1)$ and

an encrypted helper data, HD_2^* such that

$FFE(K_1, HD_2) \longrightarrow HD_2^*$ where $HD_2 = (V_2, H(K_2))$

the attacker can perform several functions to generate the corresponding outputs as,

$\mathcal{S}_3.Enrol(B_1, B_2) \longrightarrow (HD_1, HD_2^*)$ where

$FFE(K_1, HD_2) \longrightarrow HD_2^*$,

$\mathtt{P}_{\mathtt{en}}(B_1) \longrightarrow (HD_1, K_1)$,

$\mathtt{Q}_{\mathtt{en}}(B_2, K_2) \longrightarrow HD_2$,

$\mathtt{P}_{\mathtt{au}}(HD_1, B_1') \longrightarrow (1/0, K_1')$,

$FFD(K_1', HD_2^*) \longrightarrow HD_2'$,

$\mathtt{Q}_{\mathtt{au}}(HD_2', B_2') \longrightarrow (1/0, K_2')$

The attacker can individually run any or all the above-mentioned functions or algorithms such as FFE, $\mathtt{P}_{\mathtt{en}}$, $\mathtt{Q}_{\mathtt{en}}$, $\mathtt{P}_{\mathtt{au}}$, $\mathtt{Q}_{\mathtt{au}}$ and FFD to get the respective outputs.

Under the mentioned setup, we analyse our first construction under three threat models as follows:

1. **Construction:** $\mathcal{S}_3$

Mode: Offline

Threat Model: As attacker knows B_2' where $|B_2 - B_2'| < \epsilon$, the aim is to recover B_1' or K_1' such that the authentication is successful.

Since, B_1' is unknown, correct key $K_1' = K_1$ is unknown. The attacker can guess a random value of K_1' and decrypt the encrypted helper data HD_2^* as

$$FFD(K_1', HD_2^*) \longrightarrow HD_2' \tag{3.9}$$

where $HD_2' = V_2' \| (H(K_2))'$. Since, B_2' is known to the attacker, therefore, using the correct B_2' and the derived value of vault V_2' corresponding to the guessed key K_1', the attacker can derive the key K_2' using polynomial interpolation. Using $H(K_2')$, the attacker can check the condition as given in (3.7). If the condition is satisfied, the authentication is successful for the corresponding guessed key K_1', else a new value of K_1' is guessed and attack procedure is repeated.

Since the success probability such that two hash values are equal is computed as $1/|H(K_2)|$, on average, the total number of trials required is $2^{|H(K_2)|}$. However, the total number of possible values of keys K_1 used to decrypt the helper data to obtain the valid sample space of the hash value $(H(K_2))'$ is smaller, given as $|K_1| \leq |H(K_2)|$. Therefore, the attacker would choose to guess the value of the key K_1 directly. Hence, the number of trials required to guess a correct key K_1', considering the internal security of fuzzy commitment scheme as p bits= $2^{min(|K_1|,p)}$, where $|.|$ denotes the size of the parameter.

Given correct key K_1' and the known random value r_1 obtained from public HD_1, the attacker can guess the correct biometric template B_1' with high probability[3], using pre-image attack by performing the following function given as,

$$H(B_1' \| r_1) = K_1'$$

Therefore, the number of trials required to guess B_1' given a correct key K_1' is equal to $2^{|K_1|}$.

2. **Construction:** $\mathcal{S}_3$

Mode: Offline

Threat Model: As attacker knows B_1' where $d(B_1, B_1') < t$, the aim is to recover B_2' or K_2' such that the authentication is successful.

The attacker can generate the key K_1' with the help of known B_1' and helper data HD_1 as $\mathtt{P}_{\mathtt{au}}(HD_1, B_1') \longrightarrow (1, K_1')$. Since, $d(B_1, B_1') < t$), the output key K_1' is a correct key and is equal to K_1 as generated during the enrolment phase. Attacker can then decrypt the encrypted helper data HD_2^* with the help of key K_1' using (3.9) to get the original helper data $HD_2' = (V_2' \| (H(K_2))')$.

Given $(H(K_2))'$ as the correct helper data, the attacker can guess all the possible values of key K_2' and compute $H(K_2')$. It can check if the hash of any of the guessed key K_2' satisfies the condition given in (3.7). Therefore, the number of trials required to guess the correct key K_2', while considering the security of fuzzy vault scheme as q bits = $2^{min(|K_2|,q)}$. Note that the pre-image attack complexity will not hold when $|K_2| \leq |H(K_2)|$, where H is the given hash function that acts as a random oracle.

With the help of correct key K_2' and decrypted value of vault V_2', B_2' can be recovered with high probability[3] by separating the genuine and chaff points.

3. **Construction:** $\mathcal{S}_3$

   Mode: Offline

   Threat Model: As attacker does not know B_1' and B_2', the aim is to recover any or all of the B_1', K_1', K_2' and B_2' such that authentication is successful.

   We use format-preserving encryption in our proposed scheme to encrypt the helper data HD_2. Since B_1' is unknown, the attacker can guess key K_1' and perform the function FFD as shown in (3.9) to get HD_2' from HD_2^*. However, since a format-preserving encryption scheme is used, the format of HD_2' remains the same and is equal to the format of HD_2^*, irrespective of the secret key applied for decryption. Therefore, the attacker cannot guess whether the key K_1' is correct or not by observing the format of HD_2'. The attacker would have to perform an exhaustive search on all the possible values of key K_1'. For each possible K_1', it can decrypt the helper data HD_2^* using (3.9) to get the decrypted helper

data $HD_2' = V_2'||(H(K_2))'$. Thus the attacker can store all the possible values of $(H(K_2))'$ corresponding to each guessed K_1' in the form of a table.

The attacker can guess the value of K_2' and check if $H(K_2')$ matches with one of the hashes stored in the table to satisfy (3.7). If the guessed K_2' for which $H(K_2')$ does not match with any of the table entries, the attacker could guess a new key K_2'.

Therefore, the best attack possible when both B_1' and B_2' are unknown is a time-memory tradeoff attack given by security bound denoted as Time $\times$ Memory. The time includes the number of trials taken for guessing all the possible values of the key given as K_1', considering the internal security of fuzzy commitment scheme as p bits = $2^{min(|K_1|,p)}$. The memory is used to store all the possible decrypted values of helper data and is given as $2^{|K_1|}$. Thus the security would be given as time-memory tradeoff attack bound plus the number of trials of K_2 that are needed to match the two hash values, considering internal security of fuzzy vault scheme as q bits. It is given as $2^{min(|K_1|,p)+|K_1|} + 2^{min(|K_2|,q)}$.

Given a valid K_1' which satisfies (3.7) and r_1, with high probability[3], the attacker can guess the correct biometric template B_1' using pre-image attack such that $H(B_1'\|r_1) = K_1'$ in $2^{|K_1|}$ trials.

With the help of a valid key K_2' which satisfies (3.7) and the vault V_2' decrypted by a valid key K_1', B_2' can be recovered by separating out the genuine and chaff points with high probability[3].

CASE B.(Online mode setup). In the mentioned setup, the attacker gives inputs in the form of biometric templates B_1 and B_2 to get the final output as 1 or 0 which indicates whether the authentication is successful or not. Given the public parameters as

$HD_1 = (H(W_1), r_1, S_1)$ and

an encrypted helper data, HD_2^* such that

$FFE(K_1, HD_2) \longrightarrow HD_2^*$ where

$HD_2 = V_2\|H(K_2)$

the attacker can run only the following 2 functions to generate the corresponding outputs as,

Given, $\mathcal{S}_3.Enrol(B_1, B_2) \longrightarrow (HD_1, HD_2^*)$ and

$\mathcal{S}_3.Auth(HD_1, B_1', HD_2^*, B_2') \longrightarrow 1/0$

Under the mentioned setup, we analyze our first construction under three threat models as follows:

1. **Construction:** $\mathcal{S}_3$

   Mode: Online

   Threat Model: As attacker knows B_2' where $|B_2 - B_2'| < \epsilon$, the aim is to get successful authentication, i.e. it satisfies (3.8).

   Since B_1' is unknown, K_1' is also unknown. The attacker can input a random or guessed biometric template B_1'. The system can then generate the corresponding K_1' as an application of the fuzzy commitment scheme given as $H(B_1'\|r_1) \longrightarrow K_1'$ (details of key generation are provided in Section 3.4 in (3.15). In general, the key K_1' is derived after truncation from the hash output $H(B_1\|r_1)$, or it can be of the same size as of hash output. Using the derived K_1', the system can decrypt the helper data internally to get decrypted helper data HD_2' with the help of (3.9), where $HD_2' = V_2'\|(H(K_2))'$. It implies that for different values of the B_1', corresponding values of vault V_2' would be derived by the system. B_2' is known to the attacker and is provided as another input. Using the correct B_2' and the possible values of V_2', the system can derive the corresponding key K_2'. It can compute $H(K_2')$ and verify it with the hash value $(H(K_2))'$ obtained as the part of helper data HD_2'. If (3.7) is satisfied, the key K_2' is a correct key, else the process is repeated with a new value of B_1'.

   Since the key K_1' is used to decrypt the helper data, the attacker needs to perform $H(B_1'\|r_1)$ number of trials of B_1' to get all the possible values of key K_1'. Therefore, the number of trials of B_1' required to get successful authentication = $2^{|H(B_1\|r_1)|}$ when key K_1' is derived after truncation from the hash output $H(B_1'\|r_1)$ and is equal to $2^{|K_1|}$ if key $|K_1'|$ = $|H(B_1'\|r_1)|$.

2. **Construction:** $\mathcal{S}_3$

Mode: Online

Threat Model: As attacker knows B_1' where $d(B_1, B_1') < t$, the aim is to get successful authentication, i.e. it satisfies (3.8).

Attacker can provide B_1' as input, using which the system can generate the correct key K_1' as $\mathtt{P_{au}}(HD_1', B_1') \longrightarrow (1, K_1')$. Since, $d(B_1, B_1') < t$, K_1' is a correct key with high probability[3] and is equal to K_1 generated during the enrolment phase. The key can be used by system to decrypt HD_2^* to get the correct helper data $HD_2' = (V_2' \| (H(K_2))')$ using (3.9).

The attacker can guess B_2' such that on providing B_2' to the module Q_{au}, the system will compute the key K_2' using B_2' and polynomial interpolation on correct value of vault V_2'. If B_2' is correct, with high probability[3], K_2' is correctly derived and is verified using the the condition given in (3.7).

Since $(H(K_2))'$ is always correct, therefore, the number of trials of B_2' required for successful authentication is equal to the sample space of $|K_2| = 2^{|K_2|}$. Note that the pre-image attack complexity will not hold when $|K_2| \leq |H(K_2)|$, where H is the given hash function that acts as a random oracle. We consider that within $2^{|K_2|}$ trials of B_2', the system would be able to get a correct key K_2' to authenticate the attacker successfully with high probability[3].

3. **Construction:** $\mathcal{S}_3$

   Mode: Online

   Threat Model: As the attacker does not know B_1' and B_2', the aim is to get successful authentication, i.e. it satisfies (3.8).

   In the online attack mode, the attacker has no memory. The attacker would need to guess all possible combinations of B_1' and B_2' such that (3.7) is satisfied, which gives output 1 to denote a successful authentication. Therefore, the number of trials that need to be performed to obtain the correct matching of two hash values is equivalent to the pre-image attack complexity, assuming H as a random oracle. Thus the number of trials is equal to $2^{|H(K_2)|}$. Note that the attacker cannot obtain or guess the key K_2 during an online attack.

Therefore, the scope of performing a collision attack on the system by finding two keys, K_2 and K_2' such that their hash matches with each other is negligible.

3.3.1.3 FV-then-FC: $\mathcal{S}_4$

3.3.1.3.1 Construction FV-then-FC is constructed by combining a fuzzy vault scheme (FV) with a fuzzy commitment scheme (FC). Fig. 3.5 shows the enrolment and authentication phases.

Enrolment Phase: During the enrolment phase, $\mathtt{Q}_{\mathtt{en}}$ generates a helper data HD_1, given a secret cryptographic key K_1 generated internally and the user's biometric template B_1. From the second input biometric data B_2, another helper data HD_2 is generated along with a key K_2. The key K_1 encrypts the helper data HD_2 to generate a transformed helper data HD_2^* using a format-preserving encryption scheme FFE. K_2 does not play any role in enrolment.

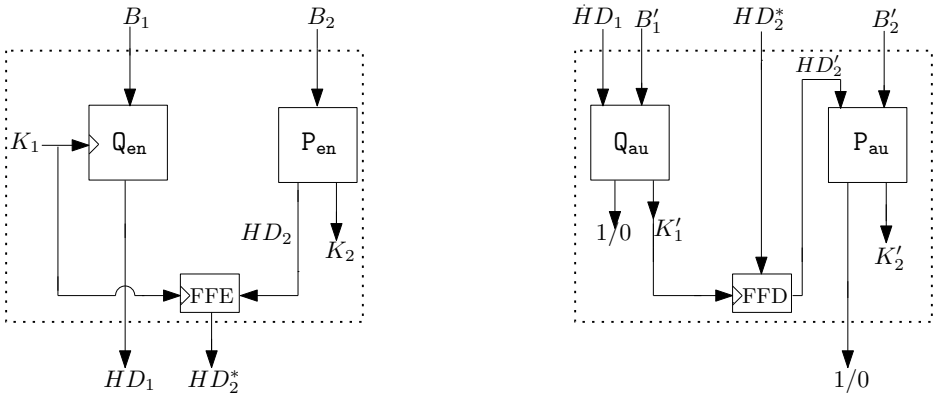


Figure 3.5: Enrolment and authentication phase of possible combinations from FV-then-FC. $\mathtt{P}_{\mathtt{en}}$ and $\mathtt{Q}_{\mathtt{en}}$ represent the enrolment modules for fuzzy commitment and fuzzy vault scheme respectively. Similarly, $\mathtt{P}_{\mathtt{au}}$ and $\mathtt{Q}_{\mathtt{au}}$ represent the authentication modules for fuzzy commitment and fuzzy vault scheme respectively. Refer Fig. 2.2 and 2.3 for details. FFE and FFD denotes the format-preserving encryption and decryption respectively.

Authentication phase: During the authentication phase, given HD_1 and B_1' as inputs, $\mathtt{Q}_{\mathtt{au}}$ helps to recover the key K_1'. If the biometric template B_1' given during authentication is similar to B_1, i.e. $|B_1 - B_1'| < \epsilon$, then K_1' is a correct key (i.e. equal to K_1). The FFD takes the key and encrypted helper data as inputs and generates the decrypted helper data HD_2'. With the help of second biometric template B_2' and helper data HD_2', hash verification is performed inside the

$\mathtt{P}_{\mathtt{au}}$ module such that

$$H(W_2') = (H(W_2))' \tag{3.10}$$

where $H(W_2')$ represents the hash of codeword W_2' generated internally from P_{au_2} (fuzzy commitment scheme) in the authentication phase. $(H(W_2))'$ represents the hash of the codeword W_2, which is stored as a part of helper data HD_2 during the enrolment in the encrypted form. If the condition is satisfied, authentication is successful with output 1 and is denoted as

$$\mathcal{S}_4.Auth(HD_1, B_1', HD_2^*, B_2') \longrightarrow 1 \tag{3.11}$$

K_2' generated as output does not play any authentication role.

3.3.1.3.2 Security Analysis We discuss the security analysis of $\mathcal{S}_4$ in 2 different modes: online and offline mode as follows:

CASE A.(Offline mode setup). In the mentioned setup, the attacker may know some of the secret parameters including the key(s), B_1 or B_2 as inputs. Given the public parameters as

$HD_1 = (V_1, H(K_1))$ and

an encrypted helper data, HD_2^* such that

$FFE(K_1, HD_2) \longrightarrow HD_2^*$ where

$HD_2 = (H(W_2), r_2, S_2)$,

the attacker can perform several functions to generate the corresponding outputs as,

$\mathcal{S}_4.Enrol(B_1, B_2) \longrightarrow (HD_1, HD_2^*)$ where

$\mathtt{Q}_{\mathtt{en}}(B_1, K_1) \longrightarrow HD_1$,

$\mathtt{P}_{\mathtt{en}}(B_2) \longrightarrow (HD_2, K_2)$,

$\mathtt{Q}_{\mathtt{au}}(HD_1, B_1') \longrightarrow (1/0, K_1')$,

$FFD(K_1', HD_2^*) \longrightarrow HD_2'$,

$\mathtt{P}_{\mathtt{au}}(HD_2', B_2') \longrightarrow (1/0, K_2')$

The attacker can individually run any or all the above-mentioned functions or algorithms such as FFE, $\mathtt{P}_{\mathtt{en}}$, $\mathtt{Q}_{\mathtt{en}}$, $\mathtt{P}_{\mathtt{au}}$, $\mathtt{Q}_{\mathtt{au}}$ and FFD to get the respective outputs.

Under the mentioned setup, we analyse our first construction under three threat models as follows:

1. **Construction:** $\mathcal{S}_4$

   Mode: Offline

   Threat Model: As attacker knows B_2' where $d(B_2, B_2') < t$, the aim is to recover B_1' or K_1' such that the authentication is successful.
   Since, B_1' is unknown, correct key $K_1' = K_1$ is unknown. The attacker can guess K_1' and perform the function FFD to get HD_2' given as,

   $$FFD(K_1', HD_2^*) \longrightarrow HD_2' \tag{3.12}$$

   where $HD_2' = ((H(W_2))', r_2', S_2')$. Further, the biometric template B_2' is known to the attacker, therefore, using the correct B_2' and all the values of S_2' derived from the helper data, the attacker can perform $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2'. From W_2', the attacker can compute $H(W_2')$ to check if the condition in (3.10) is satisfied. If the condition is satisfied, the authentication is successful for the corresponding guessed key K_1', else a new value of K_1' is guessed and attack procedure is repeated.

   Since the success probability such that two hash values are equal is computed as $1/|H(W_2)|$, on average, the total number of trials required is $2^{|H(W_2)|}$. However, the total number of possible values of keys K_1 used to decrypt the helper data to obtain a valid sample space of hash value $(H(W_2))'$ is smaller, given as $|K_1| \leq |H(W_2)|$. Therefore, the attacker would choose to guess the values of the key K_1 directly. Hence, the number of trials required to guess a correct key K_1', considering the internal security of fuzzy vault scheme as q bits= $2^{min(|K_1|,q)}$, where $|.|$ denotes the size of the parameter.

Given correct key K_1' and V_1', the attacker can recover the correct biometric template B_1' with high probability[3], by separating out the genuine and chaff points.

2. **Construction:** $\mathcal{S}_4$

   Mode: Offline

   Threat Model: As attacker knows B_1' where $|B_1 - B_1'| < \epsilon$, the aim is to recover B_2' or K_2' such that the authentication is successful.

   The attacker can generate the key K_1' with the help of known B_1' and helper data HD_1 as $\mathtt{Q}_{\mathtt{au}}(HD_1', B_1') \longrightarrow (1, K_1')$.

   $|B_1 - B_1'| < \epsilon$, K_1' is a correct key and is equal to K_1 generated during the enrolment phase. Attacker can then decrypt the given HD_2^* with the help of key K_1' using (3.12) to get the correct $HD_2' = ((H(W_2))', r_2', S_2')$.

   Note that K_2' does not play any role in enrolment or authentication phase. The attacker can guess B_2' directly such that $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2'. From W_2', the attacker can compute $H(W_2')$. Attacker can then check the condition given in (3.10). Therefore, the number of trials needed to guess a correct B_2' with high probability[3]is equal number of trials needed to guess a correct hash of codeword $H(W_2'')$ that will match with the given hash $(H(W_2))'$. It is equal to $2^{min(|H(W_2)|,p)}$ trials, considering the internal security of second fuzzy commitment scheme as p bits.

3. **Construction:** $\mathcal{S}_4$

   Mode: Offline

   Threat Model: As attacker does not know B_1' and B_2', the aim is to recover any or all of the B_1', K_1', K_2' and B_2' such that authentication is successful.

   We use format-preserving encryption in our proposed scheme to encrypt the helper data HD_2. Since B_1' is unknown, the attacker can guess key K_1' and can perform the function FFD as shown in (3.12) to get HD_2' from HD_2^*. However, since a format-preserving encryption scheme is used, the format of HD_2' remains the same and is equal to the format

of HD_2^*, irrespective of the secret key applied for decryption. Therefore, the attacker cannot guess whether the key K_1' is correct or not by observing the format of HD_2'. The attacker would have to perform an exhaustive search on all the possible values of key K_1'. For each possible K_1', it can decrypt the helper data HD_2^* to get the decrypted helper data HD_2' which is given as $HD_2' = ((H(W_2))', r_2', S_2')$. The attacker can store all the possible values of $(H(W_2))'$ corresponding to each guessed K_1' in the form of a table.

The attacker can then guess the value of $H(W_2')$ and can check if $H(W_2')$ matches with one of the hashes $(H(W_2))'$ stored in the table to satisfy (3.10). Note that the attacker would prefer to guess the hash of codeword directly rather than the codeword since $|W_2| >> |H(W_2)|$. If the guessed $H(W_2')$ does not match any of the table entries; the attacker can guess a new hash value.

Therefore, the best attack possible when both B_1' and B_2' are unknown is a time-memory tradeoff attack given by security bound denoted as Time $\times$ Memory. The time includes the number of trials taken for guessing all the possible values of the key given as K_1' considering the internal security of fuzzy vault scheme as q bits = $2^{min(|K_1|,q)}$. The memory is used to store all the possible decrypted values $(H(W_2))'$ as a part of helper data. It is given as $2^{|K_1|}$. Thus the security would be given as time-memory tradeoff attack bound plus the number of trials of $H(W_2)$ that are needed to match the two hash values, considering internal security of fuzzy commitment scheme as p bits. It is given as $2^{min(|K_1|,q)+|K_1|} + 2^{min(|H(W_2)|,p)}$.

Given a valid K_1' and public vault V_1, the attacker can guess the correct biometric template B_1' with high probability[3], by separating out the genuine from chaff points.

The attacker can recover B_2' with high probability[3] by guessing B_2' such that $W_2'' = B_2' \oplus S_2'$ where W_2'' can be decoded using error correcting codeword to obtain W_2' with the condition that $H(W_2') = (H(W_2))'$. The number of trials needed to guess a correct B_2' with high probability[3] is equal to guessing a correct hash of codeword $H(W_2')$ that will match with the given hash $(H(W_2))'$. It is equal to $2^{|H(W_2)|}$ trials.

CASE B.(Online mode setup). In the mentioned setup, the attacker gives inputs in the form of biometric templates B_1 and B_2 to get the final output as 1 or 0 which indicates whether the

authentication is successful or not. Given the public parameters as

$HD_1 = (V_1 \| H(K_1))$ and

an encrypted helper data, HD_2^* such that

$FFE(K_1, HD_2) \longrightarrow HD_2^*$ where $HD_2 = (H(W_2), r_2, S_2)$,

the attacker can run only the following 2 functions to generate the corresponding outputs as,

$\mathcal{S}_4.Enrol(B_1, B_2) \longrightarrow (HD_1, HD_2^*)$,

$\mathcal{S}_4.Auth(HD_1, B_1', HD_2^*, B_2') \longrightarrow 1/0$

Under the mentioned setup, we analyze our first construction under three threat models as follows:

1. **Construction:** $\mathcal{S}_4$

   Mode: Online

   Threat Model: As attacker knows B_2' where $d(B_2, B_2') < t$, the aim is to get successful authentication, i.e. it satisfies (3.11).

   Since, B_1' is unknown, K_1' is also unknown. The attacker can input the random biometric templates B_1'. The system can then generate a corresponding K_1' by applying polynomial interpolation on the guessed B_1' and given V_1 obtained from HD_1. The key is generated as $\mathtt{Q_{au}}(HD_1, B_1') \longrightarrow (1/0, K_1')$. Using the derived K_1', the system can decrypt the helper data internally to get decrypted helper data $HD_2' = ((H(W_2))', r_2', S_2')$.

   It implies that for different values of guessed B_1', corresponding secure sketch value S_2' would be derived. B_2' is known to the attacker, which could be provided as an input to the system. Therefore, using the correct B_2' and the derived values of S_2', the system can perform error correcting code decoding by performing $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2'. System can verify the condition given in (3.10). The process is repeated until the authentication is successful.

   Note that the attacker can only give B_1' as input to the system and not the key K_1'. Therefore, the number of trials of B_1' needed to get correct key K_1' which further leads to successful

authentication is equal to $2^{|K_1|}$, where $|.|$ denotes the size of the parameter. We consider that within $2^{|K_1|}$ trials of B_1', the system would be able to get a correct key K_1' that will decrypt the helper data correctly with high probability[3].

2. **Construction:** $\mathcal{S}_4$

   Mode: Online

   Threat Model: As attacker knows B_1' where $|B_1 - B_1'| < \epsilon$, the aim is to get successful authentication, i.e. it satisfies (3.11).

   Attacker can provide B_1' as input. Using it, the system can generate the key K_1' as $\mathtt{P_{au}}(HD_1', B_1') \longrightarrow (1, K_1')$. Since, $|B_1 - B_1'| < \epsilon$, K_1' is a correct key with high probability[3] and is equal to K_1 generated during the enrolment phase. The key can be used by system to decrypt HD_2^* to get the correct helper data $HD_2' = ((H(W_2))', r_2', S_2')$. using (3.12).

   The attacker can provide B_2' such that on providing B_2' to the module P_{au}, the system can compute $W_2'' = B_2' \oplus S_2'$ where W_2'' is decoded using error correcting codeword to obtain W_2'. If the condition given in (3.10) is satisfied, the system will show successful authentication. Thus, the attacker can get to know if the given B_2' is correct or not.

   Therefore, the number of trials of B_2' required for successful authentication is equal to guessing a correct hash of codeword $H(W_2')$ that will match with the given hash $(H(W_2))'$. It sums to $2^{|H(W_2)|}$ trials. Note that K_2' does not play any role in the enrolment or authentication phase.

3. **Construction:** $\mathcal{S}_4$

   Mode: Online

   Threat Model: As the attacker does not know B_1' and B_2', the aim is to get successful authentication, i.e. it satisfies (3.11).

   In the online attack mode, the attacker has no memory. The attacker would need to guess all possible combinations of B_1' and B_2' such that (3.10) is satisfied, which gives output 1 to denote a successful authentication. Therefore, the number of trials that need to be performed to obtain the correct matching of two hash values is equivalent to the pre-image

attack complexity, assuming H as a random oracle. Thus the number of trials is equal to $2^{|H(W_2)|}$.

3.4 Secure Construction of BIOFUSE: S-BIOFUSE ($\mathcal{S}_3$)

In Section 3.3, we discussed the security analysis of four possible constructions in which we can combine two biocryptosystems. We found that the security bound for all the four constructions are comparable to each other. However, we consider the case FC-then-FV denoted by $\mathcal{S}_3$ as the most secure case out of all, and we named it as S-BIOFUSE. The limitations of the other 3 cases are:

- **FC-then-FC:** The fuzzy commitment scheme takes a binary string as input. Hence in the case of fingerprint, etc., where features are not present in binary format, an additional feature extraction or transformation tool would be required.
- **FV-then-FV:** One of the vault data V_1 will remain unencrypted, which is prone to multiple security attacks such as brute force attack on vault, correlation attack, etc. Further, to use any biometric template with features in the binary string format, additional feature extraction or transformation tool would be required with the fuzzy vault.
- **FV-then-FC:** Like the above case, one of the vault data V_1 will remain unencrypted, which is prone to multiple security attacks such as brute force attack on vault, correlation attack, etc.

We now discuss the enrolment and authentication phase of S-BIOFUSE in detail.

3.4.1 Enrolment Phase

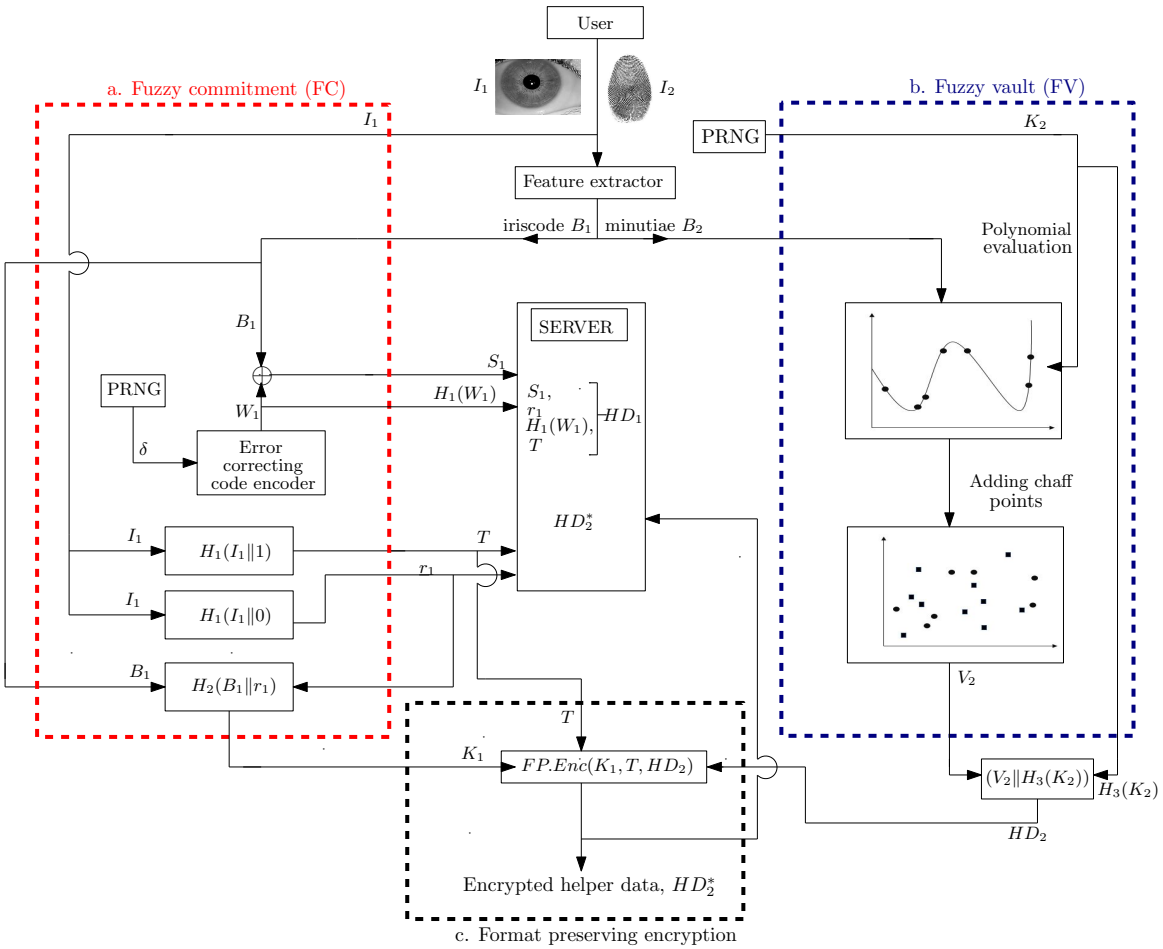


Figure 3.6: **Enrolment Phase- a.**The fuzzy commitment scheme takes iris I_1 from which iriscode B_1 is extracted. It generates helper data HD_1 denoted as $(H(W_1), r_1, S_1, T)$ along with the cryptographic key K_1. **b.** The Fuzzy vault scheme takes fingerprint template as minutiae B_2 along with an internal secret key K_2. Helper data HD_2 is generated as output and is denoted as $(V_2, H(K_2))$. **c.** HD_2 is encrypted by format-preserving encryption $FP.Enc$ using key K_1 to generate encrypted output HD_2^*.

The enrolment phase is described using Algorithm 1 and is shown in Fig. 3.6. The user provides first biometric characteristics, I_1 that represents the iris image (let say) from which a binary string B_1 known as iriscode is extracted [14].

To handle the errors present in the iriscode, an error correcting codeword W_1 is generated internally from a secret, pseudorandom message δ.

A secure sketch value S_1 is constructed using fuzzy commitment scheme as $S_1 \longleftarrow B_1 \oplus C_1$.

Further, a tweak value T used by the format-preserving encryption scheme is generated as

$$T \longleftarrow H_1(I_1 \| 1) \tag{3.13}$$

Algorithm 1: Enrolment Phase

Input: Iris image I_1, Fingerprint image I_2, iriscode B_1,
minutiae set B_2

Output: Helper data HD_1, HD_2^*

1. ▷ **Fuzzy commitment scheme:** outputs $HD_1 = (H(W_1), r_1, S_1, T)$
2. $W_1 \xleftarrow{\delta} \Delta$ ▷ Δ denotes a random secret message
3. $S_1 = B_1 \oplus C_1$
4. $T = H_1(I_1 \| 1)$ ▷ H_1, H_2, H_3 : instantiates of hash functions
5. $r_1 = H_1(I_1 \| 0)$
6. $K_1 = H_2(B_1 \| r_1)$
7. ▷ **Fuzzy vault scheme:** outputs $HD_2 = V_2 \| H_3(K_2)$
8. **for** $i = 1$ to l ▷ l : number of minutiae points in set
9. $p(B_{2_i}) = K_2^{n-1} B_{2_i}^{n-1} + K_2^{n-2} B_{2_i}^{n-2} + \ldots K_2^1 B_{2_i}^1 + K_2^0$ ▷ $p(B_{2_i})$: polynomial construction with degree n
10. $G = \{(B_{2_i}, p(B_{2_i}))\}$ ▷ G : genuine points
11. **for** $i = l + 1$ to v ▷ v : total number of points in vault
12. $Wh = \{(z_i, u_i)\}$ is generated such that $z_i \notin \{B_{2_i}\}$ and $u_i \notin \{p(B_{2_i})\}$ ▷ Wh : chaff points
13. $V_2 = G \cup Ch$
14. ▷ **Format-preserving Encryption:** outputs encrypted message HD_2^*
15. $HD_2^* = FP.Enc(K_1, T, HD_2)$

A random value r_1 is also generated from the input I_1 as

$$r_1 \longleftarrow H_1(I_1 \| 0) \tag{3.14}$$

These output values- S_1, $H_1(W_1)$, r_1 and T are stored as helper data HD_1 on the database server. Note that tweak T will be different every time (for similar but not identical samples of a particular instance) since it is generated from the hash of biometric characteristics. It prevents nonce misuse in the format-preserving encryption scheme. For simplicity, we do not include T while calculating security bounds in Section 3.3. H_1, H_2, H_3 denotes the instances of hash function that acts as a random oracle. Using the random value r_1 in (3.14), a key K_1 is generated from biometric template B_1 through fuzzy extractor as

$$K_1 \longleftarrow H_2(B_1 \| r_1) \tag{3.15}$$

Simultaneously, using the second biometric characteristics I_2 that represents a fingerprint, unique features known as minutiae points are extracted [61] and quantized to generate a set B_2 of the unordered points known as genuine points. A vault is constructed from the minutiae points and the random chaff points using an internal random key K_2 as proposed by Nandakumar et.al [39].

In BIOFUSE, instead of implementing CRC [34] to verify the correct polynomial during authentication, we use the hash of the secret key K_2 to verify the reconstruction of the correct polynomial during authentication. HD_2 represents the concatenation of vault V_2 and the hash of the secret key used in the fuzzy vault scheme.

The key K_1 encrypts the helper data HD_2 using format-preserving encryption as

$$HD_2^* \longleftarrow FP.Enc(K_1, T, HD_2)$$

.

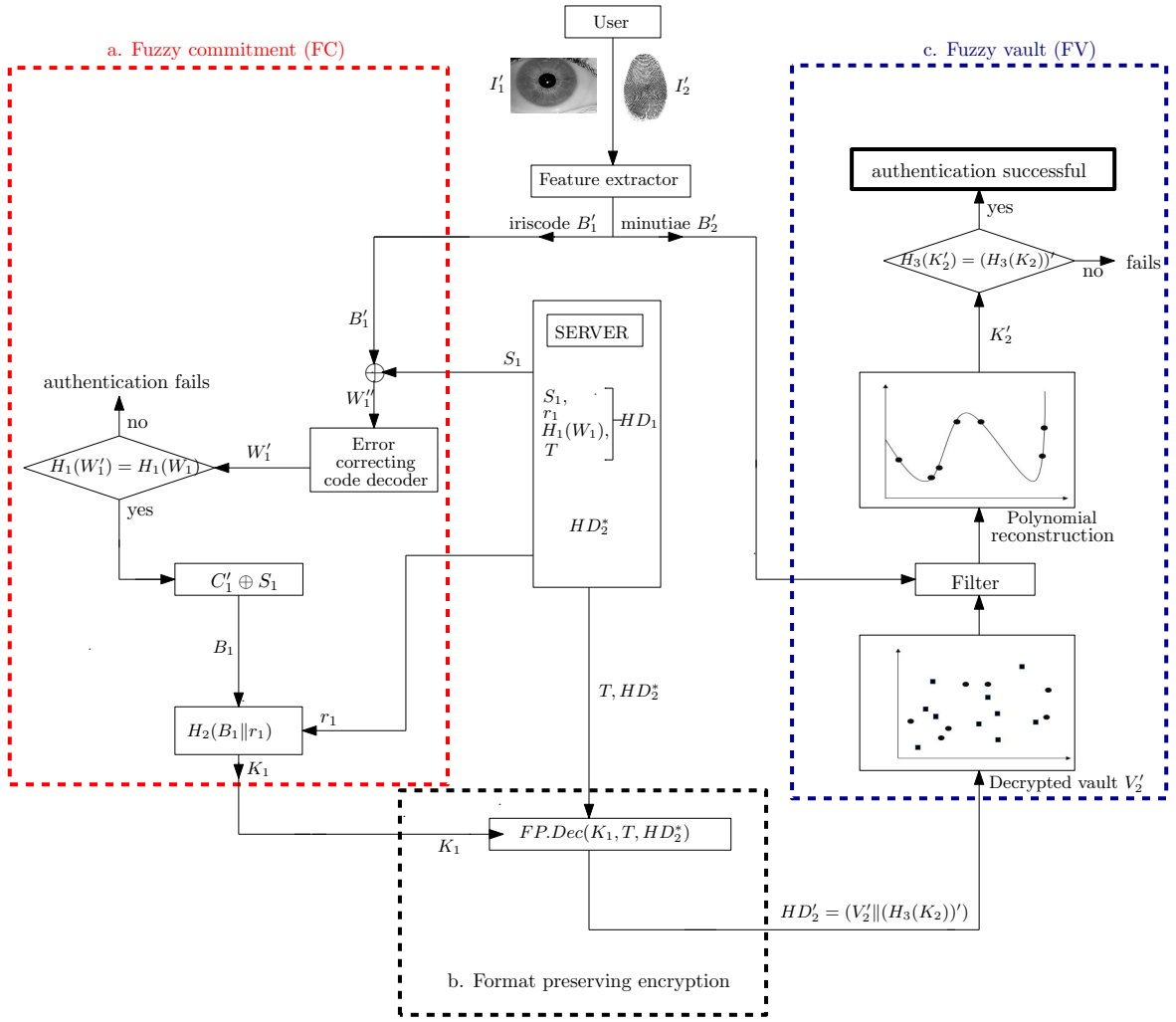


Figure 3.7: **Authentication phase. a.** Fuzzy commitment scheme takes input B_1' from the user and using S_1 stored, it generates the codeword W_1'' decoded to W_1' if the user is genuine. The key K_1 is generated by recovering original B_1. **b.** The key K_1 is used to decrypt HD_2^* using FPE scheme to get the original helper data as $HD_2' = V_2' \| (H_3(K_2))'$ **c.** Using V_2' and query minutiae point's set B_2' obtained from I_2', the secret key K_2' is derived if sufficient number of query points matches with the vault points. A user is authenticated if $H_3(K_2')$ is equal to $(H_3(K_2))'$.

3.4.2 Authentication Phase

The authentication phase is described by Algorithm 2 and is shown in Fig. 3.7. During authentication, using the first biometric characteristics I_1', iriscode B_1' is extracted. Using fuzzy de-commitment scheme, the error correcting codeword W'' is obtained which is decoded to get W'. If $H_1(W_1) = H_1(W_1')$, the original template $B_1'' = B_1$ is recovered correctly which derives the key $K_1' = K_1$ using (3.15).

Algorithm 2: Authentication Phase

Input: Iris template I_1', Fingerprint template I_2', extracted iriscode B_1', extracted minutiae set B_2',

Server: Helper data:HD_1, HD_2^*

Output: $1/0$

▷ **Fuzzy commitment scheme:** outputs key K_1'

$HD_1 = (H(W_1), r_1, S_1)$

$W_1'' = B_1' \oplus S_1$

W_1'' is decoded to W_1' if $d(B_1, B_1') < t$, ▷ t : number of errors in biometric template

if $H_1(W_1') = H_1(W_1)$

  $B_1'' = W_1' \oplus S_1$, ▷ $B_1'' = B_1$

  $K_1' = H_2(B_1''\|r_1)$

▷ **Format-preserving Decryption:** outputs decrypted message HD_2'

$HD_2' = FP.Dec(K_1, T, HD_2^*)$

$HD_2' = (V_2'\|(H_3(K_2))')$

▷ **Fuzzy vault scheme:** outputs the secret key K_2'

Using $V_2' = G \cup Ch$ and B_2', reconstruct the polynomial that returns key K_2'

if $(H_3(K_2') = (H_3(K_2))')$

  Return 1, User is successfully authenticated

The ciphertext HD_2^* is decrypted using the format-preserving decryption function as

$$HD_2' \longleftarrow FP.Dec(K_1', T, HD_2^*)$$

On successful decryption, the vault V_2' and $(H_3(K_2))'$ are obtained. From the second input biometric characteristics I_2', the genuine feature set B_2' is generated. Using polynomial interpolation, the polynomial is reconstructed that recovers the secret key $K_2' = K_2$. The user is successfully authenticated after the key is validated by comparing the hashes of stored key and the recovered

key as $H_3(K_2) = (H_3(K_2))'$.

3.4.3 Rationale behind the use of format-preserving encryption

Format-preserving encryption (FPE) [52] refers to encryption of data in a way such that the format of output (ciphertext) is the same as the format of input (plaintext), including the length of data. For example, if a credit card number consists of 14 digits where digits can take value 0-9, the encrypted credit card format after using format-preserving encryption would also remain the same, i.e., 14 digits with each digit accepting value from 0-9. Format-preserving encryption is designed for data that is not necessarily binary and can be any finite set of symbols, like the decimal numerals. Thus, FPE encrypts sensitive information and can be used for encryption in database applications that do not support changes to the format or data length.

In Table 3.4, we provide an example depicting the working of our proposed scheme. We use an iriscode and a fingerprint sample in the example. The length of tweak and a random number, T and r is 128 bits (by omitting the extra bits from the 256 bits hash output). The key size is equal to 256 bits for both K_1 and K_2.

Considering the example with,

K_1=$39c9ff0d1ffd9640da47eb638fb1d5c8295413cb8be8fc053b63c2a3b232e4ea$,

T= $bca574d6773fd57d845fcf271cc69830$

The helper data contains vault points in the numeric form (with radix= 10) and the hash of the key, $H_3(K_2)$ in the form of a binary string and is given as,
HD_2= $(V_2 || H_3(K_2))$= 90 424 67 693 . . . 13281 369 315 13 $\|$ 00100 . . . 10000

For simplifying the implementation of FPE, we encrypt the numeric data (vault points) and binary data (hash of key K_2) individually while ignoring the white spaces between the vault points.

Table 3.4: Examples for S-BIOFUSE.

Enrolment Input: I_1,I_2,	**Authentication Input:** I'_1,I'_2,
$B_1 = 1111111100.....1100000111$,	$B'_1 = 1111100000.....1100000111$,
B_2= [(90 424 67 6), (93 450 56 19), . . .]	B'_2= [(98 259 56 15), (98 250 247 14), . . .]
Output:HD_1,HD_2^*	**Server:**HD_1,HD_2^*
▷ **Fuzzy commitment scheme:**	**Output:** 1/0
outputs $HD_1 = (H_1(W_1), rand_1, \delta_1, T)$	▷ **Fuzzy commitment scheme:**
$\delta_1 = B_1 \oplus W_1$ =1101110100 . . . 0000110001	outputs key K'_1
$H_1(W_1) = 0xf7cadb0f \ldots a080b3dc$	Using HD_1 and given input B'_1,
$T = 0xbca57 \ldots 9830$	W''_1 is decoded to W'_1, we get
$rand_1 = 0x67c7 \ldots 7f07$	$B''_1 = B_1 = 1111111100.....1100000111$
$K_1 = 0x39c9ff \ldots 232e4ea$	$K'_1 = 0x39c9ff \ldots 232e4ea$
▷ **Fuzzy vault scheme:**	▷ **Format-preserving Decryption:**
outputs $HD_2 = V_2 \| H_3(K_2)$	outputs decrypted message HD'_2
From minutiae points B_2, generate vault	$HD'_2 = FP.Dec(K_1, T, HD_2^*)$
V_2= 90 424 67 6 . . . 13281 369 315 13	= 90 424 67 6 . . . 13281 369 315 13 ‖ 00100 . . . 10000
$H_3(K_2)$ = 00100 . . . 10000	▷ **Fuzzy vault scheme:**
▷ **Format-preserving Encryption:**	outputs the secret key K'_2
outputs encrypted message HD_2^*	reconstruct the polynomial to return K'_2
$HD_2^* = FP.Enc(K_1, T, HD_2)$	$(H_3(K'_2) = (H_3(K_2))')$
= 46 131 10 508 . . . 90573 163 008 23 ‖ 11000 . . . 01101	= 00100 . . . 10000
	Return 1, User is successfully authenticated

Case-1: Proposed approach- Using FPE with correct key K_1 (encryption and decryption of HD_2 using format-preserving encryption)

In this scenario, we use format-preserving encryption to encrypt the helper data generated by the second biocryptosystem (fuzzy commitment or fuzzy vault). The key used to encrypt the helper data is derived from the first biocryptosystem (fuzzy commitment or fuzzy vault). In the offline attack mode, assume that the attacker does not know any biometric templates B_1 or B_2.

By applying format-preserving encryption on HD_2, we get
HD_2^*= 46 131 10 508 . . . 90573 163 008 23 ‖ 11000 . . . 01101

Note that FPE preserves the helper data's length and format, which further means that the numeric value is encrypted to a numeric format, and the binary string is encrypted to generate

a corresponding encrypted binary string. Also, the length of the original helper data and the encrypted helper data is the same. For example, the first vault point, 90, is encrypted to 37; both are numeric and have the same length equals to 2.

The decrypted helper data is obtained using format-preserving decryption with key K_1. According to the length of vault points and binary string, we can parse the decrypted helper data, respectively, as in the original helper data. It is given as

HD'_2= 90 424 67 6 . . . 13281 369 315 13 || 00100 . . . 10000

In such a case, an attacker cannot guess the correct key since the decrypted message would always be in the same format as the original plaintext, regardless of the key. Thus, the attacker will not be able to distinguish a correct guess of the key K_1 from an incorrect guess. The best attack possible would be a time-memory tradeoff attack given by security bound, as shown in Section 3.3. The security bound is approximated as $(2^{|K_1|} \times 2^{|K_1|})$. Considering $|K_1| \approx |K_2|$, it can be observed that the bound is considerably higher than the security bound provided by the scheme without format-preserving encryption being used.

Note, the length of data in HD_2 is public and does not reveal any significant information about the helper data itself.

$HD'_2 = HD_2$ since the key is correct.

Case-2: Using format-preserving decryption with the incorrect key $K'_1 \neq K_1$

Let K'_1= **123d**$ff0d1ffd9640da47eb638fb1d5c8295413cb8be8fc053b63c2a3b232e4ea$

The decrypted helper data is given as,

HD'_2= 37 639 60 4 . . . 76899 643 657 41 || 11111 . . . 10011

Clearly, $HD'_2 \neq HD_2$.

However, it has the same format and length (vault points in numeric form and a hash of key as binary string) as that of the original helper data HD_2.

Thus, the attacker cannot guess the correct key since the decrypted message would always be in the same format as the original plaintext, even when the guessed key is wrong.

Case-3: The encryption and decryption of the helper data is done using the AES-256 algorithm (a block cipher based encryption algorithm- AES 256 with output in $GF(2^8)$)

In the case when encryption is done using block cipher mode of operations such as AES-256 with CBC mode with key K_1 derived from the first biocryptosystem, it may be possible for the attacker to guess the correct key K_1 by checking the format of the decrypted message.

We use the same values of helper data HD_1 and keys K_1 and K_1' as used in Case-1 and Case-2.

Encryption of HD_2 using key K_1 will give,
HD_2^*=$9f2314093bb \ldots fbbdf47ab918f7b989 \| 57930aec14 \ldots d7ed3d4842fb$

The decryption of HD_2^* with the correct key K_1 will generate $HD_2' = HD_2$ after hex to ASCII code conversion. It is given as,
HD_2'= 90 424 67 6 ... 13281 369 315 13 $\|$ 00100 ... 10000

Whereas, the decryption of HD_2^* with the incorrect key K_1' will give helper data as,
HD_2'= $5b659ac446c2 \ldots 850bdb4b6c62b5b6 \| 27afe3b4df \ldots 13888bf9c871$

that gives pseudorandom (gibberish) data of a fixed length as an output when hex to ASCII code conversion is performed on it. It can help the attacker to validate if the key used in decryption is correct or incorrect by checking the format (which in the case of AES is pseudorandom data) of the decrypted helper data after ASCII code conversion. For a system with two biocryptosystems, each with security parameter let say K_1 and K_2 respectively, the overall security bound (in terms of brute force attack complexity) of the system will become $(2^{K_1} + 2^{K_2})$.

Finally, we could ensure that when FPE is used, the attacker cannot perform brute force on the key K_1. Thus, FPE helps in efficiently encrypting and decrypting the helper data, even though the helper data constitutes different formats' values.

3.4.4 Security Analysis of S-BIOFUSE ($\mathcal{S}_3$)

We analyze the privacy and security properties of S-BIOFUSE ($\mathcal{S}_3$).

3.4.4.1 Key Inversion Attack

In the fuzzy vault, for successful authentication, the following equation needs to be verified: $(H_3(K_2))' = H_3(K_2')$, where $(H_3(K_2))'$ is the hash of original key stored on the server during enrolment and $H_3(K_2')$ is the hash of guessed or the recovered key during authentication. If the hash of the key is stored directly on the server without encryption, the attacker can directly verify the match between the hash values by brute force attack on the key and get unauthorized access to the system without the need of any of the biometric templates. It is known as the key inversion attack. In BIOFUSE, for successful authentication, the user has to decrypt the encrypted message HD_2^* first using the format-preserving encryption scheme, which further requires cryptographic key K_1 to be generated from the first biometric template shown in (3.15). Using the decrypted message HD_2, the hash of key $(H_3(K_2))'$ is retrieved. Thus, the key inversion attack is difficult to achieve in our proposed scheme.

3.4.4.2 Blend Substitution Attack

In the blend substitution attack [34], the attacker modifies some of the points in the fuzzy vault in multiple ways. It can be done either by removing genuine or chaff points from the vault or adding some of its genuine points. If the attacker can add a sufficient number of its feature points to the vault, it can access the system without affecting the authentication of a genuine user. Thus, the secret key can be revealed to the attacker. If an attacker can replace most of the genuine points of the legitimate user with its points, the genuine user will not be able to access the vault.

In S-BIOFUSE, to attack the system via a blend substitution attack, the attacker must first decrypt the encrypted vault using a a format-preserving encryption scheme that needs key K_1 obtained from the first biometric template. Thus, encryption of vault in BIOFUSE makes blend substitution attack difficult.

3.4.4.3 Correlation Attack

In multiple vaults of the same user, genuine points correlate well, which violates unlinkability. It results in a correlation attack in which an attacker can detect genuine correlated points from multiple vaults and thus separates the chaff points. In some techniques [62], a password is used to transform the genuine points by permuting them so that correlation cannot be found in multiple vaults or chaff points are deterministic [34] which is an overhead. In S-BIOFUSE, no additional security parameter has been used to provide unlinkability.

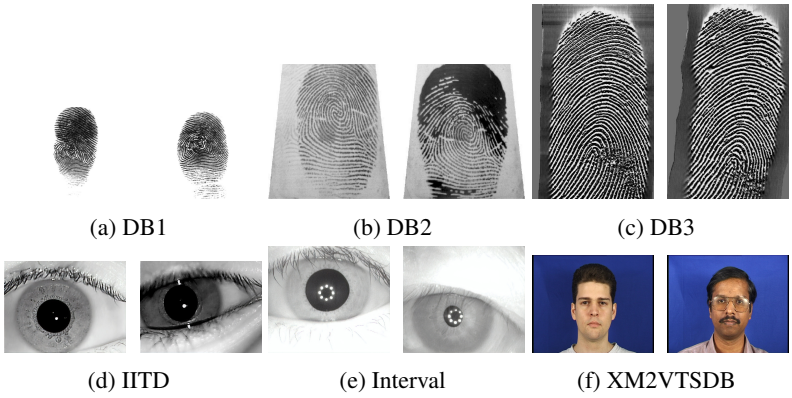
Let V_1 be the vault generated for one particular application. Using the user's biometric template, T and key K_1 values are obtained from (3.13) and (3.15) and are used to encrypt the vault V_1 through a format-preserving encryption scheme. In the case when multiple vaults $\{V_2, V_3, ...V_x\}$ of the same user are needed for multiple applications, the K_1 and T generated for each application from the same user will be entirely different for different applications due to similar but not identical biometric characteristics. Since different vaults of the same user are encrypted by different values of T and K_1, it is impossible to find a correlation between these vaults across different applications.

3.5 Experiments and Results

We perform several experiments for fingerprints, iris, and face characteristics on the publicly available databases given in Table 3.5. The example images are shown in Fig. 3.8. To create a virtual multi-modal database, we combined the samples from the two different instances considering one-to-one correspondence. We deleted the extra samples, if any. We consider 70 subjects from each combined dataset as genuine subjects and 30 subjects as impostors, with left and right samples treated as mutually independent samples. The first biometric characteristic is given as input to the fuzzy commitment scheme, whereas the second characteristic is fed to the fuzzy vault scheme.

Table 3.5: Iris and Face Database Description

Modalities	Database(s)	Subjects	Samples	Resolution
Fingerprint	FVC2002- DB1,DB3,DB3 [13]	100	8	500×500[4]
Iris	IITD [63]	448	5	320×240
Iris	CASIA-Iris-Interval[5]	337	5	640×480
Face	XM2VTSDB (CDS001) [64]	295	4	720×576



(a) DB1 (b) DB2 (c) DB3

(d) IITD (e) Interval (f) XM2VTSDB

Figure 3.8: Example images from the selected databases for S-BIOFUSE

We use open source libraries and software for iris and face feature extraction. For iriscode generation, we use OSIRIS [65] and University of Salzburg Iris Toolkit v1.0 [66]. We perform the feature extraction using the Daugman-like 1D-Log Gabor (LG) algorithm proposed by Masek [67] to generate iriscodes of size $512 \times 20 = 10240$ bits. For face features extraction, we use the FaceRecLib of the free signal and image processing toolbox Bob[6] [68, 69] to obtain a cropped 4×8 sub-image with $32 \times 2400 = 76800$ bits. Hamming distance is used to compare the face and iris templates. For the fingerprint database, we utilize the state-of-the-art commercial-off-the-shelf (COTS) feature extractor and matcher, VeriFinger [70] (Neurotechnology).

We compute the theoretical results [46] for our experiments in order to compare our recognition performance results with other start-of-the-art approaches. For the implementation of fuzzy commitment, we suggest using the BCH code. BCH codes are simple and are suitable choice [40, 41] of error correcting codes for the implementation of fuzzy extractors involved

[6] http://idiap.github.io/bob/

Table 3.6: Parameters used for Implementation

Parameters	Value (in bits)
Size of Iriscode's template	10240
Size of Face binarized template	76800
Length of keys K_1 and K_2	256
Size of tweak T used in FC scheme	64-128
Size of r used in FC scheme	128
Order of polynomial used in FV scheme	8-9

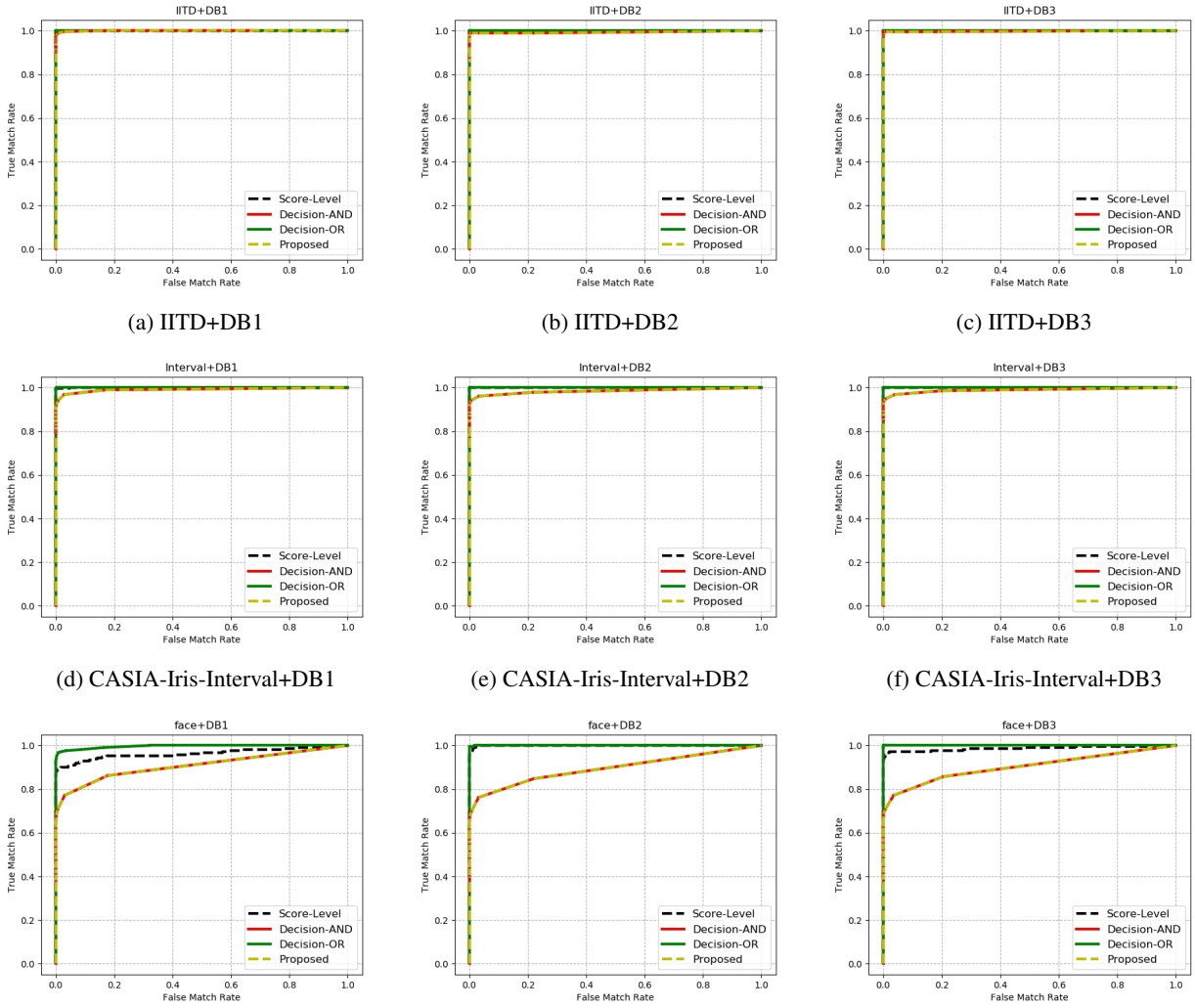
in the fuzzy commitment schemes. We select $(1023, 46, 219)$-BCH code for iriscode, which is sampled 10 times to cover 10230 bits of iriscode. Similarly, for the face template, $(1023, 46, 219)$-BCH code is sampled 75 times to generate a codeword of $76,800$ bits while ignoring the last few bits. We assume that there is some error correcting code that can correct all the errors in the biometric templates in our work.

Further, we consider the matching of fingerprints using the commercial matchers, which gives approximately the same performance as the fuzzy vault with certain parameters. Table 3.6 shows the parameters that we suggest and use for implementing the proposed algorithms. The format-preserving encryption scheme is implemented using the standard NIST source code[7] [52] with 256 bits key K_1 and 128 bits tweak T. We use SHA-256 as the hash function in the implementation. The key and tweak values are transformed from the byte array to hex strings.

3.5.1 Recognition Performance Evaluation

Biometric recognition performance measure depicts how correctly the users in a biometric system are authenticated. We plot the ROC curve (receiver operating characteristic curve) in Fig. 3.9 that demonstrates the false match rates against the true-match rates to evaluate the recognition performance. The scores are converted to a standard range before they can be fused, known as normalization. It is done using the min-max normalization [56]. The following cases are used as the underlying architecture for the various existing state-of-the-art and other related multi-biometric template protection approaches. In these cases, to compare the results with our proposed approach, we consider 2 biometric characteristics, with a fuzzy commitment and a

[7] http://www.lib4dev.in/info/capitalone/fpe/95807844



(a) IITD+DB1 (b) IITD+DB2 (c) IITD+DB3

(d) CASIA-Iris-Interval+DB1 (e) CASIA-Iris-Interval+DB2 (f) CASIA-Iris-Interval+DB3

(g) Face+DB1 (h) Face+DB2 (i) Face+DB3

Figure 3.9: Recognition performance evaluation for multi-biometric databases (a) IITD+DB1 (b) IITD+DB2 (c) IITD+DB3 (d) CASIA-Iris-Interval+DB1 (e) CASIA-Iris-Interval+DB2 (f) CASIA-Iris-Interval+DB3 (g) Face+DB1 (h) Face+DB2 (i) Face+DB3. Refer web version to interpret colors in figure legends.

fuzzy vault scheme, to get consistency in the performance measure.

- **Score-Level:** In the score level fusion approach [49, 55], the scores from individual biometric characteristics are summed up to generate a final score value (using sum-rule [56]).
- **Decision-OR:** In the decision-OR fusion [46], a Boolean OR operation is performed between the scores of individual components to generate a final decision value.
- **Decision-AND:** In the decision-AND fusion [71], a Boolean AND operation is performed between the scores of individual components to generate a final decision value.

Our proposed scheme (given as Proposed case) is compared with these three cases. The performance accuracy, along with security comparisons, is given in Table 3.7 computed from the Figure 3.9.

Proposed: Our proposed scheme is based on a decision level fusion where a Boolean AND operation is performed between the scores of individual components to generate a final decision. S-BIOFUSE ensures that the user would be authenticated when both the user's biometric characteristics are matched, given a threshold. The format-preserving encryption is used in the approach. The performance measure for the proposed scheme would be the same as the performance of the Decision-AND approach.

Table 3.7: Performance evaluation along with security comparison of various multi-biometric biocryptosystem approaches. TMR refers to true match rate and FMR refers to false match rate.

Approaches	**Database used**	**TMR at** 0.01 **FMR**	**TMR at** 0.1 **FMR**	**Security (in terms of attack complexity)**	**Approaches**	**Database used**	**TMR at** 0.01 **FMR**	**TMR at** 0.1 **FMR**	**Security (in terms of attack complexity)**
Score-level [49, 55]	IITD-DB1	0.99	0.99	$2^{\lvert K_1\rvert}+2^{\lvert K_2\rvert}$	**Decision-AND** [71]	IITD-DB1	0.98	0.99	$2^{\lvert K_1\rvert}+2^{\lvert K_2\rvert}$
	IITD-DB2	0.99	0.99			IITD-DB2	0.98	0.99	
	IITD-DB3	0.99	0.99			IITD-DB3	0.98	0.99	
	Interval-DB1	0.99	0.99			Interval-DB1	0.94	0.98	
	Interval-DB2	0.99	0.99			Interval-DB2	0.94	0.96	
	Interval-DB3	0.99	0.99			Interval-DB3	0.94	0.97	
	Face-DB1	0.89	0.92			Face-DB1	0.71	0.81	
	Face-DB2	0.97	0.99			Face-DB2	0.70	0.79	
	Face-DB3	0.95	0.97			Face-DB3	0.71	0.80	
Decision-OR [46]	IITD-DB1	0.99	0.99	$2^{\lvert K_1\rvert}+2^{\lvert K_2\rvert}$	**Proposed**	IITD-DB1	**0.98**	**0.99**	$2^{\lvert K_1\rvert}\times 2^{\lvert K_1\rvert}$
	IITD-DB2	0.99	0.99			IITD-DB2	**0.98**	**0.99**	
	IITD-DB3	0.99	0.99			IITD-DB3	**0.98**	**0.99**	
	Interval-DB1	0.99	0.99			Interval-DB1	**0.94**	**0.98**	
	Interval-DB2	0.99	0.99			Interval-DB2	**0.94**	**0.96**	
	Interval-DB3	0.99	0.99			Interval-DB3	**0.94**	**0.97**	
	Face-DB1	0.96	0.98			Face-DB1	**0.71**	**0.81**	
	Face-DB2	0.99	0.99			Face-DB2	**0.70**	**0.79**	
	Face-DB3	0.95	0.99			Face-DB3	**0.71**	**0.80**	

Following are the observations:

- Our proposed approach gives the recognition performance in terms of true match rate equal to 0.98 or 98% on the IITD-DB1 virtual database.
- The decision-level fusion with a Boolean OR operation gives the best performance (0.99)

among all the cases. It is because of the underlying OR operation, which allows the user to get authentication even if one of the two biometric characteristics is correctly matched. However, the security of such a scheme relies only on one of the biocryptosystem involved, which gives the security of $min(|K_1|, |K_2|)$ bits. Hence, it is highly insecure and is not preferred in security-sensitive scenarios. Further, we observe that the security of score-level fusion is similar to the security of decision level fusion with OR operation.

- In general, the decision-level fusion with a Boolean AND operation provides low-performance accuracy when compared to others. It is due to the AND operation involved, which is a strict criterion for authentication. However, the accuracy varies with databases and the matching approach used.

- The performance of the Decision-AND case and our proposed scheme is the same due to the underlying decision level fusion with AND operation in both cases. The difference between these two schemes lies in the security of the schemes. Our proposed scheme is implemented with the format-preserving encryption scheme, which ensures the security of $(|K_1| + |K_2|)$ bits. Whereas, the Decision-AND scheme with no format-preserving encryption in the existing state-of-the-art approaches would result in weaker security, as mentioned in the design rationale in Section 3.4.

- In Table 3.7, the virtual database with iris and fingerprint characteristics gives a high true match rate of 0.94 and above at 0.01 FMR for all the 4 cases.

- The face database combined with fingerprint gives a true match rate of about 0.70 or 71% for our proposed approach. The accuracy could undoubtedly be increased by using commercial-off-the-shelf tools.

As inferred from the related work, in the existing approaches- score level [49, 55], decision-level [46] AND/OR approaches, if one of the involved biocryptosystem is compromised, the security of the whole system is compromised. Hence, the security in all these 3 cases would be given in terms of attack complexity as $2^{|K_1|} + 2^{|K_2|}$. K_1 and K_2 are the security parameters of the two biocryptosystems, respectively. In our proposed scheme, the decision-level fusion with AND operation is implemented along with a format-preserving encryption scheme, which

makes the attack complexity equal to $2^{|K_1|} \times 2^{|K_1|}$ which is equivalent to the desired security bound $2^{|K_1|} \times 2^{|K_2|}$ in the case when $|K_1| = |K_2|$. Thus, the proposed scheme implemented with decision-level fusion with AND operation, combined with format-preserving encryption, provides the highest security level with significantly good performance accuracy.

3.6 Conclusions and Future Work

We proposed a generic, biocryptosystem level fusion framework known as BIOFUSE to design a multi-biometric cryptosystem that protects multiple biometric templates of a user. The two most popular biocryptosystems- fuzzy commitment and fuzzy vault are fused at the biocryptosystem level using a cryptographic primitive known as format-preserving encryption scheme. To get unauthorized access to the system, an attacker needs to impersonate all the input multi-biometric templates simultaneously, which is highly improbable. No additional security parameter is used for the construction of BIOFUSE. On comparing our proposed scheme's recognition performance with existing multi-biometric cryptosystems, we observe a 0.98 true match rate at 0.01 false match rate on a virtual IITD-DB1 database. We thoroughly analyze the security of all the constructions of BIOFUSE. Even though the security provided by all the constructions is comparable, we deduce that only one construction named as S-BIOFUSE (S_3) is the most reliable and secure. S-BIOFUSE achieves the security bound $\approx 2^{2\times|K_1|}$ which is similar to the desired security level ($2^{|K_1|+|K_2|}$) for a system with 2 biocryptosystems, each with key K_1 and K_2 respectively. BIOFUSE is not limited to any particular biometric characteristics or biocryptosystem and can be scaled accordingly for multiple application scenarios. S-BIOFUSE mitigates the significant attacks on existing fuzzy vault scheme such as blend substitution attack, brute force attack, key inversion attack, and correlation attack.

While our work's main aim is to provide a thorough analysis of security provided by multi-biometric cryptosystems, the experimental results of our proposed scheme on various multi-biometric databases show significant comparable recognition performance accuracy compared to the existing multi-biometric cryptosystems. It shows that S-BIOFUSE provides high security along with good accuracy and reliability to the biometric systems.

Some other directions extend this research work, which we have explored in the subsequent work.

First – is it possible to increase the recognition performance of the system to a maximum level? Ideally, the recognition performance should be equivalent to the unprotected biometric system's (baseline) performance.

Further, there is a question of the improved and efficient implementation of error correcting codes for iris and face databases to increase the overall performance of the underlying biocryptosystems. We intend to consider this question in the following work.

Chapter 4

Cancelable Multi-biometric Approach using Fuzzy Extractor and Novel Bit-wise Encryption

Multi-biometric authentication systems involve the fusion of two or more biometric instances, where fusion can be performed on feature level [6, 54, 49, 72], decision level [7] or score level [73, 55]. We propose a multi-biometric fusion framework- BIOFUSE, in Chapter 3 where the two popular biometric cryptosystems- fuzzy commitment scheme and fuzzy vault scheme are combined using the format preserving encryption scheme. Another example with a naive multi-biometric fusion approach using multi-biocryptosystems could be to implement two fuzzy extractors, denoted as FE_1 and FE_2 for the two input biometric templates B_1 and B_2 as shown in Figure 4.1a. Here, the notation fuzzy extractor emphasizes the generation of the secret key from the biometric template. It performs a similar function as that of a fuzzy commitment scheme along with the key generation process. The major limitations of such approaches with multi-biocryptosystems are as follows.

- The fuzzy extractors require encoding and decoding of the error correcting codewords, which usually takes large computations [74] due to the complex operations involved. The approach shown in Figure 4.1a with two fuzzy extractors would require twice of such

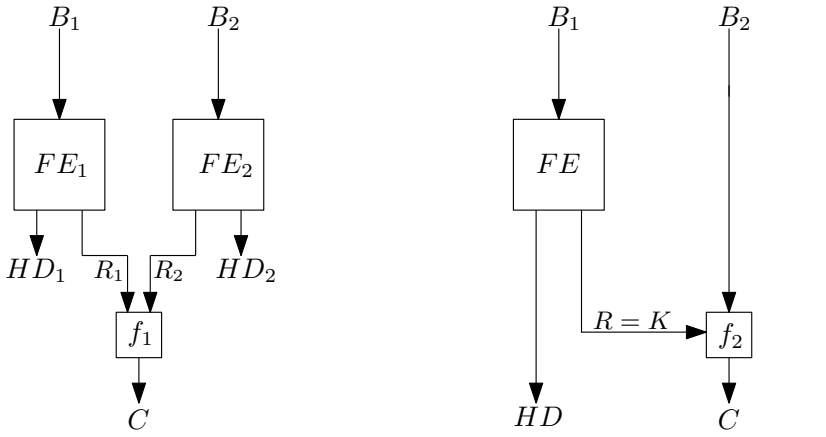


Figure 4.1: Multi-biometric template protection approaches to generate the protected template C. K denotes the secret key. **a. Use of two fuzzy extractors FE_1 and FE_2:** On input as biometric template B_i, FE_i generates helper data HD_i and a secret, random string R_i where $i = 1, 2$. f_1 denotes concatenation or XOR operation. **b. Use of a single FE:** Combines FE with a transformation function f_2 (proposed bit-wise encryption) to generate $R = K$ and HD.

complex computations.

- The helper data generated as one of the outputs of fuzzy extractor FE often leaks some linkability information about B [17]. Reusable fuzzy extractors are proposed [75, 76] to mitigate this issue. However, they require huge storage. For example, a biometric template of 1024 bits would take about 1 Terabytes of storage space to handle 25% bit errors [77]; in our case, iriscode is of 10240 bits. The implementation of two reusable fuzzy extractors with almost double storage requirements (compared to one reusable fuzzy extractor) would be impractical in real-life scenarios where a huge dataset is deployed.

- If one of the biometric characteristics possess a large number of bit errors, for example, the binarized face or fingerprint template has more bit errors than the iriscode [9], it would be difficult for the fuzzy extractor to correct all the bit errors because of the limit on error correction capabilities [1].

- Even in the scheme where a fuzzy commitment and fuzzy vault are combined, both the schemes use error correcting codes to correct the bit errors present in the biometric templates. In such a scenario, if many bit errors are present in both the biometric templates, it would be difficult to get a successful authentication.

We begin with an alternative approach, as shown in Fig 4.1b. Biometric templates are used to

generate strong, uniform random strings from the fuzzy extractors; which can be treated as the cryptographic keys [1]. Such a key generated by one biometric template can transform another biometric template of the same user in a multi-biometric system. The first biometric template B_1 is used to generate a random string R using a fuzzy extractor FE. The transformation function f_2 transforms the second biometric template B_2 into a protected template C, considering $R = K$. Only one fuzzy extractor is implemented, which will require comparatively low storage and fewer computations. The second biometric template can be transformed by the function f_2 in the following ways using the key K while dealing with bit errors during the transformation, given a pre-defined threshold.

1. Directly encrypt the biometric template using block cipher modes of operation [43]. However, it needs the decryption of the protected template during verification, revealing the original template to the attacker.

2. Cancelable biometrics such as Bloom filter based templates [78, 4] provide good performance and efficiency. Still, the error rates are high due to the transformation of biometric templates into the bloom filter based structures.

3. Homomorphic Encryption [37, 8] is used to encrypt the biometric template. However, it requires enormous computational time to perform verification.

These limitations motivate us towards solving some research questions such as-

- Can we fully preserve the bit-errors present in the biometric template during the transformation of the original biometric template to generate the protected biometric template?
  - To what extent, we can preserve the trade-off between security and performance?
- Is it possible to perform the transformation without using additional security parameters such as a key or a password?

In this regard, we propose a non-invertible, bit-wise encryption scheme (that represents f_2 in Figure 4.1b). Each bit of the second biometric template, B_2 is transformed into a corresponding

one-bit output in the protected template C. The key derived from the first biometric template B_1 ensures that even if an attacker makes random queries to the oracle (client's device), the transformed template generated will be completely random since the secret key would be unique for each instance.

The cancelable template generated is unlinkable, irreversible, and renewable. We provide an in-depth security analysis of our proposed scheme in terms of irreversibility, unlinkability, and renewability. Further, the mathematical proof shows that our bit-wise encryption scheme preserves secrecy and irreversibility.

4.1 Related Work

In this section, we discuss several existing biometric and multi-biometric template protection schemes. We primarily focus on template protection schemes for iris and face.

4.1.1 Biometric Cryptosystems or Biocryptosystems

Cimato et al. [51] proposed a modular biocryptosystem where two secure sketch schemes (as individual modules) are connected such that the key generated from the first module is XORed with the second biometric template. The above approach is generalized by Fang et al. [58], where multiple biometric templates are combined in a cascaded manner while deploying the secure sketch framework [1]. The security of the external module defines the overall security in the modular-level fusion scheme. In [57], authors proposed a cancelable secure sketch where the secure sketch is applied on cancelable biometric templates. The secure sketch provides secure biometric template protection, and the cancelable biometric template prevents correlation attacks and provides renewability property. Nagar et al. [49] provided a practical implementation of a feature-level fusion framework for fuzzy vault and fuzzy commitment schemes that simultaneously protect the multiple templates of a user using a single secure sketch. It provides high security. However, the overall performance is degraded as well as it requires adapted feature extraction tools for implementation. A decision level fusion is performed on

fingerprint-based multi-biometric biocryptosystem [7]. Hash functions are used to protect each fingerprint. However, the additional key used as a security parameter needs extra security.

4.1.2 Cancelable Biometric Systems

The cancelable biometrics, particularly for iris templates, are broadly classified into the following two categories.

4.1.2.1 Non-invertible Transforms

In the non-invertible transforms, the transformations dependent on secret parameters such as a key or a password are performed. The attacker cannot recover the original biometric template from the transformed template if the transformed template is compromised. The main disadvantage in such techniques is template degradation with loss of accuracy due to noise addition and difficult template alignment. Ratha et al. [35] is the first to introduce cancelable techniques for iris templates. They proposed two simple registration-free technique called GRAY-COMBO and BIN-COMBO, where the rows of iriscode are translated and concatenated according to a secure value. In GRAY-COMBO, the rows of unwrapped iriscode are shifted circularly based on a security parameter. Then, random rows are selected in pairs and concatenated using addition or multiplication to generate a final template using another security parameter or the part of the same security parameter. A similar approach is followed by another method known as BIN-COMBO, where the concatenation operation is performed using XOR or XNOR. Both the methods are non-invertible, but the main drawback is that the effective iris area during comparisons is reduced, which affects the accuracy rate.

Bloom-filter based Cancelable Template Generation An adaptive Bloom filters based approach [4, 2, 78] is a popular example of non-invertible transformation where a hash function defined by a simple binary to integer transformation maps the columns of the biometric template to the bloom filters.

A Bloom filter conceived by Burton Howard Bloom in 1970 is defined as a representation of

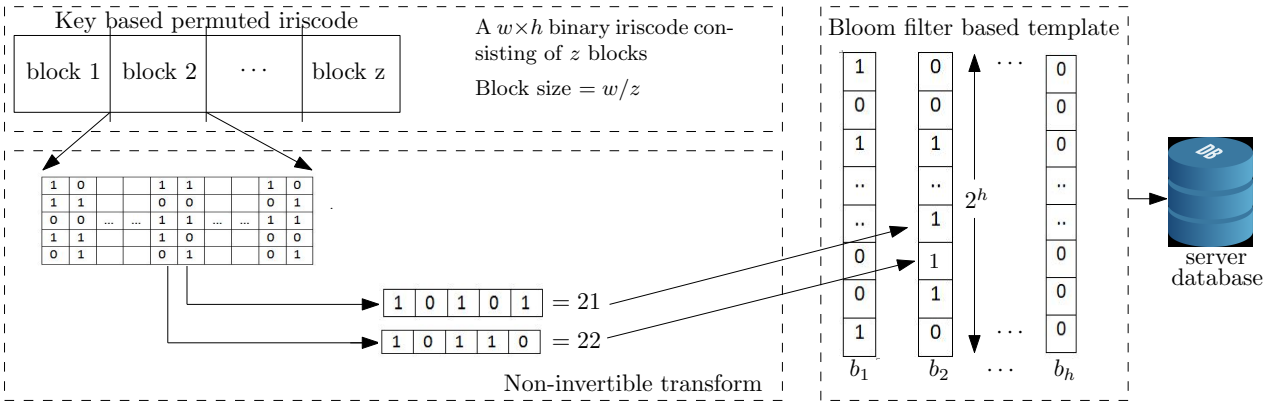


Figure 4.2: Cancelable templates generation using adaptive bloom filter technique [2]. The iriscode is permuted using a secret key using row permutation method [60] before it is mapped to the bloom filter based template.

a set $\{x_1, x_2, \cdots, x_n\}$ of n elements from a large universe using an array of m bits, initially all set to 0. It uses y independent hash functions $\{h_1, h_2, \cdots, h_y\}$ with range of size of bloom filter, such that for each element belongs to set, the output of y hash functions indicating bit positions of filter are set to 1. Then, to check if an element is present in the set, it is checked whether all bit positions corresponding to y hash functions applied on that particular element are set to 1. If not, then the element is not a member of the set. If all those bits are set to 1, we can assume that element belongs to the set. Since a small error in the form of a false rejection rate is acceptable, the bloom filter technique works well to transform a biometric template such as an iriscode or a face template. The comparisons take place by calculating the Hamming distance metrics on Bloom filter arrays.

Figure 4.2 shows the generation of cancelable iris templates using the bloom filter-based approach. A $2-$dimensional iriscode with height h and width w is divided into z blocks, each of size $l = w/z$ such that each block contains some fixed number of columns. A simple binary to integer transformation function is applied over each column in a particular block of iriscode to map each column with $h \leq H$ bits to an index in bloom filter for that block. Thus, for every block of an iriscode, a bloom filter of size 2^h is generated, resulting in a protected template of size $z \times 2^h$. Comparisons take place by simply calculating the hamming weight metrics on bloom filter arrays. Since the bloom filter is used for every block that takes input in the form of an unordered set, the scheme is alignment-free and irreversible. A secret key is used to transform the rows of iriscode [60] to generate unlinkable templates.

Bloom filter based cancelable biometric indexing using binary tree structure has been proposed [79, 80, 81] to reduce biometric comparisons workload. A multi-biometric fusion based on the Bloom filter approach is proposed [9] for face and iris templates where the biometric templates are transformed to bloom filter based templates. The fusion is then done by an OR operation between the corresponding bloom filter arrays. Notwithstanding the fact that bloom filter-based systems satisfy unlinkability, irreversibility, and renewability, the performance drop persists.

4.1.2.2 Salting

Salting adds random noise to the original biometric template. It often involves mixing in an entirely different artificial pattern to the original iriscode. The mixing pattern can be a synthetic iriscode, a random noise or a random iris-like texture. Two methods are proposed using salting approach called as GRAY-SALT and BIN-SALT [35]. In GREY-SALT, a random, artificial pattern or an iris-like texture is mixed with the original unwrapped iris image using pixel-wise addition or multiplication. A similar approach is applied to the binarised iriscode, where the mixing is done by XOR operation in the BIN-SALT approach. The main advantage of the salting method over the non-invertible transformation is that there is no shrinking or reduction of iris area in the salting approach, unlike non-invertible transformations. However, the approach faces alignment issues. BioEncoding[82] is another salting approach where consistent iriscode bits are mapped to random bit values.

4.1.3 Homomorphic Encryption Schemes for Biometric Security

Introduced by Rivest et al. in 1978 [83], it provides computation over the encrypted domain. Given two encrypted messages $E_k(m_1)$ and $E_k(m_2)$, $E_k(m_1 * m_2) = E_k(m_1) * E_k(m_2)$ where $*$ can be any operation. A somewhat homomorphic encryption scheme with a MAC-based approach is proposed [37] for secure biometric authentication where the server stores the encrypted biometric template. The client stores the decryption key to decrypt the comparison score sent by the server during authentication. The computation time required is large, which

makes the deployment impractical. Marta et al. [8] propose a multi-biometric template protection scheme in which the fusion of signature and fingerprint is done at the feature level, score level, and decision level to generate a combined template encrypted using the homomorphic encryption. The scheme is cancelable and shows good performance. However, in addition to high computation time, the secret key stored on the server is assumed to be secure. In practice, it may lead to loss of privacy in case if the secret key is leaked. A lightweight encryption scheme [84] (PassBio) is recently proposed for user-centric biometric authentication such that the comparisons are performed on the encrypted data with a threshold. The matrix multiplications involved with huge latency make the scheme considerably slower than other baseline approaches.

4.2 Models and Settings

We propose a generalized, modular framework with several components, including feature extractor, fuzzy extractor, bit-wise encryption module, and the matcher or the comparator module. Our scheme adheres to this framework.

4.2.1 System Model and Participants

The proposed system consists of a client and a server. The client contains a device capable of extracting a user's biometric data like iris, face, or fingerprint. Some examples of user devices could be Android/ iOS powered smartphones or a personal desktop. The server is an online entity (a service provider) that is honest but curious. Figure 4.3 describes the schematic model of our proposed construction. The standard feature extraction library, fuzzy extractor, and bit-wise encryption libraries are stored on the client's device. The biometric comparisons are performed on the server-side.

4.2.2 Threat Model

We focus on the following attacks to ensure user's biometric data privacy and the system's security.

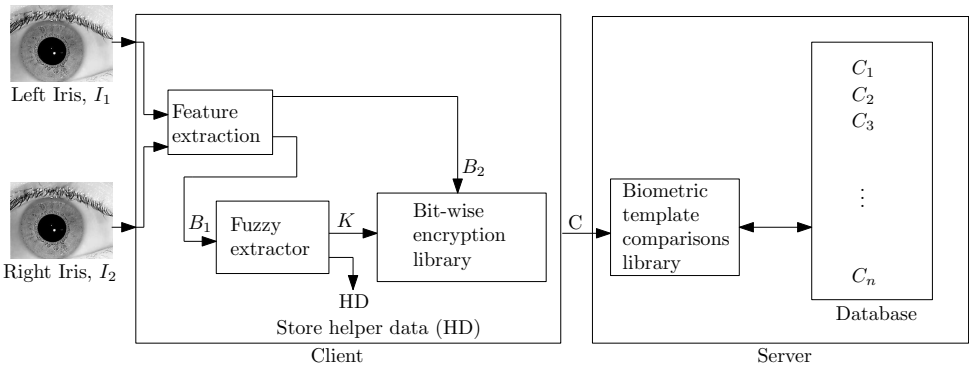


Figure 4.3: Schematic model of the proposed work.

- User-side participants can try to get the original biometric template from the protected template stored on the server. Further, they may collude arbitrarily with other system users to achieve reversibility, an attempt to break the system's security. In the proposed setup, the client device would act as a random oracle. The adversary is allowed to perform any number of queries on the client device. The user may also try to get linkability information from the information stored on the client or server-side. Further, a client's device might come under the offline attack where the adversary can read the memory, etc., of the device.
- In real-world scenarios, a server can come under two classes of attacks, passive attack (where an attacker is incapable of interacting with the server but can listen to the data routed to the server) and active attack (the attacker by any means can take control of the server and has access to the underlying database).

4.3 Proposed Work

Our scheme comprises two steps: Key generation using fuzzy extractor and bit-wise encryption to generate a cancelable template. The key generation step is different for enrolment and authentication, whereas the bit-wise encryption step is common for both phases.

4.3.1 Notations

B denotes the original biometric template represented in binary format, C denotes the protected template, (x, y) are the x and y-coordinates in the biometric template that represent the bit-wise position for a particular bit as $b_{(x,y)}$, K denotes a random, cryptographic key of length k, H denotes the hash function, HD denotes the helper data generated by the fuzzy extractor. We use the following two functions in our construction.

- **Binary p-bit mapping:** It is a p-bit binary representation of the decimal value. The binary number

$$a_{p-1}2^{p-1} + a_{p-2}2^{p-2} + \cdots + a_0$$

  is denoted as $a_{p-1}a_{p-2}\cdots a_0$ where $a_i \in \{0, 1\}$ and p is the number of digits to the left of the binary (radix) point. For example, $\text{binary}_{16}(val)$ denotes 16-bit binary representation of the value val.

- **byte-xor:**The byte-xor of a d-bit binary number $a_{d-1}a_{d-2}\cdots a_0$ where $a_i \in \{0, 1\}$ splits the whole string into individual byte strings. These strings are byte-wise XORed to output a single byte denoted as

$$(a_{d-1}a_{d-2}\cdots a_{d-8}) \oplus (a_{d-9}\cdots a_{d-16}) \ldots \oplus (a_7a_6\cdots a_0)$$

4.3.2 Key Generation

During the enrolment phase, the user provides the first biometric characteristics I_1 (left iris sample let say), from which a binary string B_1 is extracted [14]. A random error correcting codeword W_1 is generated by the system to handle the bit-errors present in the biometric template. Here $|B| = |W|$. A public helper data HD_1 is constructed as $HD_1 \longleftarrow B_1 \oplus W_1$. The cryptographic key of length k is derived by taking the hash of the codeword and is denoted as $K \longleftarrow H(W_1)$

During the authentication phase, from the similar biometric characteristics I_1', biometric template B_1' is extracted. The error correcting code W_1' is generated as $W_1' \longleftarrow B_1' \oplus HD_1$. If $\|B_1 \oplus B_1'\| \leq t$ is satisfied, W_1' is correctly decoded using t-error-correcting code to get the

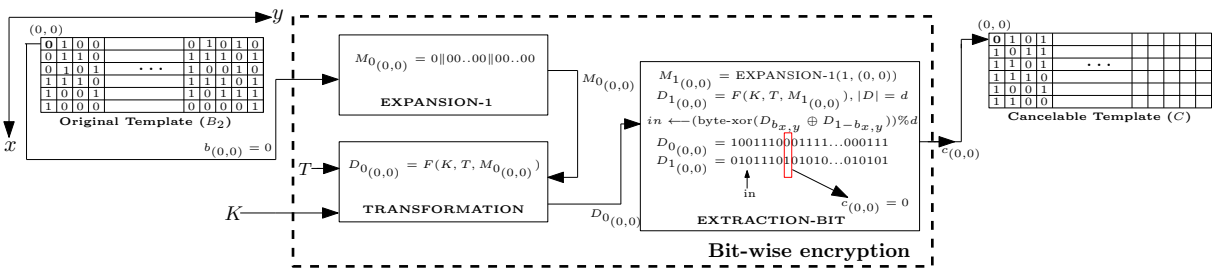


Figure 4.4: Cancelable biometric template construction using bit-wise biometric template expansion. Example shows the mapping from original to protected iriscode template for a bit position $(0, 0)$.

codeword $W_1'' = W_1$. The secret key K is recovered as $H(W_1'')$.

4.3.3 Bit-wise Encryption

Given a secret key K derived from the fuzzy extractor during enrolment phase, a bit $b_{(x,y)}$ of the biometric template at coordinates (x, y) is encrypted using the underlying cryptographic primitives such as hash function or block cipher based encryption, to generate a random bit $c_{(x,y)}$ at the corresponding coordinates (x, y). We define this mapping of each input-bit $b_{(x,y)}$ to an output-bit $c_{(x,y)}$ as bit-wise encryption for a particular coordinate (x, y). It is given as $b_{(x,y)} \longrightarrow c_{(x,y)}$ such that as an example, if for a particular coordinates (x, y),

$$\left.\begin{aligned} 1_{(x,y)} \longrightarrow 0_{(x,y)},\ \text{then}\ 0_{(x,y)} \longrightarrow 1_{(x,y)} \\ \text{and if}\ 1_{(x,y)} \longrightarrow 1_{(x,y)},\ \text{then}\ 0_{(x,y)} \longrightarrow 0_{(x,y)} \end{aligned}\right\} \tag{4.1}$$

4.3.4 Algorithmic Explanation of the Bit-wise Encryption Approach

Algorithm I and Algorithm II are defined with three functions: Expansion, Transformation, and Extraction. The Expansion function expands each input bit to some pre-defined length and can follow two different approaches denoted by Expansion-1 and Expansion-2. The expanded bits are operated with the key through the Transformation function. The Extraction function can be represented as Extraction-Bit or Extraction-XOR and returns one-bit from the Transformation function's output.

Algorithm I: Cancelable Biometric Template Construction using Bit-wise Biometric Template Expansion

Input: Original right biometric template B_2 extracted from the biometric data I_2

B_2= $\{b_{(0,0)}, b_{(0,1)}, \ldots, b_{(u-1,v-1)}\}$, tweak T, key K

Output: Protected biometric template, C= $\{c_{(0,0)}, c_{(0,1)}, \ldots, c_{(u-1,v-1)}\}$

For each bit $b_{(x,y)}$ of B:

1. **Expansion-1** ($b_{(x,y)}, (x, y)$):
2. $binary_p(x) \longleftarrow x$
3. $binary_p(y) \longleftarrow y$
4. $M_{b_{(x,y)}} \longleftarrow b_{(x,y)} \| binary_p(x) \| binary_p(y)$
5. **Transformation** ($K, M_{b_{(x,y)}}, T$):
6. $D_{b_{(x,y)}} \longleftarrow F(K, M_{b_{(x,y)}}, T)$ ▷ where F can be F_H or F_E, $|D| = d$
7. **Extraction-Bit** ($D_{b_{(x,y)}}, b_{(x,y)}, (x, y), T, K$):
8. $M_{1-b_{(x,y)}} \longleftarrow$ **Expansion-1**$((1 - b_{(x,y)}), (x, y))$
9. $D_{(1-b_{(x,y)})} \longleftarrow$ **Transformation**$(K, M_{(1-b_{(x,y)})}, T)$
10. $in \longleftarrow$ (byte-xor$(D_{b_{(x,y)}} \oplus D_{(1-b_{(x,y)})}))\%d$ ▷ selection of random index in
11. For i in (in to $(in + (d - 1))\%d$): ▷ Circular looping
12. if $D_{b_{(x,y)}}[i]! = D_{(1-b_{(x,y)})}[i]$
13. $pos \longleftarrow i$
14. break
15. $c_{(x,y)} \longleftarrow D_{b_{(x,y)}}[pos]$
16. Return $c_{(x,y)}$

4.3.4.1 Algorithm I: Cancelable biometric template construction using bit-wise biometric template expansion

Figure 4.4 shows the block diagram for the proposed construction using Algorithm I. Let an unprotected biometric template B_2 is extracted from the second biometric input I_2. The B_2 is

represented as a two dimensional matrix with u rows and v columns, where each bit is represented as $b_{(x,y)}$, where $(0 \leq x < u)$ and $(0 \leq y < v)$. T denotes a random, public parameter known as a tweak.

1. **Function EXPANSION-1 to return** $M_{b_{(x,y)}}$

   Expand the bit $b_{(x,y)}$ by concatenating the bit $b_{(x,y)}$ with the binary values of corresponding coordinates (x, y) to get an expanded bit string $M_{b_{(x,y)}}$. The binary values are obtained by using Binary p-bit mapping function.

2. **Function TRANSFORMATION to return** $D_{b_{(x,y)}}$

   For every expanded bit string $M_{b_{(x,y)}}$, apply a key-based transformation function F on it as discussed in the following cases.

   Case 1: F = Encryption function, F_E: Block cipher modes of operation using a suitable mode [43] is used with the help of a key K. An arbitrary length tweak is concatenated with the expanded bit-string message bit $M_{b_{(x,y)}}$. The output of transformation is given as $D_{b_{(x,y)}}$ of length d as

   $$D_{b_{(x,y)}} \longleftarrow F_E(K, (M_{b_{(x,y)}} \| T))$$

   Case 2: F = keyed-hash, F_H: Transformation function is the keyed-hash which involves a cryptographic hash function and the key K with the output given as

   $$D_{b_{(x,y)}} \longleftarrow F_H(K, M_{b_{(x,y)}} \| T)$$

3. **Function EXTRACTION-BIT to return** $c_{(x,y)}$

   Perform EXPANSION-1 and TRANSFORMATION functions on the complement bit $(1 - b_{(x,y)})$ to derive a corresponding transformed bit string $D_{(1-b_{(x,y)})}$ of length d. Select a random index. There are two ways to get the index value:

   - the first position of bit string is taken as index, i.e. $in \longleftarrow 0$
   - A random index position given as

   $$in \longleftarrow (\text{byte-xor}(D_{b_{(x,y)}} \oplus D_{(1-b_{(x,y)})}))\%d \tag{4.2}$$

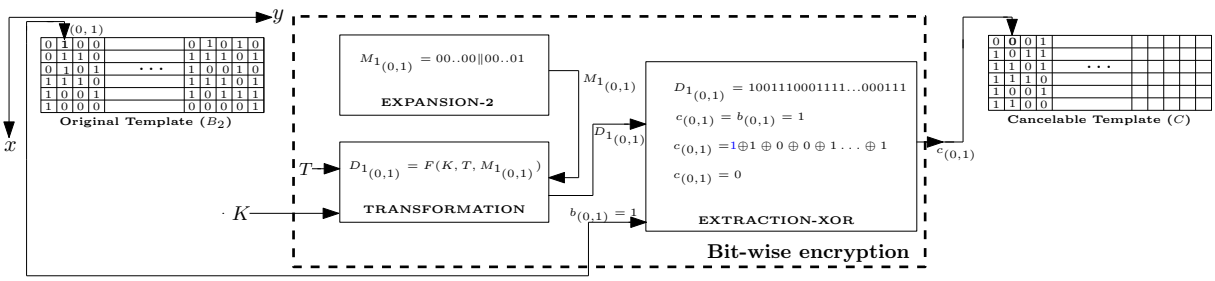


Figure 4.5: Cancelable biometric template construction using bit-wise XOR operation. Example shows the mapping from original to protected iriscode template for a bit position $(0, 1)$.

Starting from the selected index in, perform a circular lookup in both the transformed bit strings obtained from $b_{(x,y)}$ and $(1 - b_{(x,y)})$ to find the first position pos with dissimilar bits such that $D_{b_{(x,y)}}[i]! = D_{(1-b_{(x,y)})}[i]$. The corresponding bit $D_{b_{(x,y)}}[pos]$ at the particular position pos in the string $D_{b_{(x,y)}}$ represents the one-bit output $c_{(x,y)}$ at coordinates (x, y) in the protected biometric template C.

4.3.4.2 Algorithm II: Cancelable biometric template construction using bit-wise XOR operation

Figure 4.5 shows the block diagram for the proposed construction using Algorithm II. Considering the same format of the biometric template as in Algorithm I, the Algorithm II consists of the following functions:

1. **Function EXPANSION-2 to return** $M_{b_{(x,y)}}$

   Concatenate the binary values of corresponding coordinates (x, y) for a particular bit $b_{(x,y)}$ to get an expanded bit string $M_{b_{(x,y)}}$

2. **Function TRANSFORMATION to return** $D_{b_{(x,y)}}$

   The transformation is performed in the similar way as described in Algorithm I to output a transformed intermediate bit string $D_{b_{(x,y)}}$ of length d.

3. **Function EXTRACTION-XOR to return** $c_{(x,y)}$

   - The transformed bit string $D_{(x,y)}$ of length d is represented as

$$D_{b_{(x,y)}} = D_{b_{(x,y)_0}} D_{b_{(x,y)_1}} \ldots D_{b_{(x,y)_{(d-1)}}}$$

- It is bit-wise XORed with the input bit $b_{(x,y)}$ to obtain the corresponding one-bit output $c_{(x,y)}$ in the protected template C as

$$c_{(x,y)} = b_{(x,y)} \oplus D_{b_{(x,y)_0}} \oplus D_{b_{(x,y)_1}} \oplus \ldots \oplus D_{b_{(x,y)_{(d-1)}}}$$

Algorithm II: Cancelable Biometric Template Construction using Bit-wise XOR Operation

---

Input: Original right biometric template B_2 extracted from the biometric data I_2

B_2= $\{b_{(0,0)}, b_{(0,1)}, \ldots, b_{(u-1,v-1)}\}$, tweak T, key K

Output: Protected biometric template, C= $\{c_{(0,0)}, c_{(0,1)}, \ldots, c_{(u-1,v-1)}\}$

For each bit $b_{(x,y)}$ of B:

1. **Expansion-2** $((x, y))$:
2. $binary_p(x) \longleftarrow x$
3. $binary_p(y) \longleftarrow y$
4. $M_{b_{(x,y)}} \longleftarrow binary_p(x) \| binary_p(y)$
5. **Transformation** $(K, M_{b_{(x,y)}}, T)$:
6. $D_{b_{(x,y)}} \longleftarrow F(K, M_{b_{(x,y)}}, T)$ ▷ where F can be F_H or F_E, $|D| = d$
7. **Extraction-XOR** $(b_{(x,y)}, D_{b_{(x,y)}})$:
8. $D_{b_{(x,y)}} = D_{b_{(x,y)_0}} D_{b_{(x,y)_1}} \ldots D_{b_{(x,y)_{(d-1)}}}$
9. $c_{(x,y)} \longleftarrow b_{(x,y)}$ ▷ $c_{(x,y)}$ initialized with $b_{(x,y)}$
10. For $j = 0$ to $d - 1$:
11. $\quad c_{(x,y)} \longleftarrow c_{(x,y)} \oplus D_{b_{(x,y)_j}}$
12. Return $c_{(x,y)}$

4.3.4.3 Examples for Bit-wise Encryption Algorithms: Algorithm I and Algorithm II

In Tables 4.1 and 4.2, we provide the examples depicting the working of our proposed algorithms, Algorithm I, and Algorithm II, respectively. Iriscode samples are used in the examples. During the implementation, the input biometric template (with 10239 bits) is traversed in row-major

order to give 10239 bit positions starting from 0. For easy understanding and space constraints, we took only the 0th and 5th-bit positions in both the examples while executing the algorithms' steps.

For Algorithm I in Table 4.1, the expanded message $M_{b_{(x,y)}}$ is denoted by 2 bytes where the first two bits represent the input bit (including the appended 0) and the remaining 14 bits denotes the bit position of the particular input bit. Whereas in the Table 4.2 for Algorithm II, the bit positions are denoted by 16 bits to give the 2 bytes expanded message $M_{b_{(x,y)}}$. Further, in the implementation, the tweak of length 128 bits is generated, out of which 112 bits are appended during the transformation step. From both the examples shown in tables, it can be clearly inferred that the bit-wise encryption scheme preserves one-to-one mapping (to preserve the number of bit-errors) while the output bit for a particular input bit remains unpredictable given the key is secret.

Table 4.1: Example for Algorithm I. Here, B_1 and B_1' are the two samples of the same instance, whereas B_2 denotes the sample from another instance. $b_{(x,y)}$ denotes a particular bit in the sample at coordinates (x, y). T and K denotes the tweak and key respectively. Refer Section 4.3 for details. See web version to interpret colors.

Input: $B_1 = 111111110.....100000111$,

$T_1 = 01000.....0110$, K_1

Output:$C_1 = 111101011.....010011010$

for each bit $b_{(x,y)}$ of B_1:

taking 0th position bit,

$(0, 0)$ = position 0

$b_{(x,y)} = 1_{(0,0)}$

EXPANSION-1 $(1_{(0,0)}, (0, 0))$**:**

$M_{1_{(0,0)}} \longleftarrow 0100000000000000$

TRANSFORMATION $(K_1, M_{1_{(0,0)}}, T_1)$**:**

$D_{1_{(0,0)}} \longleftarrow AES(K_1, M_{1_{(0,0)}}, T_1)$

$D_{1_{(0,0)}} = 1001000.....1010010$

EXTRACTION-BIT $(D_{1_{(0,0)}}, 1_{(0,0)}, (0, 0), T_1, K_1)$**:**

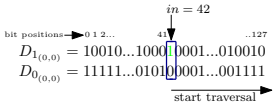
$M_{0_{(0,0)}} \longleftarrow 0000000000000000$

$D_{0_{(0,0)}}$ = AES$(K_1, M_{0_{(0,0)}}, T_1)$

$D_{0_{(0,0)}} = 1111100.....1001111$

$D_{1_{(0,0)}} \oplus D_{0_{(0,0)}} = 0110100.....0011101$

$in = 42$



$c_{(0,0)} = 1$

taking 5th position bit,

$(0, 5)$ = position 5

$b_{(x,y)} = 1_{(0,5)}$

EXPANSION-1 $(1_{(0,5)}, (0, 5))$**:**

$M_{1_{(0,5)}} \longleftarrow 0100000000000101$

TRANSFORMATION $(K_1, M_{1_{(0,5)}}, T_1)$**:**

$D_{1_{(0,5)}} \longleftarrow AES(K_1, M_{1_{(0,5)}}, T_1)$

$D_{1_{(0,5)}} = 1100011.....0111100$

EXTRACTION-BIT $(D_{1_{(0,5)}}, 1_{(0,5)}, (0, 5), T_1, K_1)$**:**

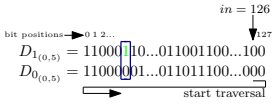
$M_{0_{(0,5)}} \longleftarrow 0100000000000101$

$D_{0_{(0,5)}}$ = AES$(K_1, M_{0_{(0,5)}}, T_1)$

$D_{0_{(0,5)}} = 1100000.....1111000$

$D_{1_{(0,5)}} \oplus D_{0_{(0,5)}} = 0000011.....1000100$

$in = 126$



$c_{(0,5)} = 1$

Similarly, we obtain $c_{(x,y)}$ for all input bits

to get $C_1 = c_{(0,0)}c_{(0,1)} \dots c_{(0,10239)}$.

Input: $B_1' = 111110000.....100000111$,

$T_1' = T_1 = 01000.....0110$, K_1

Output:$C_1' = 111100101.....010011010$

for each bit $b_{(x,y)}$ of B_1':

taking 0th position bit,

$(0, 0)$ = position 0

$b_{(x,y)} = 1_{(0,0)}$

EXPANSION-1 $(1_{(0,0)}, (0, 0))$**:**

$M_{1_{(0,0)}} \longleftarrow 0100000000000000$

TRANSFORMATION $(K_1, M_{1_{(0,0)}}, T_1)$**:**

$D_{1_{(0,0)}} \longleftarrow AES(K_1, M_{1_{(0,0)}}, T_1)$

$D_{1_{(0,0)}} = 1001000.....1010010$

EXTRACTION-BIT $(D_{1_{(0,0)}}, 1_{(0,0)}, (0, 0), T_1, K_1)$**:**

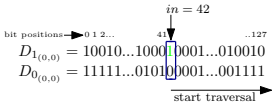
$M_{0_{(0,0)}} \longleftarrow 0000000000000000$

$D_{0_{(0,0)}}$ = AES$(K_1, M_{0_{(0,0)}}, T_1)$

$D_{0_{(0,0)}} = 1111100.....11001111$

$D_{1_{(0,0)}} \oplus D_{0_{(0,0)}} = 0110100.....0011101$

$in = 42$



$c_{(0,0)} = 1$

taking 5th position bit,

$(0, 5)$ = position 5

$b_{(x,y)} = 0_{(0,5)}$

EXPANSION-1 $(0_{(0,5)}, (0, 5))$**:**

$M_{0_{(0,5)}} \longleftarrow 0000000000000101$

TRANSFORMATION $(K_1, M_{0_{(0,5)}}, T_1)$**:**

$D_{0_{(0,5)}} \longleftarrow AES(K_1, M_{0_{(0,5)}}, T_1)$

$D_{0_{(0,5)}} = 1100000.....1111000$

EXTRACTION-BIT $(D_{0_{(0,5)}}, 0_{(0,5)}, (0, 5), T_1, K_1)$**:**

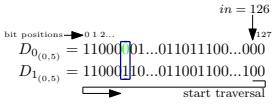
$M_{1_{(0,5)}} \longleftarrow 0000000000000101$

$D_{1_{(0,5)}}$ = AES$(K_1, M_{1_{(0,5)}}, T_1)$

$D_{1_{(0,5)}} = 1100011.....0111100$

$D_{0_{(0,5)}} \oplus D_{1_{(0,5)}} = 0000011.....1000100$

$in = 126$



$c_{(0,5)} = 0$

Similarly, we obtain $c_{(x,y)}$ for all input bits

to get $C_1' = c_{(0,0)}c_{(0,1)} \dots c_{(0,10239)}$.

Input: $B_2 = 111111110.....100000111$,

$T_2 = 01010.....0001$, K_2

Output:$C_2 = 100110111.....110010100$

for each bit $b_{(x,y)}$ of B_2:

taking 0th position bit,

$(0, 0)$ = position 0

$b_{(x,y)} = 1_{(0,0)}$

EXPANSION-1 $(1_{(0,0)}, (0, 0))$**:**

$M_{1_{(0,0)}} \longleftarrow 0100000000000000$

TRANSFORMATION $(K_2, M_{1_{(0,0)}}, T_2)$**:**

$D_{1_{(0,0)}} \longleftarrow AES(K_2, M_{1_{(0,0)}}, T_2)$

$D_{1_{(0,0)}} = 1110101.....0111010$

EXTRACTION-BIT $(D_{1_{(0,0)}}, 1_{(0,0)}, (0, 0), T_2, K_2)$**:**

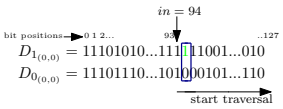
$M_{0_{(0,0)}} \longleftarrow 0000000000000000$

$D_{0_{(0,0)}}$ = AES$(K_2, M_{0_{(0,0)}}, T_2)$

$D_{0_{(0,0)}} = 1110111.....0010110$

$D_{1_{(0,0)}} \oplus D_{0_{(0,0)}} = 0000010.....0101100$

$in = 94$



$c_{(0,0)} = 1$

taking 5th position bit,

$(0, 5)$ = position 5

$b_{(x,y)} = 1_{(0,5)}$

EXPANSION-1 $(1_{(0,5)}, (0, 5))$**:**

$M_{1_{(0,5)}} \longleftarrow 0100000000000101$

TRANSFORMATION $(K_2, M_{1_{(0,5)}}, T_2)$**:**

$D_{1_{(0,5)}} \longleftarrow AES(K_1, M_{1_{(0,5)}}, T_1)$

$D_{1_{(0,5)}} = 0110101.....0110000$

EXTRACTION-BIT $(D_{1_{(0,5)}}, 1_{(0,5)}, (0, 5), T_2, K_2)$**:**

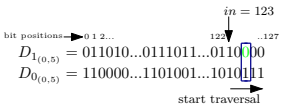
$M_{0_{(0,5)}} \longleftarrow 0100000000000101$

$D_{0_{(0,5)}}$ = AES$(K_2, M_{0_{(0,5)}}, T_2)$

$D_{0_{(0,5)}} = 1100001.....1010111$

$D_{1_{(0,5)}} \oplus D_{0_{(0,5)}} = 1010100.....1100111$

$in = 123$



$c_{(0,5)} = 0$

Similarly, we obtain $c_{(x,y)}$ for all input bits

to get $C_2 = c_{(0,0)}c_{(0,1)} \dots c_{(0,10239)}$.

Table 4.2: Examples for Algorithm II. Refer the caption of Table 4.1 for details. See web version to interpret colors.

Input: $B_1 = 111111110.....100000111$,	**Input:** $B_1' = 111110000.....100000111$,	**Input:** $B_2 = 111111110.....100000111$,
$T_1 = 01000.....0110$, K_1	$T_1' = T_1 = 01000.....0110$, K_1	$T_2 = 01010.....0001$, K_2
Output:$C_1 = 010111000.....011100010$	**Output:**$C_1' = 010110110.....011100010$	**Output:**$C_2 = 000110011.....110110101$
for each bit $b_{(x,y)}$ of B_1:	for each bit $b_{(x,y)}$ of B_1':	for each bit $b_{(x,y)}$ of B_2:
taking 0th position bit,	taking 0th position bit,	taking 0th position bit,
$(0,0)$ = position 0	$(0,0)$ = position 0	$(0,0)$ = position 0
$b_{(x,y)} = 1_{(0,0)}$	$b_{(x,y)} = 1_{(0,0)}$	$b_{(x,y)} = 1_{(0,0)}$
EXPANSION-2 $((0,0))$**:**	**EXPANSION-2** $((0,0))$**:**	**EXPANSION-2** $((0,0))$**:**
$M_{1_{(0,0)}} \longleftarrow 0000000000000000$	$M_{1_{(0,0)}} \longleftarrow 0000000000000000$	$M_{1_{(0,0)}} \longleftarrow 0000000000000000$
TRANSFORMATION $(K_1, M_{1_{(0,0)}}, T_1)$**:**	**TRANSFORMATION** $(K_1, M_{1_{(0,0)}}, T_1)$**:**	**TRANSFORMATION** $(K_2, M_{1_{(0,0)}}, T_2)$**:**
$D_{1_{(0,0)}} \longleftarrow AES(K_1, M_{1_{(0,0)}}, T_1)$	$D_{1_{(0,0)}} \longleftarrow AES(K_1, M_{1_{(0,0)}}, T_1)$	$D_{1_{(0,0)}} \longleftarrow AES(K_2, M_{1_{(0,0)}}, T_2)$
$D_{1_{(0,0)}} = 1111100.....1001111$	$D_{1_{(0,0)}} = 1111100.....1001111$	$D_{1_{(0,0)}} = 1110111.....0010110$
EXTRACTION-XOR $(1_{(0,0)}, D_{1_{(0,0)}})$**:**	**EXTRACTION-XOR** $(1_{(0,0)}, D_{1_{(0,0)}})$**:**	**EXTRACTION-XOR** $(1_{(0,0)}, D_{1_{(0,0)}})$**:**
XORing together each bit in $D_{1_{(0,0)}}$ to get	XORing together each bit in $D_{1_{(0,0)}}$ to get	XORing together each bit in $D_{1_{(0,0)}}$ to get
$1 \oplus 1 \oplus \ldots \oplus 1 \oplus 1 \oplus 1 = 1$	$1 \oplus 1 \oplus \ldots \oplus 1 \oplus 1 \oplus 1 = 1$	$1 \oplus 1 \oplus \ldots \oplus 1 \oplus 1 \oplus 0 = 1$
$c_{(0,0)} = 1 \oplus 1$	$c_{(0,0)} = 1 \oplus 1$	$c_{(0,0)} = 1 \oplus 1$
$c_{(0,0)} = 0$	$c_{(0,0)} = 0$	$c_{(0,0)} = 0$
taking 5th position bit,	taking 5th position bit,	taking 5th position bit,
$(0,5)$ = position 5	$(0,5)$ = position 5	$(0,5)$ = position 5
$b_{(x,y)} = 1_{(0,5)}$	$b_{(x,y)} = 0_{(0,5)}$	$b_{(x,y)} = 1_{(0,5)}$
EXPANSION-2 $((0,5))$**:**	**EXPANSION-2** $((0,5))$**:**	**EXPANSION-2** $((0,5))$**:**
$M_{1_{(0,5)}} \longleftarrow 0000000000000101$	$M_{0_{(0,5)}} \longleftarrow 0000000000000101$	$M_{1_{(0,5)}} \longleftarrow 0000000000000101$
TRANSFORMATION $(K_1, M_{1_{(0,5)}}, T_1)$**:**	**TRANSFORMATION** $(K_1, M_{0_{(0,5)}}, T_1)$**:**	**TRANSFORMATION** $(K_2, M_{1_{(0,5)}}, T_2)$**:**
$D_{1_{(0,5)}} \longleftarrow AES(K_1, M_{1_{(0,5)}}, T_1)$	$D_{0_{(0,5)}} \longleftarrow AES(K_1, M_{0_{(0,5)}}, T_1)$	$D_{1_{(0,5)}} \longleftarrow AES(K_2, M_{1_{(0,5)}}, T_2)$
$D_{1_{(0,5)}} = 1100000.....1111000$	$D_{0_{(0,5)}} = 1100000.....1111000$	$D_{1_{(0,5)}} = 1100001.....1010111$
EXTRACTION-XOR $(1_{(0,5)}, D_{1_{(0,5)}})$**:**	**EXTRACTION-XOR** $(0_{(0,5)}, D_{0_{(0,5)}})$**:**	**EXTRACTION-XOR** $(1_{(0,5)}, D_{1_{(0,5)}})$**:**
XORing together each bit in $D_{1_{(0,5)}}$ to get	XORing together each bit in $D_{0_{(0,5)}}$ to get	XORing together each bit in $D_{1_{(0,5)}}$ to get
$1 \oplus 1 \oplus \ldots \oplus 0 \oplus 0 \oplus 0 = 0$	$1 \oplus 1 \oplus \ldots \oplus 0 \oplus 0 \oplus 0 = 0$	$1 \oplus 1 \oplus \ldots \oplus 1 \oplus 1 \oplus 1 = 1$
$c_{(0,5)} = 1 \oplus 0$	$c_{(0,5)} = 0 \oplus 0$	$c_{(0,5)} = 1 \oplus 1$
$c_{(0,5)} = 1$	$c_{(0,5)} = 0$	$c_{(0,5)} = 0$
Similarly, we obtain $c_{(x,y)}$ for all input bits to get $C_1 = c_{(0,0)}c_{(0,1)} \ldots c_{(0,10239)}$.	Similarly, we obtain $c_{(x,y)}$ for all input bits to get $C_1' = c_{(0,0)}c_{(0,1)} \ldots c_{(0,10239)}$.	Similarly, we obtain $c_{(x,y)}$ for all input bits to get $C_2 = c_{(0,0)}c_{(0,1)} \ldots c_{(0,10239)}$.

4.4 Design Rationale

In this section, we discuss the significance of functions and parameters considered for the design of our proposed construction.

4.4.1 Role of Bit-wise Encryption:

Each bit $b_{(x,y)}$ of original template is mapped to the corresponding bit $c_{(x,y)}$ in the protected template, i.e.,$b_{(x,y)} \longrightarrow c_{(x,y)}$. The bit-wise encryption ensures that for each input bit, the probability of guessing the corresponding protected bit is no more than 2^{-1}. Further, any error in the original template reflects only in the corresponding bit in the protected template. Therefore, unlike the existing cancelable schemes as discussed in Section 4.1, the number of bit errors remains the same in both the original and the protected template.

4.4.2 Role of Expansion using Coordinates (x, y)

It ensures that each input bit is mapped to a corresponding output bit at the same bit positions, thus binds both the input and the output bits. Further, in Expansion-1, the concatenation of coordinates x and y helps to distinguish each input corresponding to a bit $b_{(x,y)}$ from others. Hence it provides different inputs to the Transformation function F resulting in a different output $D_{b_{(x,y)}}$ for each input bit $b_{(x,y)}$. In the EXTRACTION-XOR step (Algorithm II), $b_{(x,y)}$ is XORed to the output of the transformation step, hence the use of $b_{(x,y)}$ as input in the Expansion-2 would not add any extra security to the scheme.

If expansion is not done, then input to the Transformation function could be 1 or 0, which would lead to only 2 different outputs, either corresponding to 1 or 0 after the transformation step. In such a case, with the knowledge of transformation output corresponding to even a single bit of input biometric template, the attacker can reveal almost the whole input biometric template.

4.4.3 Role of Transformation Function F

The Transformation function F can be an encryption function such as any block cipher with the modes of operation or a keyed-hash such as HMAC. The properties of underlying cryptographic primitives ensure the secure transformation of the expanded message $M_{b_{(x,y)}}$ to generate a random transformed string $D_{b_{(x,y)}}$ of length d. Randomization makes it difficult to predict the input from $D_{b_{(x,y)}}$.

4.4.4 Role of Biometric Derived key K

Case 1: A single key is generated by the system for all the users (contradictory case): Assume that the attacker has access to the cancelable template stored on the database. In such a scenario, the targeted-user attacker can attack by querying the oracle (client's device) with a random query biometric template B_i' for the user i. The attacker can also attempt to get authenticated by impersonating a genuine user with its biometric data (plaintext) using zero-effort attack [85, 86]. It can then obtain the corresponding random template C_i' in the protected domain using the common system's key. Since the bit-wise encryption is a deterministic mapping of input bit to output bit, and the key is the same for all the users, the attacker can get the original (correct) biometric template B_i from another known protected template C_i by mapping C_i accessed from the database to B_i according to the known mapping (C_i', B_i').

Case 2 (Proposed approach)- A biometric-derived key generated from the fuzzy extractor for a particular instance: In this case, the key is unique for each instance and is dependent on the biometric data of that instance. Thus the attacker cannot query the client's device as an oracle by sending multiple random query templates. This is because the random biometric template will generate a random (incorrect) key. It results in the wrong transformation by a bit-wise encryption approach.

Table 4.3: Iris and Face Database Description

Modalities	Database(s)	Subjects	Samples	Resolution
Iris	IITD [63]	448	5	320×240
Iris	CASIA-Iris-Thousand[1]	2000	10	640×480
Iris	CASIA-Iris-Interval[0]	337	5	640×480
Face	XM2VTSDB (CDS001) [64]	295	4	720×576

Table 4.4: Virtual Multi-modal Databases with Genuine and Impostor Distribution. Face denotes XM2VTSDB database.

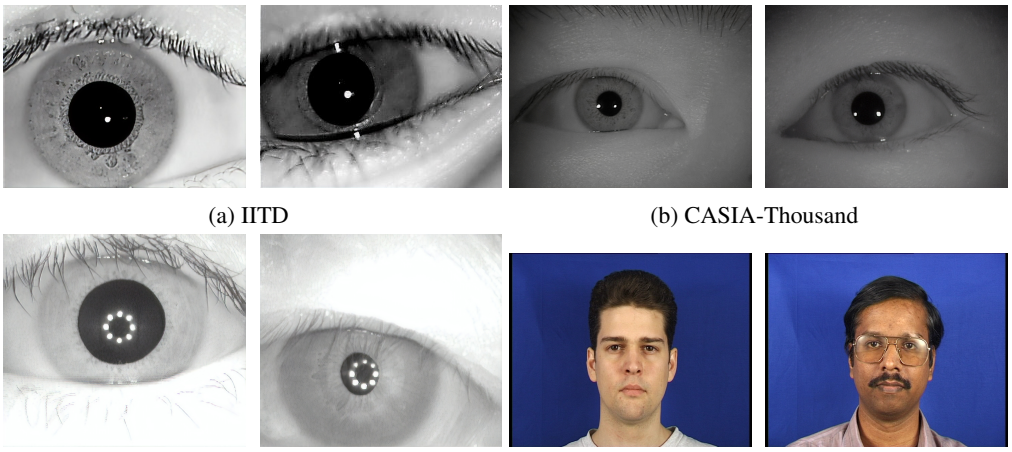
Database	Genuine	Impostor
IITD-CASIA Left Iris & Right Iris	128	2079
CASIA-Iris-Interval Left Iris & Right Iris	128	36
IITD-CASIA & CASIA-Iris-Interval	256	81
IITD-CASIA & Face	256	39
CASIA-Iris-Interval & Face	256	39

4.4.5 Role of Tweak T

Tweak T is public data of an arbitrary length. The tweak is used for padding in the input bits before the input is transformed using the Transformation function F for Algorithm I and Algorithm II. The tweak can or cannot be a unique value. The use of tweak in our system does not reveal any information about the biometric data of a user.

4.4.6 Role of Extraction Function:

The Extraction function returns one-bit from the d-bit output of the transformation function F. It is designed to provide one-wayness, i.e., it should not be possible to get the d-bits output of function F from the Extraction function's one-bit output. Even if an attacker gets access to the key K involved during the transformation, it is computationally infeasible to decrypt the protected template's bits to get the original iriscode. This is due to the truncation of one-bit from the whole d-bit output of F.



(a) IITD (b) CASIA-Thousand

(c) CASIA-Iris-Interval (d) XM2VTSDB (CDS001)

Figure 4.6: Example images from the selected databases for cancelable template generation

4.4.7 Role of Selecting a Disagreeing Bit Position posn Algorithm I

The random string $D_{b_{(x,y)}}$ is traversed starting from index in to get a position pos with dissimilar bits. It ensures that if for a particular coordinate (x, y), bit $'1'$ maps to $'0'$ in the cancelable template, then bit $'0'$ will surely map to bit $'1'$ for the same instance and vice versa as shown in (4.1). This preserves the number of errors in the unprotected and protected (cancelable) domain. If the condition is unsatisfied, then the bits $'1'$ and $'0'$ taken as input from two different samples of the same instance at coordinates (x, y) might both map to either bit $'1'$ or bit $'0'$ in the cancelable template, increasing the bit-errors during authentication.

4.4.8 Role of Selecting Random Index inor Extraction-bit Function in Algorithm I

From the empirical data, we observed that for index 0, most of the positions pos that are selected with dissimilar bits are from the range $0 - 7$ out of the total $d = 256$ bit positions. Whereas, for the random index value (generated using (4.2)), the values of dissimilar positions are uniformly distributed across the whole 256 bits output range, which makes guessing of a bit $c_{(x,y)}$ highly unpredictable. Thus, selecting a random index over-index value of 0 is preferable.

4.5 Experiments and Performance Analysis

We evaluate the iris and face experiments on the publicly available databases given in Table 4.3. The example images are shown in Figure 4.6. We consider the left and right instances as mutually independent subjects. To create a multi-modal database, we combined the samples from the two different instances with one-to-one correspondence while deleting the extra samples. We combine the IITD database with the CASIA-Iris-Thousand database, denoted as IITD-CASIA database in which all the (2000×10) samples from the CASIA-Iris-Thousand database are taken as impostors. The distribution of genuine and impostor samples for multi-modal datasets is shown in Table 4.4. We have not considered the case where both the input templates are from face characteristics. Implementing fuzzy extractor (with error correcting code) for face template as input is inefficient due to its large size $(76,800$ bits).

For Baseline-A, the existing template protection schemes and our proposed algorithms, we use open source libraries and software for feature extraction. For iriscode generation, we use OSIRIS [65] and University of Salzburg Iris Toolkit v1.0 [66]. Feature extraction is performed with Daugman-like 1D-Log Gabor (LG) Algorithm proposed by Masek [67] to generate iriscodes of size $512 \times 20 = 10240$ bits. 16 bloom filters are constructed considering best parameters [79] with height of iriscode = 10, the width of each block of iriscode = 32. For face features extraction based on local Gabor pattern histogram sequences, we use the FaceRecLib of the free signal and image processing toolbox Bob[2] [68, 69]. We cropped each image to obtain central 4×8 sub-image with $32 \times 2400 = 76800$ bits. 960 bloom filters are generated for binarized face templates, each of size 16 bits as described in [4, 9]. All the comparisons for the face and iris templates are made using the Hamming distance metric between two binarized templates. Circular bit-shifts are often done to align two iriscodes [14], to compensate for the binary misalignment during comparisons. Hamming distance is computed for each shift position (i.e., relative tilt angle, in our case, ± 4 bits), and the minimum Hamming distance is taken as the final comparison score. For Baseline-B, we utilize the state-of-the-art commercial-off-the-shelf (COTS) matchers that provide the comparison scores. We use VeriEye [87] (Neurotechnology) for iris and VeriLook [88] (Neurotechnology) for the face feature extraction and comparisons.

[2] http://idiap.github.io/bob/

Table 4.5: Parameters used for Implementation

Parameters	Value (in bits)
Size of Iriscode's template	10240
Size of Face binarized template	76800
Length k of key K	128
Size of fixed-length tweak T	128
Size of arbitrary-length tweak T	64-128

4.5.1 Parameters for Transformation Function

This section briefly discusses the parameters used for transforming the original biometric templates to the protected cancelable templates.

We use BCH code for the implementation of the fuzzy extractor. It has been observed from the literature [31, 32] that BCH codes are simple and are the suitable choice of error correcting codes for the implementation of fuzzy extractors. Considering approximately 20% errors in the iriscode, for iriscode of length 10240 bits, we selected $(1023, 46, 219)$-BCH Code. We sampled the $(1023, 46, 219)$-BCH Code 10 times to cover 10230 bits of iriscode, ignoring the last few bits. The number of bit-errors in binarized face templates is high; however, with an efficient error correcting code, the face templates can be used.

Table 4.5 shows the parameters that we suggest and use for implementing the proposed algorithms. To impose security while implementing the proposed algorithms, SHA-2 or SHA-3 are preferred hash functions. Besides, for the transformation function instantiated with encryption using block ciphers, we choose AES-CBC mode [43]. An IV is required to encrypt the plaintext using AES-CBC mode. To generate an IV required during encryption, we use the standard recommended method [43]. Apart from encryption using a block cipher, we can use a key-based hash function known as hash-based message authentication code (HMAC) that provides one-way transformation. The motivation behind choosing AES-CBC and HMAC is their practical deployment on large scale systems. Both these cryptographic primitives are well analyzed, secure, and efficient in terms of performance.

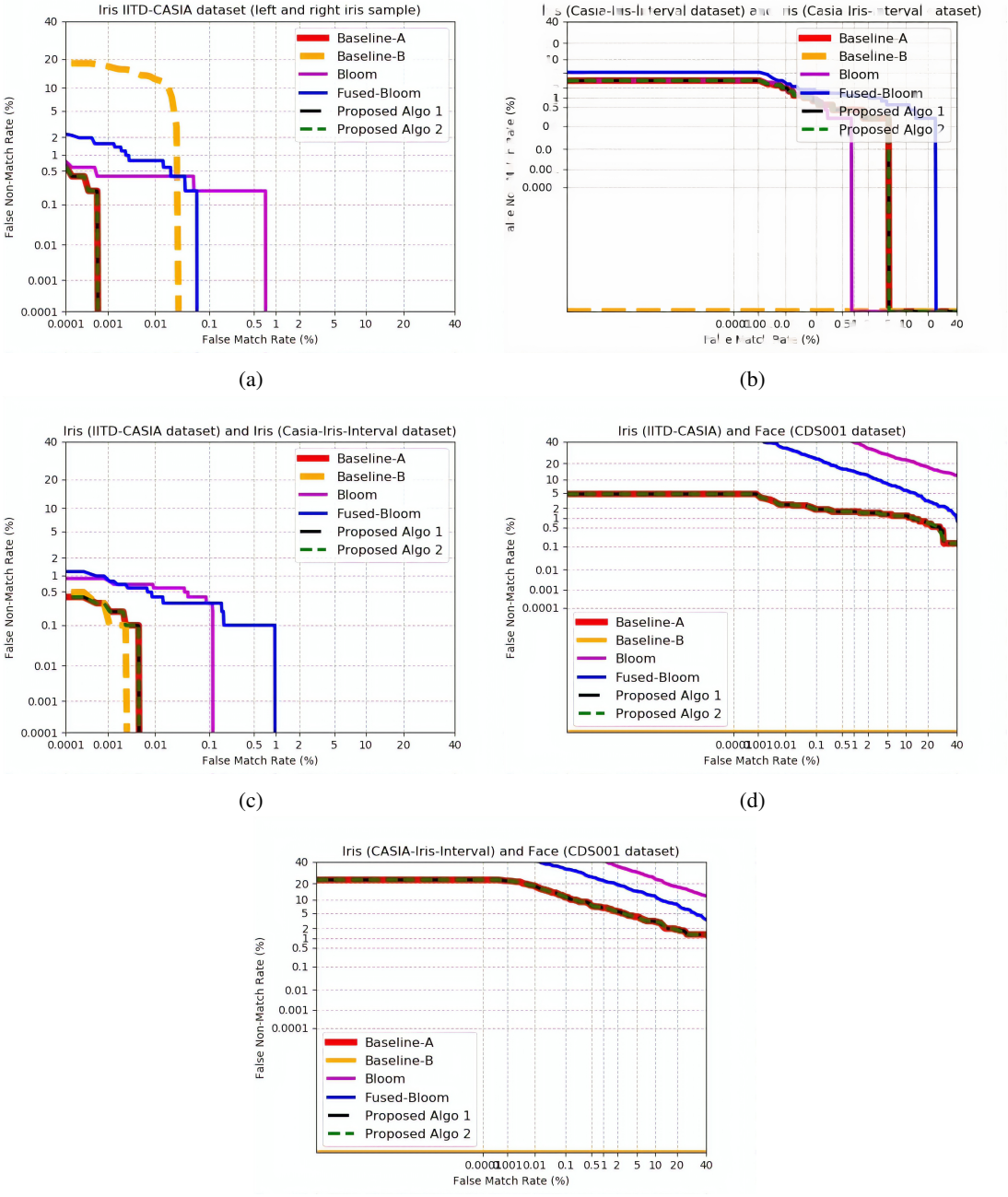


Figure 4.7: Recognition performance evaluation for multi-biometric databases (a) IITD-CASIA (left & right samples) (b) CASIA-Iris-Interval (left & right samples) (c) IITD-CASIA & CASIA-Iris-Interval (d) IITD-CASIA & Face (e) CASIA-Iris-Interval & Face. Here, Baseline-A (open source) and Baseline-B (COTS), both represent the unprotected template approaches, bloom represents combining fuzzy extractor with bloom filter based approach [4, 2], fused-bloom represents the bloom filter based fusion approach [9]. Proposed algo 1 and proposed algo 2 represent combining fuzzy extractor with proposed Algorithm I and Algorithm II. (Refer web version to interpret colors in figure legends)

4.5.2 Recognition Performance Evaluation

We plot the DET curve (Detection Error Trade-off) that demonstrates the false match rates (FMR) against the false non-match rates (FNMR) to evaluate the recognition performance. While considering various existing biometric template protection approaches, the following cases are constructed depending on the role of f_2 (shown in Figure 4.1):

- Baseline-A- The fuzzy extractor and f_2 have no role. The comparisons for both the input templates are made using Hamming distance metric [14].
- Baseline-B- COTS systems used for face and iris' features extraction and comparisons.
- Bloom filter-based approach [4]- The first template is given to the fuzzy extractor to generate the key K with the comparisons done using the Hamming distance metric. f_2 transforms the second biometric template to bloom filter based arrays using K. The comparisons are made by matching corresponding bloom filter arrays [2] using the Hamming distance metric.
- Bloom filter based fusion approach [9]- Here, the bloom filter based templates generated from the two input biometric templates are fused using the simple OR operation. The comparisons are made on the fused bloom filter arrays. The fuzzy extractor and f_2 is not used.
- Proposed approach- The first template is given as input to the fuzzy extractor to generate the secret key. The comparisons are made using the Hamming distance metric. f_2 performs the bit-wise encryption on the second template where it is transformed by one of our proposed algorithms with comparisons done using Hamming distance metric.

Fusion of scores: In the third case, multi-biometric bloom filter based fusion approach [9], the fused bloom filter arrays are directly compared using Hamming distance metric [9] to give the final scores without any fusion required on scores. In the rest of the cases, the two individual scores are computed for each of the two input biometric templates, based on the underlying approach. The scores are first converted to a common range before they can be fused, known

Table 4.6: **Efficiency evaluation during enrolment: Time taken (in milliseconds) to generate a cancelable biometric template by f_2 (shown in Figure 4.1). It takes inputs as a second biometric template and the key derived from the fuzzy extractor using the first biometric template (iriscode always, in our experiments).** We have not added the time taken by the fuzzy extractor to derive the secret key from the first template given by user during enrolment in the table. Experimentally, the fuzzy extractor takes 4.37 milliseconds to derive a key from the iriscode template.

Biometric characteristics	**Iris**								**Face**							
Platform/ Approaches	**Desktop (Processor clock freq. (GHz)**				**Android (smartphone model)**				**Desktop (Processor clock freq. (GHz)**				**Android (smartphone model)**			
	2.1	**2.5**	**2.6**	**3.3**	**Samsung Galaxy S6**	**One Plus 6**	**Google Pixel 2**	**Samsung Note 9**	**2.1**	**2.5**	**2.6**	**3.3**	**Samsung Galaxy S6**	**One Plus 6**	**Google Pixel 2**	**Samsung Note 9**
Bloom filter based [2]	7	6	4	2	24	7	12	9	20	18	17	14	24	9	12	9
Paillier cryptosystem based [8]	105	100	95	70	510	98	780	470	1240	1100	700	488	510	330	780	470
Proposed Algorithm I	17	16	14	12	800	460	580	600	220	190	130	105	800	460	580	600
Proposed Algorithm II	9	8	7	7	678	350	524	550	190	175	110	90	687	230	524	550

as normalization. It is done using the reduction of high-scores effect (RHE) normalization [56]. The normalized scores are then fused with the sum-rule based fusion approach, i.e., for two normalized scores x_1 and x_2, considering equal weights to both, the fused score is represented as $x_1 + x_2$.

It can be observed from the DET curves plotted in Figure 4.7 that the recognition performance of both our proposed algorithms is the same as that of the Baseline-A approach, and the wholly overlapped curves can depict the same for the Baseline-A and the proposed algorithms. Further, our proposed scheme's performance outperforms the performance of the Bloom filter-based approach [4] and multi-biometric bloom filter-based fusion approach [9] at 0.01% false match rate. It can be noticed that homomorphic encryption schemes [8] generally preserves the error rate, producing a similar curve as the baseline approach. The Baseline-B (COTS systems) achieve the highest performance in most cases, though we can achieve the equivalent performance by optimizing the use of underlying feature extractor and comparison modules in our proposed algorithms.

4.5.3 Efficiency Evaluation for the Generation of Protected Template

We calculate the efficiency of our proposed work in terms of the time taken during enrolment and authentication.

Table 4.7: **Efficiency evaluation during authentication: Time taken (in milliseconds) to perform biometric authentication (specifically the time taken for verification between any two templates). It includes the time taken by f_2 to transform the template and the comparison time to compare the two biometric templates.** We have not added the time taken by fuzzy extractor to recover the secret key from the first template given by user during authentication in the table. Experimentally, the fuzzy extractor takes around 10.952 milliseconds for key recovery (by BCH codeword decoding) during authentication.

Biometric characteristics	**Iris**								**Face**							
Platform/ Approaches	**Desktop (Processor clock freq. (GHz)**				**Android (smartphone model)**				**Desktop (Processor clock freq. (GHz)**				**Android (smartphone model)**			
	2.1	**2.5**	**2.6**	**3.3**	**Samsung Galaxy S6**	**One Plus 6**	**Google Pixel 2**	**Samsung Note 9**	**2.1**	**2.5**	**2.6**	**3.3**	**Samsung Galaxy S6**	**One Plus 6**	**Google Pixel 2**	**Samsung Note 9**
Unprotected (Baseline-A)	0.042	0.040	0.031	0.027	0.243	0.224	0.221	0.213	1.609	1.365	1.600	1.597	3.141	2.994	3.119	3.064
Bloom filter based [2]	7.044	6.042	4.030	2.023	24.209	7.201	12.212	9.211	21.510	19.476	18.443	15.335	26.730	11.314	14.947	11.490
Paillier cryptosystem based [8]	2172	2142	2132	2112	2901	2241	2783	2466	2811	2744	2715	2634	3010	2811	2783	2791
Proposed Algorithm I	17.048	16.042	14.032	12.022	800.223	460.191	580.212	600.209	221.790	191.766	131.711	106.680	803.341	463.040	583.210	603.092
Proposed Algorithm II	9.043	8.042	7.031	7.022	678.213	350.201	524.211	550.220	191.712	176.680	111.600	91.581	690.203	233.003	527.129	553.010

Enrolment time: the time taken to generate a secret key using fuzzy extractor (using BCH encoder) from the first template + time taken to transform the second template using bit-wise encryption.

Authentication time: is given by the time taken to decode the secret key using fuzzy extractor from first template + time taken to transform the second template using bit-wise encryption + verification time taken to compare two templates.

Our implementation consists of two steps.

- The fuzzy extractor implementation that majorly involves BCH encoding (during enrolment) and decoding module (during authentication). We use BCH encoding and decoding modules written in C language[3]. A server-grade processor, Intel Xeon 2.10 GHz, 8 core with hyper-threading for profiling BCH encoding and decoding subroutines are used in our experiments.

- Generate a cancelable template using proposed bit-wise encryption. To compute the time taken by f_2, we profiled our Java-based implementation on desktop-grade hardware (4 cores) of varying clock frequencies, including Intel i5 (Kaby Lake, 2.1 GHz), Intel i5 (Skylake 2.5 GHz), Intel i7 (Skylake, 2.6 GHz) and Intel i5 (Skylake, 3.3 GHz). We ran our experiments on various Android-based smartphones as well. An AES-NI (cryptographic

[3] http://www.eccpage.com/bch3.c

hardware instructions) enabled processor can perform around 22952162 times *AES128* encryption in *CBC* mode. In our desktop implementation, we used Intel's AES-NI instruction set for fast *AES* computation inside a *C* library and interfaced the *C* library to our Java implementation using Java Native Interface (JNI).

In Table 4.6 and 4.7, we show a comparison of enrolment and authentication performance with 4 cases: (i) f_2 as existing Bloom filter based scheme [2], (ii) f_2 as homomorphic encryption-based scheme [8], (iii) f_2 as proposed Algorithm I and (iv) f_2 as proposed Algorithm II. All these 4 cases are protected template approaches. We also consider the baseline cases during authentication, where hamming distance is computed between the two unprotected iriscodes. The BCH code- encoding and decoding function consumes most of the time in the fuzzy extractor step. This time is the same for all the approaches (bloom filter-based, multi-biometric fusion, and our two proposed algorithms).

The following inferences can be drawn:

- During enrolment and authentication, the proposed Algorithm II defined over block cipher modes of operation on desktop architecture on all the given clock frequencies is significantly comparable to the existing Bloom filter based scheme [2] in terms of time.
- In general, it can be seen on both the desktop as well as the Android-based implementation, the Algorithm I is slower than Algorithm II; This is because, in Algorithm I, we perform two cryptographic operations for each bit, thereby increasing the computation and hence higher turnaround time.
- During authentication, the homomorphic encryption-based scheme [8] uses an additional call to the client to get the encrypted threshold, thereby adding a round trip time or RTT (we fixed it to 2 seconds). Due to this RTT, the homomorphic encryption-based scheme is comparatively slow.
- While the time taken for enrolment by homomorphic encryption-based scheme [8] on Android-based devices outperforms both the algorithms proposed, in the desktop-based architectures, both our proposed algorithms significantly outperform the homomorphic

encryption-based scheme [8]. This contrariety is due to the use of the AES-NI instruction set in desktop-based implementation. It provides a multitude increase in encryption and decryption operations. We can get the same efficiency on Android (where we use only a high-level application program interface) by implementing the hardware instructions. Further, we achieved faster implementation for encryption using block cipher modes of operation than keyed-hash functions (HMAc) since we use the AES-NI instruction set to implement encryption based on block ciphers.

- In the case when ± 4 bit-shifts are considered, the time taken to generate the protected template increases to about 2 to 4 times of the time taken by a single protected template.[4] In the case of Android, the time would be about 5 to 8 times of the single protected template creation time since Android manages memory more aggressively. For faster implementation, enrolment can be done offline with the storage of all the protected templates with bit-shifts applied for a particular instance. During authentication, the Hamming distance is computed with the protected templates stored for a user, which takes almost negligible time for all the ± 4 bit-shift cases for a particular instance.

- Although the bloom filter based scheme [2] outperforms all the other protected approaches, the error rates in the bloom filter based scheme are high as compared to our proposed algorithms. It limits the applicability of these schemes in a broader scale.

4.6 Security and Privacy Analysis

Our proposed bit-wise encryption scheme is secure in terms of secrecy and irreversibility.

Assumption: We assume that the system is robust enough to handle spoofing using a spoof detection module. Even though the face is easily prone to spoof attacks compared to iris, much research has been done on the detection of presentation attacks (PAD) in [89, 90, 15]. Though we suggest designing the system with biometric characteristics that are difficult to spoof, such as iris, which is more robust to spoofing [91]. In such a scenario, an attacker's attack complexity to

[4]Most of the data like a key, tweak, etc. remains in memory for all bit shifts.

get access to one biometric template of a genuine user is equivalent to the attack complexity to get both the biometric templates of the particular user.

Mathematical Proof of Bit-wise Encryption Scheme

We establish the following results. Our bit-wise encryption scheme S is secure and irreversible for a given input sample space $\mathcal{M} \in \{0,1\}$, a transformed output sample space $\mathcal{C} \in \{0,1\}$ and a random key space $\mathcal{K} \in \{0,1\}^{|K|}$. Following are the definitions that are used in the proofs.

- Secrecy: Given an input sample $m \in \mathcal{M}$ and output sample $c \in \mathcal{C}$, $Pr[S(m) = c] = Pr[S(m') = c], \forall\{m, m'\} \in \mathcal{M}$
- Irreversibility: Given an output sample $c \in \mathcal{C}$, $Pr[\exists m \in \mathcal{M} | S(m) = c]$ is computationally bounded by $O(2^{|K|})$ where $K \in \mathcal{K}$ is a security parameter.

Statement 1**:** Bit-wise security implies the overall security of the scheme S.

Proof. The proposed bit-wise encryption scheme follows two conditions:

- Each input bit taken from the original biometric template is mapped to its corresponding output bit in the protected template as shown in (4.1). This condition clearly ensures one-to-one mapping between the input bit and the corresponding output bit, thereby ensuring that bit errors are fully preserved in the protected biometric template.
- The output bit in the protected template corresponding to the given input bit is entirely random, i.e., the probability of guessing each output bit is no more than 2^{-1}.

A function F with a secret key K transforms a fixed-length input to a fixed-length random output. In real life, such F can be instantiated with various cryptographic primitives. Two such choices could be deterministic encryption function F_E and keyed hash function F_H. In general,

$F_E : \{0,1\}^n \times \{0,1\}^{|K|} \rightarrow \{0,1\}^n$ or

$F_H : \{0,1\}^n \times \{0,1\}^{|K|} \rightarrow \{0,1\}^\lambda$

where $|K|$ is the size of the key, and λ is the size of hash output.

Thus, from the property of transformation function (computationally bounded) given as $D_{b_{(x,y)}} \longleftarrow F(K, M_{b_{(x,y)}}, T)$, the obtained output string $D_{b_{(x,y)}}$ is random, where input $M_{b_{(x,y)}}$ is generated by considering the coordinates (x, y) at positions of each input bit $b_{(x,y)}$ (in the expansion step). Tweak is a public parameter. Hence for a user, different inputs are processed through the same secret key to provide different random outputs.

Each bit after transformation produces an output $D_{b_{(x,y)}}$ of size n or λ that is random. Further, from $D_{b_{(x,y)}}$, a random output bit $c_{(x,y)}$ is extracted (in Statement 2). Cryptographic properties of F ensure that it is computationally infeasible to find the corresponding input bit $b_{(x,y)}$, given the output $D_{b_{(x,y)}}$. Hence, bit security is achieved. □

Statement 2: The proposed bit-wise scheme S is irreversible, given that K is a secret and is unique for each user.

Proof. A random string $D_{b_{(x,y)}}$ is obtained using function F as shown in Statement 1. From this random string, an output bit $c_{(x,y)}$ corresponding to each input bit $b_{(x,y)}$ is obtained in the extraction step.

In the Extraction-Bit step (Algorithm I), a random output bit $c_{(x,y)}$ is extracted from $D_{b_{(x,y)}}$ and $D_{(1-b)_{(x,y)}}$, with the help of a random index in obtained from (4.2). Since the key K involved in underlying function F is secret, the attacker cannot obtain the input bit $b_{(x,y)}$ from the corresponding output bit $c_{(x,y)}$. Hence, the security bound can be given as $2^{|K|}$.

In the Extraction-XOR step (Algorithm II), $D_{b_{(x,y)}}$ of length $n = \lambda = d$ is random represented as

$$D_{b_{(x,y)}} = D_{b_{(x,y)_0}} D_{b_{(x,y)_1}} \dots D_{b_{(x,y)_{(d-1)}}}$$

$$\implies D_{b_{(x,y)_0}} \oplus D_{b_{(x,y)_1}} \dots \oplus D_{b_{(x,y)_{(d-1)}}} \text{ is random [from the property of XOR operation]}$$

So, given an unknown input bit $b_{(x,y)}$,

$c_{(x,y)} \longleftarrow b_{(x,y)} \oplus (D_{b_{(x,y)_0}} \oplus D_{b_{(x,y)_1}} \dots \oplus D_{b_{(x,y)_{(d-1)}}})$ is random [from the property of XOR operation]

Hence, security is similar to the security of a one-time pad. Given the secret parameter K, it is impossible for an attacker to get $b_{(x,y)}$ from the random output bit $c_{(x,y)}$. Hence, irreversibility is achieved. □

From statements 1 and 2, it can be inferred that the proposed encryption scheme S provides both the properties, secrecy, and irreversibility over the bound of the secret parameter. Further, each input bit is independent of every other bit and is random. Hence, it never disregards the loop invariant condition of the computation theory and becomes the reason behind the preservation of error rate across the entire iris code.

As stated in ISO/IEC IS24745:2011 [22], the protected biometric template satisfies 3 main privacy properties:

4.6.1 Irreversibility

Irreversibility or non-invertibility states that it is computationally infeasible for an attacker to recover the original biometric template with or without the knowledge of secret parameters involved for a given protected biometric template. The original biometric templates can be guessed in the following ways:

1. **Predict one or both the original templates from the given protected biometric template:** Given that key is unknown, for each bit of the protected template, the attacker needs to reverse the Extraction and the Transformation functions of the bit-wise encryption scheme to get the corresponding one-bit of the original template. The mathematical proof shows that the bit-wise encryption scheme preserves irreversibility for both the algorithms, given the underlying transformation function F with a secret parameter, key K. Further, following the suggestions provided in Section 4.6 B, the helper data generated from the first biometric template using fuzzy extractor would not reveal any information about the original biometric template.

2. **Chosen Plaintext Attack:** Chosen plaintext attack presumes that an attacker can choose some arbitrary plaintext and can obtain corresponding ciphertexts. Let the client's device

acts as a random encryption oracle. Assume that attacker knows the key K and has access to the cancelable templates stored on the database. The CPA can be carried out by the targeted user attacker, as explained in Section 4.4. In our case, if the first biometric template B_1 is not known to the attacker, the attacker would not be able to derive the correct key K from it. Without the correct key, attacker cannot perform a chosen-plaintext attack on the second biometric template B_2.

4.6.2 Unlinkability

Unlinkability states that given different samples of the same instance of a particular user, an attacker should not be able to determine whether two biometric templates are derived from the same or the different instances.

4.6.2.1 Unlinkability Analysis of Fuzzy Extractor

The helper data generated as output from the fuzzy extractor is stored as public on the client's device. It often leaks information about linkability of biometric templates [1, 17] if accessed by an attacker during the offline attack mode. To prevent the linkability information leakage, we suggest the following procedures:

- The helper data can be encrypted using a system's specific secret key stored in the trusted platform module (TPM) [92] on the client's device. Since it is not possible to break the TPM, the key is secure, and hence, during the offline mode, the attacker will get only encrypted helper data.
- Recently proposed reusable fuzzy extractors [75, 77, 76] remain secure even when the attacker knows the helper data generated from correlated values of a user multiple times.

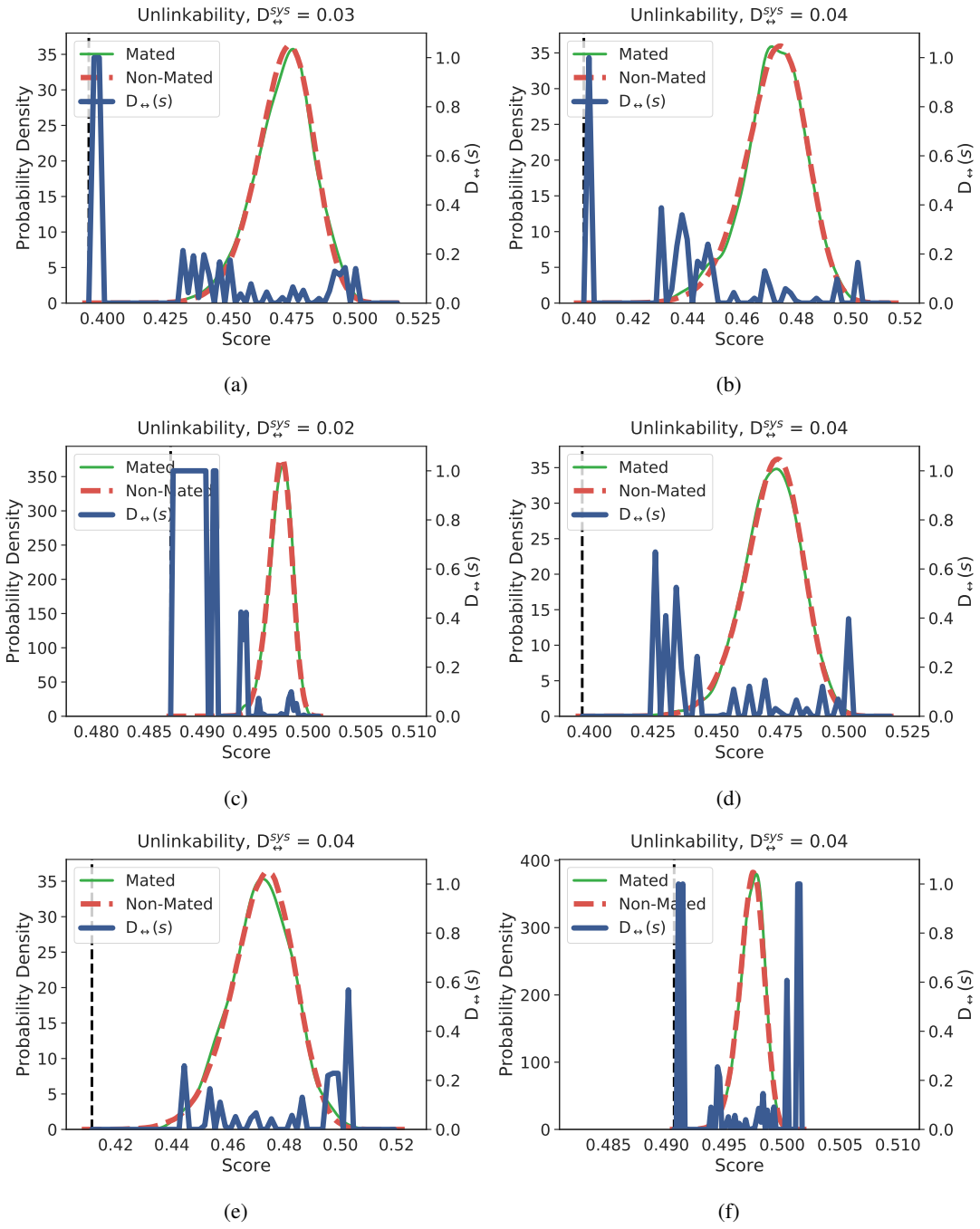


Figure 4.8: Unlinkability evaluation [3]: (a) IITD-CASIA Algorithm I (b) CASIA-Iris-Interval Algorithm I (c) Face Algorithm I (d) IITD-CASIA Algorithm II (e) CASIA-Iris-Interval Algorithm II (f) Face Algorithm II

4.6.2.2 Unlinkability Analysis of Bit-wise Encryption Scheme

Unlinkability is measured based on the mated (genuine and enroled samples across different applications) and non-mated (impostors and enrolled samples across different applications)

samples distribution. As proposed by Gomez et al. [3], we compute two different measures for the linkability of iriscodes. Local measure $D_{\leftrightarrow}(s)$ evaluates the linkability in a score-wise basis. Further, Global measure $D_{\underset{\leftrightarrow}{sys}}$ provides the linkability of the whole system and is independent of the system's score domain. Figure 4.8 shows the distribution of the mated and non-mated samples calculated from the dissimilarity scores. It is clear from the graph that the distributions are significantly overlapped with global linkability $D_{\underset{\leftrightarrow}{sys}}$ close to 0. Since the samples of biometric templates for a particular instance are similar but not identical, the key generated from these samples would be completely different for each sample. Thus, the unlinkability is preserved due to the use of different secret keys across multiple applications.

4.6.2.3 Further Linkage Functions: Hamming weights function applied on bloom template

The two bloom filter based biometric samples of the same instance protected using the different keys have similar Hamming weights [93, 4]. The Hamming weight difference $diff$ between two samples, bf_1, bf_2 is evaluated as $diff = |HW(bf_1) - HW(bf_2)|$, where HW is the number of ones in the bloom filter based template. In this case, knowledge of protected templates is required by an attacker to find the linkage between them. From the protected template generated using our proposed approach, the attacker can generate the bloom filter based templates, and the Hamming weights function can be applied to detect linkability. The plots for unlinkability measure are shown in Figure 4.9.

Considering two linkage functions (Hamming distance based comparison on protected iriscodes using proposed Algorithms and Hamming weights function on bloom filter based templates), for Algorithm I, the global linkability value of system for IITD-CASIA database is given as $D_{\underset{\leftrightarrow}{sys}} = max\{0.03, 0.09\} = 0.09$, for CASIA-Iris-Interval database, $D_{\underset{\leftrightarrow}{sys}} = max\{0.04, 0.04\} = 0.04$ and for face database, $D_{\underset{\leftrightarrow}{sys}} = max\{0.02, 0.05\} = 0.05$. For Algorithm II, the global linkability value of system for IITD-CASIA database is given as $D_{\underset{\leftrightarrow}{sys}} = max\{0.04, 0.09\} = 0.09$, for CASIA-Iris-Interval database, $D_{\underset{\leftrightarrow}{sys}} = max\{0.04, 0.06\} = 0.06$ and for face database, $D_{\underset{\leftrightarrow}{sys}} = max\{0.04, 0.05\} = 0.05$.

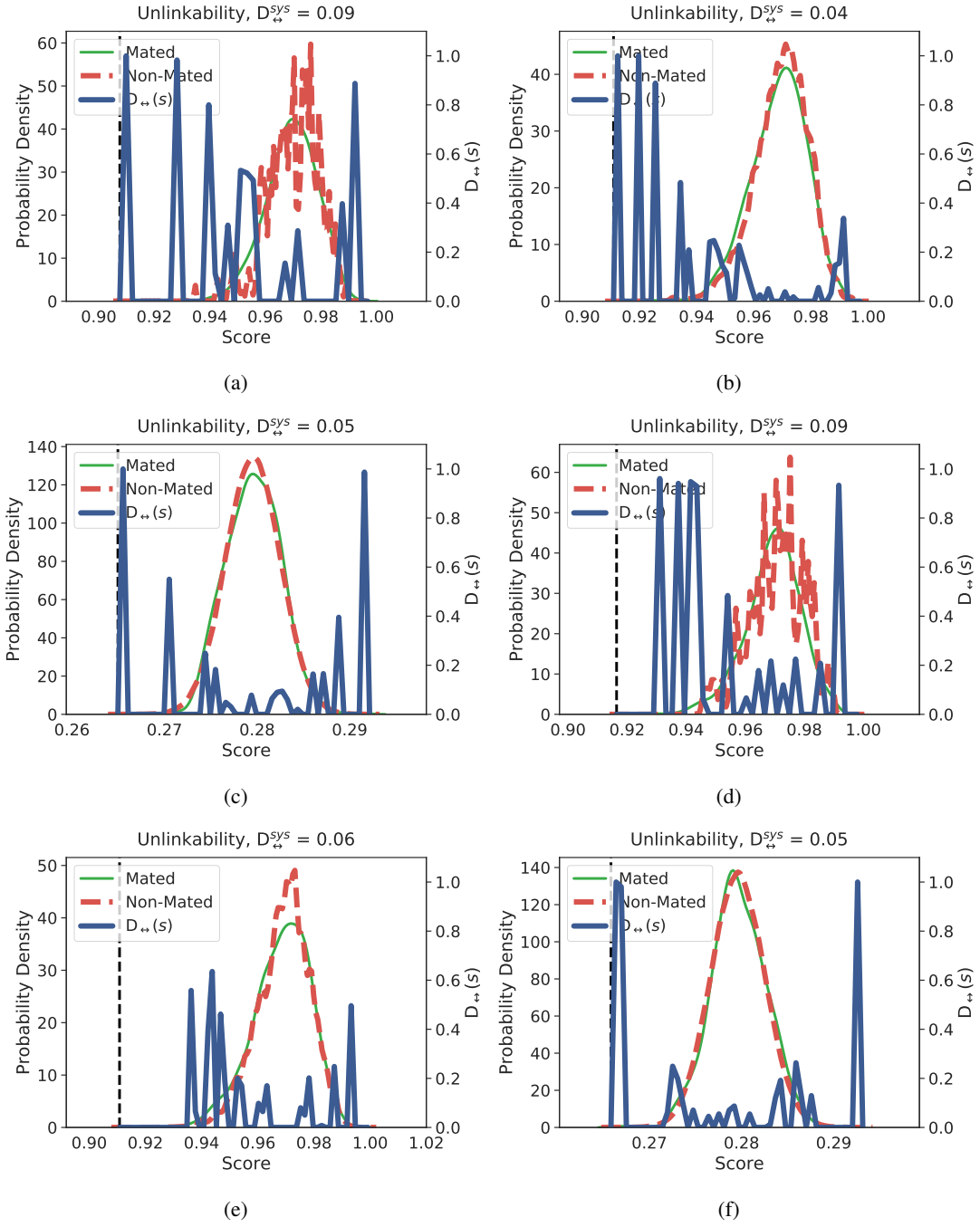


Figure 4.9: Further linkage functions [4]. Unlinkability evaluation using Hamming weight function: (a) IITD-CASIA Algorithm I (b) CASIA-Iris-Interval Algorithm I (c) Face Algorithm I (d) IITD-CASIA Algorithm II (e) CASIA-Iris-Interval Algorithm II (f) Face Algorithm II. The black dashed line represents likelihood ratio $LR(s)$ as denoted in [4]

.

4.6.3 Renewability

Renewability states that when an existing protected template is compromised, it should be possible to revoke or cancel the compromised protected template and re-generate a new protected biometric template using a different security parameter. This can be ensured by having a large key space, preferable with a key of length equal to or greater than 128. A new key can be issued if a key is compromised by changing the underlying error correcting codeword for a particular user.

4.7 Conclusions and Future Work

We propose a cancelable multi-biometric approach for a biometric authentication system by combining the fuzzy extractor with a novel bit-wise encryption scheme to generate cancelable biometric templates. The proposed work fulfills the ISO/IEC IS 24745:2011 recommended prerequisites for biometric template protection schemes stated as irreversibility, unlinkability, renewability, and high biometric recognition performance. The novel bit-wise encryption scheme ensures that no additional noise in terms of bit errors is generated in the protected template, making the performance of our proposed scheme equivalent to the unprotected systems' performance (Baseline-A). Further, the experimental results show that the proposed system outperforms the existing cancelable biometric scheme in terms of recognition performance. The empirical results for measuring efficiency in time units show that for desktop with clock frequency 3.3 GHz, our Algorithm II based approach takes around 12 milliseconds during enrolment and 18 milliseconds during authentication phase on an average of over 200 runs. The mathematical proof justifies the secrecy and irreversibility of the bit-wise encryption scheme.

Applications: We conclude that the proposed scheme is not only theoretically sound but can also be practically deployable on large scale systems such as India's Aadhaar project, particularly for Aadhaar-based verification [94], while achieving the high recognition performance and efficiency as well as strong security. In addition to the biometric authentication applications, the proposed bit-wise encryption scheme can be applied to a wide range of scenarios where

the input that needs to be secured is in the binary format. For instance, for any crucial security data (binarized) that needs to be stored on cloud servers. Another application could be digital forensics where fuzzy hashing [95] is used, one-bit encryption could be used as a part of the fuzzy hashing approach.

Limitations and Future Work: We provide a generalized, modular approach where the bit-wise encryption scheme is used with a feature extractor, a matcher, and a fuzzy extractor to provide secure storage of biometric templates on the database servers. The bit-wise encryption scheme preserves the errors completely, independent of the dataset or the underlying modules used. However, the recognition performance as well as efficiency is dependent on the feature extractors and the matchers used for the comparisons of two templates. To improve our approach's performance and efficiency, it could be built over the commercial-off-the-shelf systems (with optimized feature extractors and matchers) to achieve the equivalent performance by generating enhanced feature vectors and hence, could provide wider scalability.

Further, we assumed that error correcting codes can correct all the bit errors in the biometric templates. Working on improving the fuzzy commitment scheme's implementation would certainly be useful for this work, mainly while working with face templates. We worked on the efficient implementation of fuzzy commitment schemes in Chapter 5. Additionally, the reusable fuzzy extractor's practical implementation is still an open problem as the reusable fuzzy extractor takes vast storage. We would like to work on its implementation feasibility as a part of our future work.

Chapter 5

On the Security of Fuzzy Commitment Scheme for Biometric Authentication

Several multi-biometric authentication systems have been proposed in the literature [48, 6, 46, 49] that involves the use of one of the popular biometric cryptosystem– fuzzy commitment [27]. Besides, we proposed some multi-biometric authentication systems [59, 96] where the underlying biometric cryptosystems being used is the fuzzy commitment or the fuzzy extractor (fuzzy commitment with a key generation module) as discussed in Chapter 3 and Chapter 4. Figure 2.2 shows the fuzzy commitment scheme with the details given in Chapter 2.

The biometric template is noisy, i.e., bit-errors are present in the biometric templates such that no two samples of a particular instance are identical. Thus, the biometric template cannot be transformed directly into a protected template without incorporating any error tolerance mechanisms. It is, therefore, necessary to use error correcting codes (ECC) [41] defined in Chapter 2 to address the effect of noisy biometric data [1, 97] without leaking any information about the original template to the attackers. Several error correcting codes [41] are present such as Reed Solomon (RS) Codes [98], Hadamard Codes [99], Binary Bose-Chaudhuri-Hocquenghem (BCH) Codes [40], Turbo Codes [100] and the various combinations of these codes. Our work focuses on BCH codes as they have been extensively used in the literature [59, 96, 101] in the domain of biometric security. Also, it has been observed from [102] that these are simple to

perform and are more advantageous over other codes such as RS codes for correcting random and burst errors in the biometric data. The details of the BCH code are given in Chapter 2. The error correcting codeword is generated from a secret, random message Δ that contains δ bits. Note that in Chapter 2, the random message length δ is denoted by r_1.

The efficient implementation and the security of the fuzzy commitment scheme majorly rely on the implementation of underlying error correcting codeword. The efficiency of a fuzzy commitment scheme is measured in terms of the performance accuracy and the time taken to perform the user authentication. The size or length of the biometric template given as input B should be the same as that of the length of error correcting codeword W. It is due to the XOR operation between these parameters to generate secure sketch value S given as

$$B \oplus W \longrightarrow S \tag{5.1}$$

The above equation implies that if W is revealed to the attacker, the original biometric template B could be obtained using the public secure sketch value S. Furthermore, in the real-life scenarios, the biometric templates contain multiple bit errors, which could be higher than the error correcting capability t of the underlying error correcting codeword. In such cases, it would be challenging to implement the error correcting code with a sufficiently large secret message, preferably with $\delta \geq 128$ bits where δ denotes the length of the secret message.

These challenges prompt the need for an efficient implementation of error correcting codewords in the fuzzy commitment schemes. To fulfill the implementation requirements mentioned above, several approaches have been proposed in the literature as follows.

1. In case if the length of the biometric template is larger than the length of the error correcting codeword, a few extra bits from input biometric template are discarded from the beginning or the end [31] to equalize the lengths. It may lead to loss of discrimination [97] between the biometric templates of different users as the discarded bits may be relevant. For example, if the consistent bits in an iriscode are discarded [103], it might happen that the two samples of the same instance would not match.

2. To increase the error correcting capability of codewords, a larger size codeword is taken to correct more number of bit errors. The biometric template is padded with some random but fixed bits such as all zeros [31, 104, 105] in several ways. The details are given in Section 5.2. The padding technique results in higher accuracy; however, fixed padding such as zero insertions results in severe leakage of information about the codeword bits [106]. Knowing a few bits from codeword due to existing bit padding schemes could reveal the whole codeword and, in turn, the original biometric template. It could lead to an impersonation attack. We explain the attack in Section 5.3.

3. Flexible length error correcting codes like Turbo codes can be used whose length can be adjusted according to the biometric template's size. However, in turbo coding, the decoding process is hard to implement, and an additional efficient memory management scheme is needed [97].

In this regard, we focus on the following research questions.

- To what extent, we can preserve the trade-off between security and performance in biocryptosystems, particularly in the fuzzy commitment schemes?
- Can we prevent the existing security attack such as impersonation attack on the fuzzy commitment scheme while preserving the system's recognition performance?
  - Is there the need of any additional key or a password to improve the security?

Taking the motivation from requirements as mentioned above and challenges in the implementation of fuzzy commitment schemes along with the generalized attack proposed in literature [106, 97], we propose a user-specific, random padding scheme to prevent the attack on error correcting codewords. A few bits from the error correcting codeword are used as padding bits instead of using any other password or a key. We evaluate the recognition performance of our proposed system that shows a tremendous improvement in performance accuracy measured in terms of true match rate (TMR) as compared to the existing schemes, which are discussed in Section 5.2 (in our proposed scheme, TMR = 0.63). We further provide the security analysis

that shows that our proposed scheme's attack complexity is equivalent to the brute force attack complexity.

5.1 Models and Settings

We discuss the system participants, assets and the possible attack scenarios.

5.1.1 System Model and Participants

The system consists of users (genuine or impostors) and a server. The user provides input as the biometric characteristics, such as iris. The server stores the helper data generated using the iris template, along with the user ID corresponding to each user. The helper data constitutes the secure sketch value and the hash of codeword implemented in the underlying fuzzy commitment scheme as discussed in Chapter 2. The comparisons during authentication are performed on the server.

5.1.2 Attack Model

In the real world, an attack is possible on the server where the attacker can access the server database. In our scenario, such a server database includes the helper data corresponding to each enroled user, which is stored during the enrolment phase. When a padding approach is used (whether fixed, zero padding or random, secret padding), the attacker could know the system's algorithm. It further means that the attacker may know the biometric template positions where the fixed (such as all $0's$) or random padding has been done. In the case, if padding bits are also known to the attacker, it would lead to the security attack as discussed in Section 5.3 which reveals the original biometric template of a particular enroled/genuine user to the attacker.
The disclosed biometric template could be used for any malicious activities. For example, the attacker may impersonate the genuine user by providing the disclosed biometric template to successfully authenticate. It results in an impersonation attack.

5.2 Related Work

We describe the literature work related to the implementation of fuzzy commitment schemes for biometric templates. It includes the existing padding schemes proposed to implement error correcting codewords in the fuzzy commitment schemes.

5.2.1 Existing Fuzzy Commitment Schemes in Biometrics

Hao et al. [107] are the first to introduce the fuzzy commitment scheme for iris template protection. They design a secure, integrated biometric authentication system using two error correcting codes, Hadamard and Reed-Solomon in the fuzzy commitment scheme. While dealing with 10 to 20 percent of bit errors within an iriscode, the scheme generates a 140 bits biometric-dependent key. The key could be used for cryptographic purposes. Kanade et al. [31] propose two modifications to the scheme. They first introduce a secret shuffling of iriscode in which iriscode is permuted using a password provided by user [105, 104, 108]. The user's iriscode is shuffled to distinguish between genuine and impostor matches and increase attack complexity. It is followed by zero insertion in iriscode to improve the error correcting capability of Hadamard codes (increased beyond 25%). Two zeros are inserted after every three bits of iriscode, and the modified iriscode is XORed with error correcting codeword to generate a secure sketch value.

The zero padding is done either to correct more errors in the original biometric template or to equalize the length of the biometric template and the error correcting codeword. The zeros can be concatenated to the biometric template in the beginning, end, or somewhere in the middle of the biometric template. The insertion of zeros in the original biometric template increases the overall error correcting capability of the system. The insertion of zeros increases the size of the original biometric template, while the number of bit-error remains the same in the biometric template. Since the biometric template and codeword length is kept the same, a larger codeword would be required for the modified biometric template (with zero insertions) while implementing the fuzzy commitment scheme. Larger the codeword size, the more errors it will correct. Thus, it is possible to correct more errors with the modified biometric template while the number of bit-errors is the same in the original and modified biometric template. Thus the overall accuracy

of the system is increased.

The scheme proposed by Kanade et al. has two significant drawbacks. Firstly, the passwords and keys are not secure to implement. Passwords need to be memorized, which is difficult if passwords are not strong (entropy-wise), and a dictionary attack is a feasible option for attackers. Large keys are hard to memorize and thus needs to be stored somewhere. It is not a secure option as once an attacker discloses the key, padding bits can be revealed. Secondly, in [106, 97] it is shown that an attacker could get the entire δ secret message bits by just knowing the positions of δ or more than δ zeros in error correcting code. We describe the attack in Section 5.3 in which the problem of revealing the secret message Δ from δ known codewords bits can be reduced to a problem of solving a set of linear equations. Note that δ denotes the size of the secret message used to generate the particular codeword.

Most of the recent work [109, 110, 59, 96, 101] that incorporate fuzzy commitment schemes in biometrics implement BCH codes as the error correcting codes. In [111], a biometric cryptosystem is constructed by combining fingerprint templates with various error correcting codes, including BCH code, which gives promising results. Hoang et al. [110] propose an authentication system on mobile devices for gait characteristics in which BCH code is employed in the fuzzy commitment scheme. The scheme is proposed for gait features. However, deploying the same scheme for iriscodes, with BCH codewords involving no padding, would results in unsatisfactory recognition performance if $\delta \leq 128$ bits. Similarly, an improved fuzzy commitment scheme is proposed in [109], where additional security parameters are included in the generation of error correcting codeword. The evaluation is done on the BCH and RS code used as the underlying error correcting codes. The use of additional keys results in an additional security threat, as discussed above. Further, the implementation of BCH or RS code with no padding and $\delta \geq 128$ bits results in low recognition performance accuracy on the real databases. Reusable fuzzy extractors have been proposed in [75, 76] to prevent the leakage of linkability information from the secure sketch value stored on the database server. They do not use error correcting codes in the implementation. However, they require massive storage to store public data. Further, the reusable fuzzy extractors' recognition performance is low, which makes them impractical to use in real-time operations.

5.3 Existing Attack on Error Correcting Codewords

In the subsection, we describe an attack mentioned in [106, 97] on error correcting codes. It allows an attacker to reveal the entire biometric template by just solving a set of linear equations with a complexity less than that of a brute force attack. Since the security of codeword depends on δ message bits from which it is generated, the brute force attack to get the correct codeword will take 2^{δ} number of trials. To explain the attack, we consider the BCH-error correcting code in our paper. With the assumption that zero padding/fixed padding is performed on the iriscode template, the attacker knows the relative positions where zeros have been inserted [31].

In the case of a fixed padding scheme, if the input string is zero-padded with δ or more than δ bits, after padding, the fuzzy commitment scheme is given as

$$B \| \{0\}^{\delta+} \oplus W = S \tag{5.2}$$

where $||$ denotes concatenation between two strings and $\{0\}^{\delta+}$ denotes that δ or more than δ zeros are inserted. Because of zero insertion, the attacker will get to know about the corresponding δ or more than δ bits of the codeword$W$ from public, secure sketch value S. The known bits as a part of codeword W are used to solve a set of a linear system of $(n-\delta)$ equations with δ or more than δ linearly independent equations to get δ unknown message bits that generate codeword, where $(n-\delta) \geq \delta$. The $(n-\delta)$ equations are derived from $(n-\delta)$ parity bits. Atleast δ parity bits are known in codeword W due to zero padding. Therefore, we can construct atleast δ independent linear equations out of $(n-\delta)$ equations which are further solved to get δ unknown secret message bits. Once, the codeword is revealed from the δ secret message bits, the biometric template B can be known by the attacker from S using (5.2). It can lead to an impersonation attack. The step-by-step procedure is described as follows.

Let the parity-check bits in the polynomial form in error correcting codeword $W(x)$ is given by $P(x)$ which is of size $(n-\delta)$. The secret message Δ of length δ bits is used to generate parity-check bits and is represented in the form of polynomial given as $\Delta(x)$. $g(x)$ is the generator polynomial of BCH code. Then,

$$W(x) = \Delta(x) * P(x)$$

where

$$P(x) = x^{n-\delta}\Delta(x) mod(g(x))$$

and

$$\Delta = \Delta_{\delta-1}\Delta_{\delta-2}\cdots\Delta_0$$

Since, the equation is linear under modulus operation,

$$\begin{aligned} P(x) &= x^{n-\delta}\Delta_{\delta-1}x^{\delta-1} mod(g(x)) \oplus x^{n-\delta}\Delta_{\delta-2}x^{\delta-2} mod(g(x)) \oplus \ldots \oplus x^{n-\delta}\Delta_0 mod(g(x)) \\ &= \Delta_{\delta-1}x^{n-1} mod(g(x)) \oplus \Delta_{\delta-2}x^{n-2} mod(g(x)) \oplus \ldots \oplus \Delta_0 x^{n-\delta} mod(g(x)). \end{aligned}$$

Representing $x^{n-i} mod(g(x))$ as vector V_{n-i}, where $n-\delta \leq i \leq n-1$, parity-check can be written as

$$P(x) = \Delta_{\delta-1}V_{n-1} \oplus \Delta_{\delta-2}V_{n-2} \oplus \ldots \oplus \Delta_0 V_{n-\delta}. \tag{5.3}$$

As generator polynomial $g(x)$ is known for a pair (n, δ) of ECC, therefore, vector V_{n-i} is known to an attacker. Constructing $n-\delta$ linear equations from (5.3) with Δ_j as unknowns, where $0 \leq j \leq (\delta-1)$, we get

$$\begin{bmatrix} V_{n-1}^1 & V_{n-2}^1 & \ldots & V_{n-\delta}^1 \\ V_{n-1}^2 & V_{n-2}^2 & \ldots & V_{n-\delta}^2 \\ V_{n-1}^3 & V_{n-2}^3 & \ldots & V_{n-\delta}^3 \\ \vdots & \vdots & \ldots & \vdots \\ \vdots & \vdots & \ldots & \vdots \\ V_{n-1}^{n-\delta-1} & V_{n-2}^{n-\delta-1} & \ldots & V_{n-\delta}^{n-\delta-1} \\ V_{n-1}^{n-\delta} & V_{n-2}^{n-\delta} & \ldots & V_{n-\delta}^{n-\delta} \end{bmatrix} \begin{bmatrix} \Delta_{\delta-1} \\ \Delta_{\delta-2} \\ \Delta_{\delta-3} \\ \vdots \\ \vdots \\ \Delta_1 \\ \Delta_0 \end{bmatrix} = \begin{bmatrix} P_{n-\delta-1} \\ P_{n-\delta-2} \\ P_{n-\delta-3} \\ \vdots \\ \vdots \\ P_1 \\ P_0 \end{bmatrix} \tag{5.4}$$

Since, input biometric template is padded with δ or more than δ zeros, the attacker will check the corresponding bit positions in the error correcting codeword to recover the bits in codeword at those positions. We have taken the case when the recovered bits in codeword are a part of parity-check bits denoted by P_u, where $0 \leq u \leq (n-\delta-1)$. Using (5.4) with δ or more than

δ known parity-check bits on the right hand side of the matrix, we can generate δ independent linear equations with δ unknown secret message bits (given by δ), forming a system of linear equations.

System of Linear Equations: The linear equation in the matrix form written as $AX = B$ gives the solution set in the matrix form. Considering a square matrix A with all the rows as independent rows, then the system has a unique solution given by [112]

$$X = A^{-1}B$$

where A^{-1} is the inverse of A.

Taking X as the unknown secret message Δ, A as the known vector V and B as the parity check matrix P with $\geq \delta$ known values, the attacker can solve the independent linear equations using the above equation to get δ unknown secret message bits from which codeword is generated. Once the attacker gets the codeword W, the original input binary template of user can be obtained by simply XORing W with secure sketch value S using (5.2). Thus, knowing only the δ bits from parity-check bits can reveal the user's biometric template. In the case of brute force, the attack complexity would be 2^{δ} bits, whereas, in our mentioned attack scenario, the attack complexity is much less as it becomes equal to the complexity of solving a system of linear equations.

In our paper, the attack mentioned above is mainly documented on BCH codes. However, it is also possible to perform the attack on the family of BCH codes, including the Reed Solomon codes and other members of the same family[106, 97].

5.4 Proposed Work

We propose a user-specific, random padding approach in implementing a fuzzy commitment scheme that enhances the security of the overall system while preserving the high recognition performance and efficiency. Our proposed scheme prevents the existing attack on error correcting codes, which is described in Section 5.3. The fixed padding scheme, such as zero insertions in

the biometric template, could reveal the biometric template to the attacker. Our proposed scheme provides a secure means of authentication by introducing the random, secret padding bits in biometric templates while implementing the fuzzy commitment scheme. It leaks no information about the original biometric template.

The proposed scheme works on the principle that same, but random padding bits must be used during enrolment and authentication for a particular individual. Since we solely rely on the biometric template without any use of a password or a key, the padding bits are derived from the underlying error correcting codeword. While at the same time, the padding bits are random and are not stored anywhere during the enrolment in order to avoid being guessed by the attacker.

5.4.1 User-specific, random Padding Using Codeword Bits

The user provides its biometric template in the form of a binary string $B \in \{0,1\}^n$. For error correction, an error correcting codeword $W \in \{0,1\}^m$ is implemented. The biometric template and codeword length should be exactly equal, i,e, $n = m$. We consider the case when the length of the underlying codeword is greater than the length of the input biometric template given as $|B| < |W|$ or $n \leq m$. It ensures the correction of more number of bit errors in the biometric template due to the large size codeword. In such a case, padding of a few bits is required in B such that the padding bits are random, secret, and at the same time, $|B| = |W|$. Insertion of a few random, padding bits increases the size of B while keeping the bit-errors same as that of the original B (before padding). With a large codeword of length equal to the length of B after padding, the fuzzy commitment scheme can correct more number of bit errors, thus improving the system's recognition performance.

We split B into i different blocks such that

$$B = B_1 \| B_2 \| \ldots \| B_{i-1} \| B_i$$

Each block B_j where $1 \leq j \leq i$ is of length $n_1, n_2, \ldots, n_{i-1}, n_i$, where the length could vary for each block. For each of the block B_j, a codeword W_j of length m_j is generated. For the

purpose of generality and simplicity, we consider the case when the length n_i for last splitted block B_i is less than the size m_i of the corresponding codeword W_i used for that particular block. It means that we selected an error correcting codeword for the last block B_i such that the length m_i of W_i is sufficiently greater than the length n_i. In such case, we would require padding bits for B_i to equate the lengths.

Algorithm 1: Enrolment Phase

Input: Iriscode B

Output: Secure sketch values S_j for $1 \leq j \leq i$ and $i \geq 2$, hash value $H(W_i)$

1. $B = B_1 \| B_2 \| \ldots \| B_{i-1} \| B_i$
2. **For** $1 \leq j \leq i-1$
3. $W_j \xleftarrow{\delta_j} \Delta_j$, $\triangleright$ Δ_j denotes j-th random secret message
4. $S_j = B_j \oplus W_j$, $\triangleright$ $|W_j| = |B_j|$
5. $j = j + 1$
6. **For** $j = i$
7. $W_i \xleftarrow{\delta_i} \Delta_i$, $\triangleright$ $|W_i| > |B_i|$
8. $H(W_i) \xleftarrow{H} W_i$
9. $pad = m_i - n_i$, $\triangleright$ $m_i = |W_i|$, $n_i = |B_i|$
10. $B_i \| \{W_{i-1}\}^{pad} \oplus W_i = S_i$
11. **Return** $S_1, S_2, \ldots, S_i, H(W_i)$

For $1 \leq j \leq i-1$, a secure sketch value S_j is derived using fuzzy commitment scheme with B_j and corresponding W_j as inputs. It is given as

$$B_1 \oplus W_1 = S_1$$

$$B_2 \oplus W_2 = S_2$$

$$\vdots$$

$$B_{i-1} \oplus W_{i-1} = S_{i-1}$$

For the last block B_i, we propose to use a few codeword bits from the codeword W_{i-1} belonging to a previous block to generate the random padding bits. The number of padding bits required are given by

$$pad = m_i - n_i \tag{5.5}$$

These padding bits are replicated from the codeword W_{i-1} and are appended to block B_i. Using the fuzzy commitment scheme, the enrolment phase for last block of biometric template is given as,

$$B_i \| \{W_{i-1}\}^{pad} \oplus W_i = S_i$$

$\{W_{i-1}\}^{pad}$ denotes that the pad number of bits from W_{i-1} are taken for concatenation. The secure sketch values S_1, S_2, ..., S_{i-1}, S_i are stored on server as public values along with the hash of codeword W_i, denoted as $H(W_i)$. The values S_1 to S_i and $H(W_i)$ constitute the helper data which is stored on database server.

The enrolment phase is described in Algorithm 1. The user provides iriscode as input, which is known as the biometric template B. We could split the biometric template into any number of blocks of biometric templates with 2 minimum blocks, i.e., $i \geq 2$. In Section 5.5, we perform our experiments with biometric templates split into 2 blocks.

During the authentication phase, the user provides $B' \in \{0,1\}^l$. Similar to the enrolment phase, B' is splitted into i blocks of the respective lengths as in the enrolment phase such that

$$B' = B'_1 \| B'_2 \| \ldots \| B'_{i-1} \| B'_i$$

From the secure sketch values S_j for $1 \leq j \leq i-1$ and $i \geq 2$ stored as a part of helper data on the database server, the original codeword W_j is decoded as

$$\begin{aligned} B'_1 \oplus S_1 &= W'_1 \\ B'_2 \oplus S_2 &= W'_2 \\ &\vdots \\ B'_{i-1} \oplus S_{i-1} &= W'_{i-1} \end{aligned}$$

Algorithm 2: Authentication Phase

Input: Iriscode B', Secure sketch values S_j, where $1 \leq j \leq i$ and $i \geq 2$, hash value $H(W_i)$

Output: a bit $1/0$

1. $B' = B'_1 \| B'_2 \| \ldots \| B'_{i-1} \| B'_i$
2. **For** $1 \leq j \leq i-1$
3. $W'_j = B'_j \oplus S_j$, $\triangleright$ $|W_j| = |B'_j|$ and error correcting capability of W_j is t_j
4. W'_j is decoded to W''_j if $HD(B_j, B'_j) < t_j$
5. $j = j + 1$
6. **For** $j = i$
7. $pad = m_i - n_i$, $\triangleright$ $m_i = |W_i|$, $n_i = |B'_i|$
8. $B'_i \| \{W_{i-1}\}^{pad} \oplus S_i = W'_i$
9. W'_i is decoded to W''_i if $HD(B_i, B'_i) < t_i$
10. **if** $H(W''_i) = H(W_i)$
11. **Return** 1, $\triangleright$ User is successfully authenticated
12. **else**
13. **Return** 0

If B'_j,B_j (for $1 \leq j \leq i-1$) satisfy the condition that $HD(B'_j, B_j) \leq t$, where HD calculates the hamming distance between two binary strings, W'_j is decoded using t-error correcting codeword to get the codeword W''_j which is equal to codeword W_j.

For $j = i$, i.e. for the last block, the length of B'_i is not equal to length of W_i, we need padding bits in order to perform XOR operation between these two. The pad number of bits are replicated from the decoded codeword $W''_{i-1} = W_{i-1}$ and appended to the B'_i. We perform,

$$B'_i \| \{W_{i-1}\}^{pad} \oplus S_i = W'_i$$

If B'_i,B_i satisfy $HD(B'_i, B_i) \leq t$, W'_i will be decoded to the codeword W''_i. If $H(W''_i)$ is equal to

the hash value stored on server $H(W_i)$, the user is authenticated successfully. The authentication phase is described in Algorithm 2. The user provides input as the similar but not identical, biometric template B'.

The proposed scheme's multiple constructions can be designed by taking the blocks in any order while maintaining their order during the enrolment and authentication phase. Further, in a system with i blocks with $i \geq 2$, padding can be applied to any number of blocks between 1 block to $i-1$ blocks. According to the statistical distribution of bit errors among multiple datasets, the design flexibility helps to achieve high-performance accuracy while preserving security.

5.4.2 An Example of Proposed Random Padding Scheme

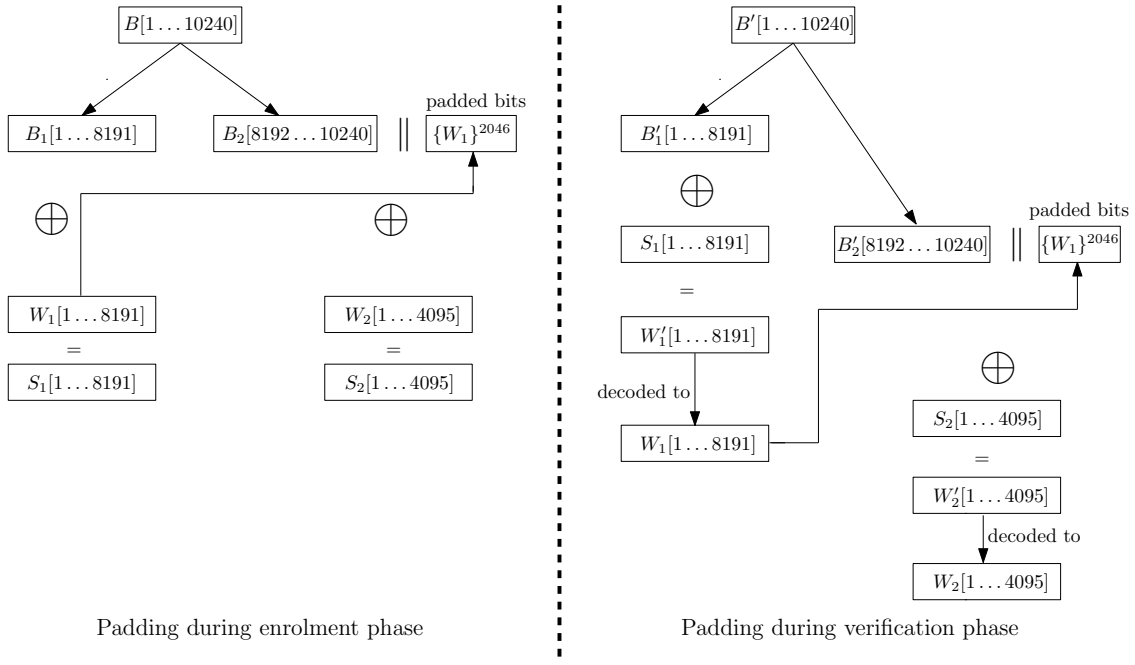


Figure 5.1: Proposed user-specific, random padding scheme. B, B' represents the input biometric templates during registration/enrolment and verification respectively. C denotes the error correcting codeword and S represents the secure sketch value.

An iriscode B consists of l= 10240 bits. For a secure configuration of codewords, we aim to correct a maximum number of errors by using BCH error correcting codewords such that the length of the secret message is given as $\delta \geq 128$ bits. We split the iriscode into 2 blocks. For the blocks of iriscode, we select two BCH-codewords $C_1 = (8191, 131, 1759)$ and $C_2 = (4095, 134, 763)$. Figure 5.1 shows an example of our proposed padding scheme.

Now, B is splitted as

$$B = B_1[1 \ldots 8191] \| B_2[8192 \ldots 10240]$$

Accordingly, the corresponding error correcting codewords are selected as follows.

$$C = C_1[1 \ldots 8191] \| C_2[1 \ldots 4095]$$

Since the number of bits in B_2 are less than the number of bits in C_2, padding bits are required which are obtained by replicating first $4095 - (2048 + 1) = 2046$ bits from C_1.

$$B = B_1[1 \ldots 8191] \| B_2[8192 \ldots 10240] \| \{C_1\}^{2046}$$

$$C = C_1[1 \ldots 8191] \| C_2[1 \ldots 4095]$$

$$S_1 = B_1[1 \ldots 8191] \oplus C_1[1 \ldots 8191]$$

$$S_2 = B_2[8192 \ldots 10240] \| \{C_1\}^{2046} \oplus C_2[1 \ldots 4095]$$

During the enrolment phase, the secure sketch values S_1 and S_2 along with the hash of codeword C_2 represented as $H(C_2)$ will be stored on the database server.

During the verification phase, user provides $B'[1 \ldots 10240]$. Using the secret sketch value S_1 stored on server, C_1' is retrieved as

$$B_1'[1 \ldots 8191] \oplus S_1 = C_1'[1 \ldots 8191]$$

which is decoded to C_1''. Now, if the user is genuine user with the number of errors between B_1' and B_1 are less than 1759, the decoding is correct, i.e. $C_1'' = C_1$. Note that 1759 is the maximum number of error count that the codeword C_1 could correct.

The padding is performed in the second block of the biometric template. It is then XORed

with the secure sketch value S_2 to derive C_2' given as

$$B_2'[8192 \ldots 10240] \| \{C_1\}^{2046} \oplus S_2 = C_2'[1 \ldots 4095]$$

The C_2' is correctly decoded to C_2'' if the number of errors in B_2 and B_2' is less than the maximum error correcting capability of the second block, i.e., 763. If $H(C_2) = H(C_2'')$, the authentication is successful.

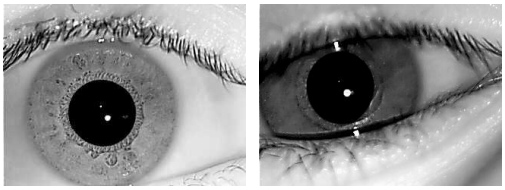
5.4.3 Design Rationale Behind the Use of Error Correcting Codeword Bits as Padding Bits

As discussed above, padding of some random, secret bits is required in the biometric template to implement the fuzzy commitment scheme. An additional parameter, such as a secret key or a password provided by each user or the system, could be used to generate padding bits. However, the use of such an additional secure parameter has its drawbacks, as discussed in Section 5.2. Therefore, we could choose the padding bits either from the biometric template B itself or from the error correcting codeword W. The biometric template B is not uniformly random [113]. When padding bits are taken from B, an attacker may reveal the padding bits due to low entropy. Whereas the error correcting codeword is a random string [1, 40]. It is derived from a system's generated secret message Δ. Duplicating some of the bits from error correcting codeword to get the padding bits would not reveal any information to the attacker. Even if the attacker knows the length of padded bits and the positions where padding bits are placed, it does not disclose any significant information about the original biometric template to the attacker. Refer Section 5.6 for details.

5.5 Experiments and Performance Analysis

We analyze the system's performance in terms of recognition accuracy and the time taken to perform biometric authentication. We perform our experiments on the publicly available IIT-Delhi iris database [63]. The IIT-Delhi database consists of 420 instances, 5 samples per instance.

The example images are shown in Figure 5.2. We consider the left and right instances of the database as mutually independent subjects.



(a) IIT-Delhi Database

Figure 5.2: Example images from the selected database for proposed randing padding approach

To obtain the iriscode from the iris biometric characteristics, we use several open-source libraries such as OSIRIS [65] and University of Salzburg Iris Toolkit v1.0 [66]. We use Daugman-like 1D-Log Gabor (LG) algorithm proposed by Masek [67] for feature extraction in the iris. An iriscode of length 10240 bits is generated as an output. For experimentation, we split the input biometric template, i.e. the iriscode into 2 blocks, i.e. $i = 2$. We use BCH codes with various configurations (shown in Table 5.1) for the comparisons of various existing approaches with our proposed scheme. We use BCH encoding and decoding modules written in C language[1]. The configurations are selected according to the lengths of each block in the iriscode. In our proposed work, we generate the random error correcting codeword such that the size of secret message Δ used to generate the particular codeword is large, preferably $\delta \geq 128$ bits. It ensures that an attacker would not be able to guess the random codeword using the brute force approach on the secret message Δ. Further, we use SHA-256 as the hash function in the implementation of the fuzzy commitment scheme.

5.5.1 Recognition Performance Evaluation

Biometric recognition performance measures how well the system authenticates the genuine users. We evaluate the recognition performance comparison of the iris database for various existing constructions of fuzzy commitment schemes. The performance is evaluated in terms of the true match rate generated by the system for various values of the length δ of secret message Δ. δ is a security parameter. The higher the length δ, the more is the security as discussed in

[1] http://www.eccpage.com/bch3.c

Table 5.1: True match rate (TMR) along with security comparison for various fuzzy commitment schemes designed for iris biometric templates. The padding bits are placed after the concatenation operator $'\|'$. δ_1 and δ_2 denotes the length of secret message Δ_1 and Δ_2 used to generate codewords W_1 and W_2.

Approaches	**Biometric Template** B		**Codeword** W, (n, δ, t)		**TMR** (in %)	**Security** (in δ **bits**
	B_1	B_2	W_1	W_2		
Baseline A [1]	8191	2047	(8191,131,1759)	(2047,133,365)	0.19	$\delta_1, \delta_2 \geq 128$
	2047	8191	(2047,133,365)	(8191,131,1759)	0.17	$\delta_1, \delta_2 \geq 128$
Baseline B [107]	8191	2047	(8191,14,2047)	(2047,12,511)	0.49	$\delta_1, \delta_2 \leq 128$
	2047	8191	(2047,12,511)	(8191,14,2047)	0.61	$\delta_1, \delta_2 \leq 128$
Zero padding [31]	$10240\|\|\{0\}^{6143}$		(16383,134,3575)		0.93	$\delta_1, \delta_2 \geq 128$[2]
Hoang et al. [110]	8191	2047	(8191,131,1759)	(2047,133,365)	0.19	$\delta_1, \delta_2 \geq 128$
	2047	8191	(2047,133,365)	(8191,131,1759)	0.17	$\delta_1, \delta_2 \geq 128$
Chauhan et al. [109]	8191	2047	(8191,14,2047)	(2047,12,511)	0.49	$\delta_1, \delta_2 \leq 128$
	2047	8191	(2047,12,511)	(8191,14,2047)	0.61	$\delta_1, \delta_2 \leq 128$
Proposed	**8191**	$2049\|\|\{W_1\}^{6142}$	**(8191,131,1759)**	**(8191,131,1759)**	**0.63**	$\delta_1, \delta_2 \geq 128$
	8191	$2049\|\|\{W_1\}^{2046}$	**(8191,131,1759)**	**(4095,134,763)**	**0.64**	
	$2049\|\|\{W_2\}^{2046}$	8191	(4095,134,763)	(8191,131,1759)	0.59	
	$2049\|\|\{W_2\}^{6142}$	8191	(8191,131,1759)	(8191,131,1759)	0.59	

Section 5.6. True match rate is defined as the number of genuine samples correctly verified as genuine by the system. False match rate is defined as the number of impostors correctly verified as genuine by the system.

Figure 5.3 shows the recognition performance comparison of our proposed approach with the existing approaches. It is observed from the performance graphs that the true match rate tends to decrease on increasing the length of the secret message, which is denoted as δ and increases with the decreasing length δ. It is implicit from the fact that the higher the length of secret message Δ, the less number of errors t would be corrected by a particular error correcting codeword [114]. It will reduce the true match rate. Therefore, a performance-security trade-off always exists in the implementation of error correcting codes. The higher the length δ, the fewer errors would be corrected, which implies high security with low true match rates. Based on the performance derived from Figure 5.3, we select the value of δ such that $\delta \geq 128$ for our proposed approach and compare our proposed approach with the existing constructions. When $\delta \leq 128$, a brute force attack is possible, which could reveal the correct secret message Δ. The performance accuracy, along with security comparisons, is given in Table 5.1 for the below-mentioned approaches.

- **Baseline A:** It denotes the original fuzzy commitment scheme proposed by Dodis et al. [1]. The iriscode is combined with BCH codeword without any padding involved. To equate the lengths of iriscode and codeword, we consider two codewords such that B_1=8191 bits

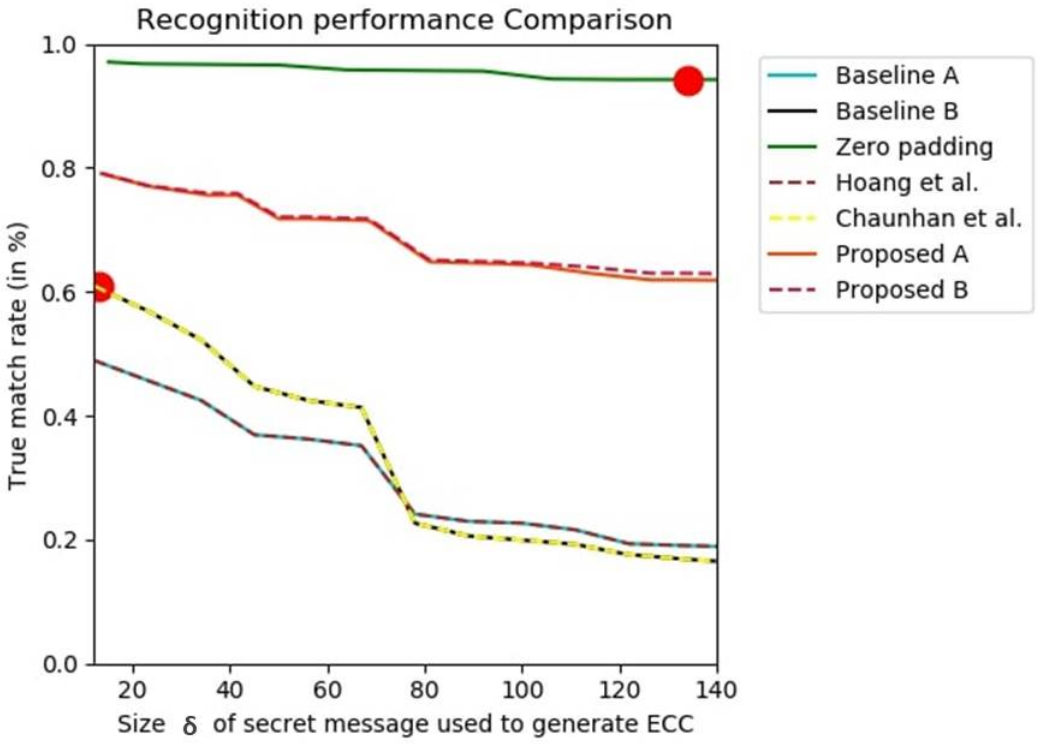


Figure 5.3: Comparison of our proposed scheme with existing schemes based on true match rate (TMR) with respect to the length δ of secret message Δ, which is used to generate the error correcting codeword in the fuzzy commitment scheme. The red, round markers on the curves denote the TMR at a particular value of δ. It further denotes that the underlying scheme provides the specific true match rate shown by the red marker at the corresponding value of δ. In the remaining schemes (with no red mark), the performance in terms of TMR is calculated along with several values of δ, while $k \geq 128$ being the secure value. If δ is small, the underlying error correcting codeword (C) will correct a large number of errors (t) in the biometric template (B), and thus, the true match rate would be high and vice versa.

with codeword W_1=8191 bits and B_2=2047 bits with codeword W_2=2047 bits. The two extra bits from the biometric template are discarded. The order of lengths of blocks could also be interchanged, as shown in Table 5.1. In Figure 5.3, we plot the curve for the best case out of two possible, i.e. B_1=8191 bits and B_2=2047 bits

- **Baseline B:** Denotes the scheme proposed by Hao et al. [107]. To maintain consistency in the results, we modify the approach using BCH codes instead of the concatenation of Hadamard and RS codes. We selected the value of δ such that the BCH codeword could correct maximum errors as in the proposed scheme [107]. In Figure 5.3, we plot the curve for the best case out of two possible, i.e. B_1=2047 bits and B_2=8191 bits.

- **Zero padding:** Kanade et al. [31] propose the insertions of zeros in the biometric template

before implementing the fuzzy commitment scheme. We implement it by considering zero insertions in the biometric template B of length 10240 to get the modified length of 16383 bits. A BCH codeword of similar length is then used.

- **Haong et al. [110]:** No padding approach is taken into consideration in [110]. For the iris template, we perform similar operations as done in the Baseline A case.
- **Chauhan et al. [109]:** In [109], the maximum number of errors are corrected from the BCH code by keeping the value of δ low. We perform a similar approach to the chosen iris dataset. The experimental setup is similar to Baseline B.
- **Proposed:** In the proposed approach, we split the iriscode into $i = 2$ blocks as described in Section 5.4. The experiments are conducted on various designs, where the best two cases are given as:
  - **Proposed A:** B_1=8191 bits with codeword W_1=8191 bits and B_2=2049 bits padded with 6142 bits of codeword W_1. The length of codeword W_2=8191 bits.
  - **Proposed B:** B_1=8191 bits with codeword W_1=8191 bits and B_2=2049 bits padded with 2046 bits of codeword W_1. The length of codeword W_2=4095 bits.

Following are the observations:

- The proposed scheme gives the best recognition performance accuracy among all the secure schemes by ensuring no fixed padding and $\delta \geq 128$ bits. We achieve a true match rate of approximately 64% as the best case for our database.
- The schemes involving zero padding [31] gives the highest accuracy of around 93%. Even though $\delta \geq 128$, the overall security of the system is compromised due to the attack discussed in Section 5.3.
- In Figure 5.3, the Baseline B case is shown by the red marker, where the high accuracy of the system is achieved by taking the lowest values of δ, which could enable the BCH code to correct the errors to its maximum capability. Even though the performance is quite

equivalent to our proposed approach's performance, the Baseline B approach is highly insecure due to the small length of δ, which results in a brute force attack.

- We also calculate the false match rate (FMR) of our proposed approach as well as the existing approaches. We found that the false match rate is zero or almost negligible for all the cases. It contributes to high security in the critical systems when the priority is to reject an intruder or an impostor rather than to keep the true match rate high.

- From Table 5.1, we observe that the security in terms of lengths of Δ_1 and Δ_2 is low, i.e. $\delta_1, \delta_2 \leq 128$ bits for the Baseline B [107] and Chauhan et al. [109]. Further, in Baseline A [1] and Hoang at al. [110], even though the security is high, the performance is highly degraded. For our proposed approach, the security is high, i.e. $\delta_1, \delta_2 \geq 128$, and as we use random padding instead of fixed (such as [31]) padding, our scheme is protected against the possible security attack discussed in Section 5.3.

- Several other designs of our proposed scheme are mentioned in Table 5.1. These include the design where the block B_2 is processed first, and then the block B_1 is padded with the error correcting codeword bits from W_2 (as shown in the last two rows). In such a case, W_2 represents the codeword W_{i-1} and W_1 represents the codeword W_i explained in Section 5.4. Further, several other designs with various configurations of B_1, B_2, W_1 and W_2 are possible such as considering B_1 of size 4095 bits and B_2 of size 6145 bits with padding. We observe that such designs generally degrades the performance of the system. The reason behind the low performance for these configurations is the high error rate in the block B_1, which requires a codeword with large error correcting capability. If the size of B_1 is large (8191 bits as in the first two cases of the proposed approach), the equal-sized codeword would correct more errors in that block. Hence, the true match rate is high. Whereas, for the remaining cases, the size of B_1 is small, resulting in the correction of fewer errors by the small-sized corresponding W_1 therefore, low true match rate.

- While the overall accuracy of our proposed scheme is quite low for implementation in the real-life scenarios; it still surpasses the accuracy achieved with baseline approaches by a significant number. For real-life implementations, the configurations can be manipulated

Table 5.2: Time taken (in seconds) by the best performance cases of various schemes plotted in Figure 5.3. The enrolment time includes the time taken for BCH encoding to generate the codeword and the time taken to compute the XOR operation between the biometric template and codeword. The authentication time includes BCH decoding time and the time taken to compute XOR operation between the secure sketch value and the biometric template. The time taken to perform XOR operation is computed to be approximately 0.00002 seconds.

Approaches	**Biometric Template B**		**Efficiency (in Seconds)**	
	B_1	B_2	**Enrolment**	**Authentication**
Baseline A [1]	8191	2047	0.002213	0.256226
Baseline B [107]	2047	8191	0.002213	0.256226
Zero padding [31]	$10240 \| \{0\}^{6143}$		0.004952	0.531250
Hoang et al. [110]	8191	2047	0.002213	0.256226
Chauhan et al. [109]	2047	8191	0.002213	0.256226
Proposed A	8191	$2049 \| \{W_1\}^{6142}$	0.003536	0.409550
Proposed B	8191	$2049 \| \{W_1\}^{2046}$	0.002694	0.308337

according to the database and the particular database's error distribution to get the desired accuracy. We can increase the accuracy by taking sufficiently larger size error correcting codewords so that more number of errors could be corrected in the biometric templates. However, it would increase the authentication time due to the increase in computations of the large-size codeword.

5.5.2 Efficiency In Terms of Authentication Time

We also evaluate our proposed scheme's efficiency in terms of time taken during the enrolment and authentication phase. The time is taken during the enrolment phase, which majorly includes the BCH encoding time and an XOR operation performed between the input biometric template B and the codeword$W$ generated from BCH encoding. The time taken during the authentication phase includes the time taken for an XOR operation between B' and the secure sketch value S to obtain codeword W'. It also includes the time taken for BCH decoding that decodes the codeword W' to obtain the original codeword $W'' = W$. Table 5.2 shows the enrolment and authentication time for various approaches. In the approaches where the biometric template is split into 2 blocks, the BCH code encoding/decoding and XOR operation are performed twice, once for each block. The time is measured in terms of seconds on an average of over 200 runs each.

The experiments to calculate the average time are performed on a server-grade processor;

Intel i7 2.6 GHz quad-core processor (with hyper-threading) with 16GB RAM architecture for profiling BCH encoding and decoding subroutines. The source code is compiled using GCC 10.2 compiler. Further, we have used several compiler flags that allow fast compilation of the code.

Following are the inferences:

- We obtain the authentication time as approximately 0.3 seconds for one of the best cases in our proposed approach denoted as Proposed B. It is comparable to the time taken by the original or the baseline cases. Shorter authentication time helps to deploy our proposed approach on wider scale biometric systems where real-time processing is a major requirement.
- Small authentication time proves the fact that the entire authentication process is not resource-intensive. Hence, our proposed scheme could also be scaled to low powered or IoT devices.
- Hardware instructions could be utilized for several mathematical computations, which can potentially improve both the enrolment as well as authentication time.

5.6 Security Analysis

In our proposed work, the security majorly depends on two parameters: the input biometric template B and the secret message Δ of length δ, which is used to generate error correcting codeword W. A significant step involved in the fuzzy commitment scheme is the XOR operation performed between B and W, as shown in (5.1). Therefore, even if one of these parameters is compromised or revealed, it would reveal the original biometric template B to the attacker, compromising the security and privacy of the corresponding user.

- **Brute force complexity on getting the biometric template B:** Assuming the attacker has no better way to get the biometric template other than random guessing, we evaluate the brute force complexity of the attack. While the attacker tries to perform an exhaustive search over the biometric template B provided by the user, the number of trials to get B is

given in terms of the biometric template entropy. Usually, the size of the input biometric template B is very large [4] (with a relatively smaller entropy value). Given the entropy of a biometric template as e, the brute force attack complexity is denoted in terms of the number of trials as 2^e. Since B is 10240 bits in our case, it would have a very high entropy value. Even an iriscode with $|B| = 2048$ bits has an entropy of 249 bits [113], where $|B|$ is the length of biometric template. Thus, it is computationally infeasible to get B using a brute force approach.

- **Brute force complexity on getting the error correcting codeword W:** In the fuzzy commitment scheme, we store the hash of codeword W on the server as a part of helper data. Further, the error correcting codeword is generated once using a secret message Δ, which is not being stored anywhere on the server during the enrolment process. The authentication is successful when $(H(W'') = H(W))$, where $H(W'')$ is the hash of codeword obtained during the authentication phase.

  To compromise W, there could be 2 different approaches: (i) In the first approach, the target would be to obtain W. Given $H(W)$, the attacker could perform pre-image attack on W to satisfy the authentication condition that $(H(W'') = H(W))$. The attack complexity is given by the number of trials required to guess W, which limits to $|H(W)|$ since $|W| > |H(W)|$. It is equal to $2^{|H(W)|}$. (ii) In the second approach, the attacker could perform a brute force attack on the secret message Δ from which the codeword is generated with the number of trials as 2^{δ}, where δ is the length of δ. Therefore, the security bound is given in terms of the number of trials as $2^{min(|H(W)|,\delta)}$. Usually, $\delta \leq |H(W)|$, therefore, attacker chooses to perform brute force on secret message Δ.

  To prevent the attacks given in the two approaches mentioned above, first, the hash output size should be considerably large; for example, we use SHA-256 as the hash function, which gives 256 bits hash value. Second, the secret message bits δ used to generate codewords must be preferably equal to or greater than 128 bits. We further suggest that for every block of the biometric template B_j, with $1 \leq j \leq i$ obtained after splitting the biometric template B into i blocks, the secret message bits used for generating the respective codewords must have lengths $\delta_j \geq 128$ bits.

- **Resistance to Bounding Box Security:** Hao et al. [107] introduce the concept of security based on the sphere-packing bound [115] to calculate the entropy estimation on iriscode as a biometric template. Bounding box security gives the probability of how difficult or easy it is to guess a correct iriscode by an attacker. It follows that the number of trials required by an attacker to guess the correct iriscode is given by

$$M \geq \frac{2^N}{\sum_{i=0}^{t} \binom{N}{i}}$$

where t is the number of bits that can be corrected using error correcting codes and N denotes the uncertainty or degree of freedom of iriscode. It uses the distribution of iriscode bits across a particular dataset to calculate N. Our proposed scheme allows a certain number of errors (t) to be corrected using the random padding. Further, for the iriscode of 10240 bits, the degree of freedom will be high (greater than that of a 2048 bits iriscode with $N = 249$ bits). It could result in lower security bound following the equation of bounding box security. However, in a real-life scenario, the bound is difficult to achieve. It is because to calculate the degree of freedom for a particular iris data, an attacker needs to know how the bits in a genuine user's iris are correlated with one another [107] in a large real-time database, which is not a practical approach.

- **Attack Complexity of the Proposed Work:** Our proposed random-padding scheme prevents the security attack discussed in Section 5.3 where the whole biometric template could be revealed to the attacker, leading to impersonation. The following result justifies the security of our proposed scheme.

  Statement: The security of the overall scheme lies on the security of the δ bits of the secret message Δ, which is used to generate the error correcting codeword W.

  Proof. To compromise the system, an attacker could follow the below two approaches:

  - **Case 1: Guessing δ:** δ is a secret message generated internally by the system. It is assumed to be a uniformly distributed random value. There is no other way to find δ other than a random guess. Therefore, the number of trials required to obtain δ is

equal to the brute force complexity and is given as 2^{δ}, where δ is the length of Δ.

– **Case 2: Guessing padding bits pad in our proposed random padding scheme:** The fuzzy commitment scheme for the $i-th$ block of biometric template following our proposal is shown as

$$B_i \| \{W_{i-1}\}^{pad} \oplus W_i = S_i$$

where pad denotes the number of bits which are padded in the input biometric template B.

The padding bits are secret bits and are random (generated from W, which is generated from Δ). Further, the padding bits are not stored anywhere on the server during enrolment. It is noted that the number of padding bits, $pad \geq \delta$, where δ is the length of secret message Δ used to generate the underlying codeword W and $\delta \geq 128$ bits. If $pad < \delta$ bits, an attacker can perform a brute force approach on the padding bits and can be successful with attack complexity less than 2^{128}.

Since, unlike the fixed padding schemes discussed in Section 5.2, the padding bits are unknown, the attacker aims at guessing at least δ padding bits out of pad number of bits which requires 2^{δ} number of trials to guess the δ bits. If the padding bits are revealed, the attacker could solve the system of linear equations to perform a similar attack, as shown in Section 5.3. If the attack is successful, the original biometric template B could be revealed to the attacker.

To prevent the attack, we consider that the number of padding bits $pad \geq \delta \geq 128$. In case if $pad < 128$, an attacker can perform brute force attack on the padding bits which could lead to the attack shown in Section 5.3. Besides, it is always easy to avoid the case when $pad < 128$. For an example, we have i blocks of biometric template, out of which the padding is required for the a block B_j such that $1 \leq j \leq i$ and $i \geq 2$. The error correcting codeword of a specific length m_j could be chosen by the system for block B_j in a way that from (5.5), we get

$$m_j - n_j \geq 128$$

where n_j is the length of block B_j.

Therefore, the whole system's security depends on the security of δ bits of the secret message, which is equivalent to the brute force attack complexity. It is given in terms of the number of trials as 2^{δ}.

□

5.7 Conclusions

We propose a novel, user-specific, random padding scheme for the efficient implementation of error correcting codes to enhance the security of existing fuzzy commitment schemes. Our proposed padding scheme satisfies the following requirements. Firstly, it ensures that the biometric template's length is equal to the length of error correcting codeword. Secondly, it prevents the existing security attack on error correcting codewords that could reveal the whole biometric template to the attackers. Whereas in the existing approaches, an additional key or a password is commonly used for security enhancement, and to satisfy the requirements mentioned above; we did not use any of these in our proposed scheme. The experimental results show that our proposed padding scheme tremendously improves the baseline's recognition performance, unprotected schemes by approximately 3 times by providing a true match rate of 0.64. The time taken for authentication is around 0.30 seconds on an average of over 200 trials. We provide a security analysis for our proposed work. It shows that our proposed random padding scheme provides the attack complexity of δ bits, equivalent to the brute force attack complexity, where δ denotes the length of secret message used to generate the codeword and $\delta \geq 128$ bits.

We conclude that our proposed scheme significantly improves recognition performance and efficiency in terms of authentication time while preserving the security of the overall system. Other than for user authentication, our scheme could be applied in the areas where the fuzzy commitment scheme is prominently used to generate a secret key from the biometric templates [59, 96]. Since the secret, the cryptographic key generated from a particular instance is unique, it is essential to preserve maximum bit errors in different samples of the same instance.

In such scenarios, implementation of fuzzy commitment with our proposed padding scheme would enhance the error preservation, and hence the performance of the system. Further, the performance accuracy can be significantly increased by considering multiple configurations of the blocks of biometric templates according to the distribution of bit errors in a particular database.

Chapter 6

Conclusion

Motivated by the need to protect the biometric templates in the multi-biometric authentication systems, we design the techniques that help in protecting the biometric templates while preserving the overall recognition performance of the systems. A key insight is a cryptology based approach towards designing such systems.

In the direction of improving the security of multi-biometric biometric cryptosystems, we propose a fusion framework– BIOFUSE, by combining the fuzzy commitment scheme with the fuzzy vault scheme. Format preserving encryption is used to combine the two biocryptosystems. BIOFUSE makes it improbable for an attacker to get unauthorized access to the system without impersonating all the biometric inputs of the genuine user at the same instant. As a consequence, we obtained the overall security bound comparable to the ideal security bound equal to $2^{|K_1|} \times 2^{|K_2|}$. We also observe a 0.98 true match rate at 0.01 false match rate on a virtual IITD-DB1 database.

Though highly secure, the recognition performance in the BIOFUSE ultimately depends on the underlying implementation of the error correcting codes in the underlying biocryptosystems. We propose a novel bit-wise encryption scheme that transforms the biometric template into the protected template, while the key generated from another biometric template is used in the transformation process. The empirical results for measuring efficiency in time units show that for desktop with clock frequency 3.3 GHz, our Algorithm II based approach takes around 12

milliseconds during enrolment and 18 milliseconds during authentication phase on an average of over 200 runs. The bit-wise encryption scheme fully preserves the bit-errors in the biometric templates that enhance the work's applicability on large-scale authentication systems and other domains.

Finally, we work towards improving the fuzzy commitment schemes, particularly from the security and recognition performance aspects. We propose a user-specific, random padding scheme that prevents the impersonation attacks possible in the existing fuzzy commitment schemes with fixed padding approaches. Our proposed scheme provides 3 times better recognition performance with a true match rate of 0.63 on the IIT Delhi iris database comparing to the baseline, unprotected systems. Further, it takes 0.4 seconds for user authentication on an average of over 500 trials. Through security analysis, we show that the attack complexity of our proposed work is 2^k, where k is the length of secret message used to generate codeword, with $k \geq 128$ bits.

The proposed techniques collectively provide a secure way for designing the multi-biometric authentication systems while achieving a justifiable trade-off between the security and the recognition performance.

6.0.1 Future Work

Some future research directions could be explored further.

1. **Design of Large Scale Cancelable Multi-biometric Identification Systems** While our proposed approaches– BIOFUSE in Chapter 3 and the bit-wise encryption in Chapter 4 work mainly in the verification mode, it is an open area of research to improve these approaches to work in the identification mode. Simultaneously, investigating some other data structures, similar to bloom filter-based structures, could be useful in designing the secure cancelable approaches for transforming the biometric templates.

2. **Improving the implementation of fuzzy commitment schemes** Although, from the security perspective, we achieved an improved version of the fuzzy commitment scheme

in Chapter 5, while improving the recognition performance of the system, there is still a scope of enhancing the performance in such systems. It would be particularly beneficial for the scenarios where biometric data size is quite large, such as binarized face template with $76,800$ bits [4].

3. **Use of Deep Learning in Biometric Security** We live in the most exciting time involving deep learning research. Literature related to deep learning techniques for biometric recognition shows a tremendous boon. However, deploying security within the deep learning architecture for generating secure biometric features is an exciting problem to work on. A recent work [101] attains the solution; however, the security, as well as practicality of this work, could be enhanced for deployment at a larger scale. Working on the extraction of secure features from the given biometric template with a deep learning model, particularly for the face characteristics, would be an exciting research area.

References

[1] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *International conference on the theory and applications of cryptographic techniques*. Springer, 2004, pp. 523–540.

[2] C. Rathgeb, F. Breitinger, C. Busch, and H. Baier, "On application of bloom filters to iris biometrics," *IET Biometrics*, vol. 3, no. 4, pp. 207–218, 2014.

[3] M. Gomez-Barrero, J. Galbally, C. Rathgeb, and C. Busch, "General framework to evaluate unlinkability in biometric template protection systems," *IEEE Transactions on Information Forensics and Security*, 2017.

[4] M. Gomez-Barrero, C. Rathgeb, J. Galbally, C. Busch, and J. Fierrez, "Unlinkable and irreversible biometric template protection based on bloom filters," *Information Sciences*, vol. 370, pp. 18–32, 2016.

[5] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *IEEE Transactions on circuits and systems for video technology*, vol. 14, no. 1, pp. 4–20, 2004.

[6] K. Nandakumar and A. K. Jain, "Multibiometric template security using fuzzy vault," in *Biometrics: Theory, Applications and Systems, 2008. BTAS 2008. 2nd IEEE International Conference on*. IEEE, 2008, pp. 1–6.

[7] C. Li, J. Hu, J. Pieprzyk, and W. Susilo, "A new biocryptosystem-oriented security analysis framework and implementation of multibiometric cryptosystems based on decision level fusion," *IEEE transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1193–1206, 2015.

[8] M. Gomez-Barrero, E. Maiorana, J. Galbally, P. Campisi, and J. Fierrez, "Multi-biometric template protection based on homomorphic encryption," *Pattern Recognition*, vol. 67, pp. 149–163, 2017.

[9] M. Gomez-Barrero, C. Rathgeb, G. Li, R. Ramachandra, J. Galbally, and C. Busch, "Multi-biometric template protection based on bloom filters," *Information Fusion*, vol. 42, pp. 37–50, 2018.

[10] A. K. Jain and A. Ross, "Multibiometric systems," *Communications of the ACM*, vol. 47, no. 1, pp. 34–40, 2004.

[11] P. Basak, S. De, M. Agarwal, A. Malhotra, M. Vatsa, and R. Singh, "Multimodal biometric recognition for toddlers and pre-school children," in *2017 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, 2017, pp. 627–633.

[12] J. Daugman, "600 million citizens of india are now enrolled with biometric id," *SPIE newsroom*, vol. 7, 2014.

[13] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of fingerprint recognition*. Springer Science & Business Media, 2009.

[14] J. Daugman, "The importance of being random: statistical principles of iris recognition," *Pattern recognition*, vol. 36, no. 2, pp. 279–291, 2003.

[15] J. Hernandez-Ortega, J. Fierrez, A. Morales, and J. Galbally, "Introduction to face presentation attack detection," in *Handbook of Biometric Anti-Spoofing*. Springer, 2019, pp. 187–206.

[16] N. K. Ratha, J. H. Connell, and R. M. Bolle, "Enhancing security and privacy in biometrics-based authentication systems," *IBM systems Journal*, vol. 40, no. 3, pp. 614–634, 2001.

[17] K. Nandakumar and A. K. Jain, "Biometric template protection: Bridging the performance gap between theory and practice," *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 88–100, 2015.

[18] R. Cappelli, D. Maio, A. Lumini, and D. Maltoni, "Fingerprint image reconstruction from standard templates," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 9, 2007.

[19] J. Galbally, A. Ross, M. Gomez-Barrero, J. Fierrez, and J. Ortega-Garcia, "Iris image reconstruction from binary templates: An efficient probabilistic approach based on genetic algorithms," *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1512–1525, 2013.

[20] K. Simoens, B. Yang, X. Zhou, F. Beato, C. Busch, E. M. Newton, and B. Preneel, "Criteria towards metrics for benchmarking template protection algorithms," in *Biometrics (ICB), 2012 5th IAPR International Conference on*. IEEE, 2012, pp. 498–505.

[21] G. D. P. Regulation, "Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46," *Official Journal of the European Union (OJ)*, vol. 59, no. 1-88, p. 294, 2016.

[22] ISO, "Information technology - security techniques - biometric information protection," International Organization for Standardization, ISO ISO/IEC 24745:2011(en), 2011.

[23] M. Sandhya and M. V. Prasad, "Biometric template protection: A systematic literature review of approaches and modalities," in *Biometric Security and Privacy*. Springer, 2017, pp. 323–370.

[24] C. Rathgeb and A. Uhl, "A survey on biometric cryptosystems and cancelable biometrics," *EURASIP Journal on Information Security*, vol. 2011, no. 1, p. 3, 2011.

[25] I. Natgunanathan, A. Mehmood, Y. Xiang, G. Beliakov, and J. Yearwood, "Protection of privacy in biometric data," *IEEE access*, 2016.

[26] A. Juels and M. Sudan, "A fuzzy vault scheme," in *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*. IEEE, 2002, p. 408.

[27] A. Juels and M. Wattenberg, "A fuzzy commitment scheme," in *Proceedings of the 6th ACM conference on Computer and communications security*. ACM, 1999, pp. 28–36.

[28] K. Xi and J. Hu, "Bio-cryptography," *Handbook of Information and Communication Security*, pp. 129–157, 2010.

[29] C. Rathgeb and A. Uhl, "The state-of-the-art in iris biometric cryptosystems," *State of the art in Biometrics*, pp. 179–202, 2011.

[30] A. B. J. Teoh and J. Kim, "Error correction codes for biometric cryptosystem: An overview," vol. 32, no. 6, pp. 39–49, 2015.

[31] S. Kanade, D. Camara, E. Krichen, D. Petrovska-Delacrétaz, and B. Dorizzi, "Three factor scheme for biometric-based cryptographic key regeneration using iris," in *Biometrics Symposium, 2008. BSYM'08*. IEEE, 2008, pp. 59–64.

[32] F. Hao, R. Anderson, and J. Daugman, "Combining crypto with biometrics effectively," *IEEE transactions on computers*, vol. 55, no. 9, pp. 1081–1088, 2006.

[33] A. Stoianov, "Security of error correcting code for biometric encryption," in *Privacy Security and Trust (PST), 2010 Eighth Annual International Conference on*. IEEE, 2010, pp. 231–235.

[34] M. T. Nguyen, Q. H. Truong, and T. K. Dang, "Enhance fuzzy vault security using nonrandom chaff point generator," *Information Processing Letters*, vol. 116, no. 1, pp. 53–64, 2016.

[35] J. Zuo, N. K. Ratha, and J. H. Connell, "Cancelable iris biometric," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE, 2008, pp. 1–4.

[36] V. M. Patel, N. K. Ratha, and R. Chellappa, "Cancelable biometrics: A review," *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 54–65, 2015.

[37] J. H. Cheon, H. Chung, M. Kim, and K.-W. Lee, "Ghostshell: Secure biometric authentication using integrity-based homomorphic evaluations." *IACR Cryptology ePrint Archive*, vol. 2016, p. 484, 2016.

[38] M. Barni, G. Droandi, and R. Lazzeretti, "Privacy protection in biometric-based recognition systems: A marriage between cryptography and signal processing," *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 66–76, 2015.

[39] K. Nandakumar, A. K. Jain, and S. Pankanti, "Fingerprint-based fuzzy vault: Implementation and performance," *IEEE transactions on information forensics and security*, vol. 2, no. 4, pp. 744–757, 2007.

[40] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and control*, vol. 3, no. 1, pp. 68–79, 1960.

[41] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*. Elsevier, 1977, vol. 16.

[42] J. Black and P. Rogaway, "Ciphers with arbitrary finite domains," in *Cryptographers' Track at the RSA Conference*. Springer, 2002, pp. 114–130.

[43] M. Dworkin, "Recommendation for block cipher modes of operation. methods and techniques," National Inst of Standards and Technology Gaithersburg MD Computer security Div, Tech. Rep., 2001.

[44] D. Chang, M. Ghosh, A. Jati, A. Kumar, and S. K. Sanadhya, "A generalized format preserving encryption framework using MDS matrices," *J. Hardw. Syst. Secur.*, vol. 3, no. 1, pp. 3–11, 2019. [Online]. Available: https://doi.org/10.1007/s41635-019-00065-x

[45] A. K. Jain, K. Nandakumar, and A. Nagar, "Biometric template security," *EURASIP Journal on advances in signal processing*, vol. 2008, p. 113, 2008.

[46] C. Li, J. Hu, J. Pieprzyk, and W. Susilo, "A new biocryptosystem-oriented security analysis framework and implementation of multibiometric cryptosystems based on decision level fusion," *IEEE transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1193–1206, 2015.

[47] P. Mihailescu, "The fuzzy vault for fingerprints is vulnerable to brute force attack," *arXiv preprint arXiv:0708.2974*, 2007.

[48] C. Rathge, A. Uhl, and P. Wild, "Reliability-balanced feature level fusion for fuzzy commitment scheme," in *Biometrics (IJCB), 2011 International Joint Conference on*. IEEE, 2011, pp. 1–7.

[49] A. Nagar, K. Nandakumar, and A. K. Jain, "Multibiometric cryptosystems based on feature-level fusion," *IEEE transactions on information forensics and security*, vol. 7, no. 1, pp. 255–268, 2012.

[50] I. Natgunanathan, A. Mehmood, Y. Xiang, G. Hua, G. Li, and S. Bangay, "An overview of protection of privacy in multibiometrics," *Multimedia Tools and Applications*, vol. 77, no. 6, pp. 6753–6773, 2018.

[51] S. Cimato, M. Gamassi, V. Piuri, R. Sassi, and F. Scotti, "Privacy-aware biometrics: Design and implementation of a multimodal verification system," in *Computer Security Applications Conference, 2008. ACSAC 2008. Annual*. IEEE, 2008, pp. 130–139.

[52] M. Dworkin, "Recommendation for block cipher modes of operation: methods for format-preserving encryption," *NIST Special Publication*, vol. 800, p. 38G, 2016.

[53] I. 24722, "Information technology – biometrics – multimodal and other multibiometric fusion," International Organization for Standardization, Tech. Rep., 2015.

[54] S. Kanade, D. Petrovska-Delacrétaz, and B. Dorizzi, "Obtaining cryptographic keys using feature level fusion of iris and face biometrics for secure user authentication," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*. IEEE, 2010, pp. 138–145.

[55] R. Dwivedi and S. Dey, "Score-level fusion for cancelable multi-biometric verification," *Pattern Recognition Letters*, 2018.

[56] M. He, S.-J. Horng, P. Fan, R.-S. Run, R.-J. Chen, J.-L. Lai, M. K. Khan, and K. O. Sentosa, "Performance evaluation of score level fusion in multimodal biometric systems," *Pattern Recognition*, vol. 43, no. 5, pp. 1789–1800, 2010.

[57] J. Bringer, H. Chabanne, and B. Kindarji, "The best of both worlds: Applying secure sketches to cancelable biometrics," *Science of Computer Programming*, vol. 74, no. 1, pp. 43–51, 2008.

[58] C. Fang, Q. Li, and E.-C. Chang, "Secure sketch for multiple secrets," in *International Conference on Applied Cryptography and Network Security*. Springer, 2010, pp. 367–383.

[59] D. Chang, S. Garg, M. Hasan, and S. Mishra, "Cancelable multi-biometric approach using fuzzy extractor and novel bit-wise encryption," *IEEE Transactions on Information Forensics and Security*, 2020.

[60] M. Gomez-Barrero, C. Rathgeb, J. Galbally, C. Busch, and J. Fierrez, "Unlinkable and irreversible biometric template protection based on bloom filters," *Information Sciences*, vol. 370, pp. 18–32, 2016.

[61] A. Jain and L. Hong, "On-line fingerprint verification," in *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, vol. 3. IEEE, 1996, pp. 596–600.

[62] F. Benhammadi and K. B. Bey, "Password hardened fuzzy vault for fingerprint authentication system," *Image and Vision Computing*, vol. 32, no. 8, pp. 487–496, 2014.

[63] A. Kumar and A. Passi, "Comparison and combination of iris matchers for reliable personal authentication," *Pattern recognition*, vol. 43, no. 3, pp. 1016–1026, 2010.

[64] J. Ortega-Garcia, J. Fierrez, F. Alonso-Fernandez, J. Galbally, M. R. Freire, J. Gonzalez-Rodriguez, C. Garcia-Mateo, J.-L. Alba-Castro, E. Gonzalez-Agulla, E. Otero-Muras *et al.*, "The multiscenario multienvironment biosecure multimodal database (bmdb)," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 1097–1111, 2009.

[65] N. Othman, B. Dorizzi, and S. Garcia-Salicetti, "Osiris: An open source iris recognition software," *Pattern Recognition Letters*, vol. 82, pp. 124–131, 2016.

[66] C. Rathgeb, A. Uhl, P. Wild, and H. Hofbauer, "Design decisions for an iris recognition sdk," in *Handbook of Iris Recognition*. Springer, 2016, pp. 359–396.

[67] L. Masek, "Recognition of human iris patterns for biometric identification," 2003.

[68] A. Anjos, M. Günther, T. de Freitas Pereira, P. Korshunov, A. Mohammadi, and S. Marcel, "Continuously reproducing toolchains in pattern recognition and machine learning experiments," in *International Conference on Machine Learning (ICML)*, Aug. 2017. [Online]. Available: http://publications.idiap.ch/downloads/papers/2017/Anjos_ICML2017-2_2017.pdf

[69] A. Anjos, L. E. Shafey, R. Wallace, M. Günther, C. McCool, and S. Marcel, "Bob: a free signal processing and machine learning toolbox for researchers," in *20th ACM Conference on Multimedia Systems (ACMMM), Nara, Japan*, Oct. 2012. [Online]. Available: https://publications.idiap.ch/downloads/papers/2012/Anjos_Bob_ACMMM12.pdf

[70] VeriFinger, "Neurotechnology verifinger sdk, version: 11.2," www.neurotechnology.com/verifinger.html, Accessed: May 1, 2020.

[71] M. Sudhamani, M. Venkatesha, and K. Radhika, "Fusion at decision level in multimodal biometric authentication system using iris and finger vein with novel feature extraction," in *2014 Annual IEEE India Conference (INDICON)*. IEEE, 2014, pp. 1–6.

[72] Y. Sutcu, Q. Li, and N. Memon, "Secure biometric templates from fingerprint-face features," in *Computer Vision and Pattern Recognition, CVPR'07*. IEEE, 2007, pp. 1–6.

[73] S. C. Dass, K. Nandakumar, and A. K. Jain, "A principled approach to score level fusion in multimodal biometric systems," in *Audio- and Video-Based Biometric Person Authentication*, T. Kanade, A. Jain, and N. K. Ratha, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 1049–1058.

[74] E. C. Shannon, "Error detection and correction using the bch code," 2001.

[75] X. Boyen, "Reusable cryptographic fuzzy extractors," in *Proceedings of the 11th ACM conference on Computer and communications security*. ACM, 2004, pp. 82–91.

[76] R. Canetti, B. Fuller, O. Paneth, L. Reyzin, and A. Smith, "Reusable fuzzy extractors for low-entropy distributions," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2016, pp. 117–146.

[77] J. H. Cheon, J. Jeong, D. Kim, and J. Lee, "A reusable fuzzy extractor with practical storage size: Modifying canetti et al.'s construction," in *Australasian Conference on Information Security and Privacy*. Springer, 2018, pp. 28–44.

[78] C. Rathgeb and C. Busch, "Cancelable multi-biometrics: Mixing iris-codes based on adaptive bloom filters," *Computers & Security*, vol. 42, pp. 1–12, 2014.

[79] P. Drozdowski, C. Rathgeb, and C. Busch, "Bloom filter-based search structures for indexing and retrieving iris-codes," *IET Biometrics*, 2017.

[80] C. Rathgeb, F. Breitinger, H. Baier, and C. Busch, "Towards bloom filter-based indexing of iris biometric data," in *Biometrics (ICB), 2015 International Conference on*. IEEE, 2015, pp. 422–429.

[81] P. Drozdowski, S. Garg, C. Rathgeb, M. Gomez-Barrcro, D. Chang, and C. Busch, "Privacy-preserving indexing of iris-codes with cancelable bloom filter-based search structures," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 2360–2364.

[82] O. Ouda, N. Tsumura, and T. Nakaguchi, "Tokenless cancelable biometrics scheme for protecting iris codes," in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 882–885.

[83] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

[84] K. Zhou and J. Ren, "Passbio: Privacy-preserving user-centric biometric authentication," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 12, pp. 3050–3063, 2018.

[85] M. Inuma, A. Otsuka, and H. Imai, "Theoretical framework for constructing matching algorithms in biometric authentication systems," in *International Conference on Biometrics*. Springer, 2009, pp. 806–815.

[86] A. Rattani and N. Poh, "Biometric system design under zero and non-zero effort attacks," in *Biometrics (ICB)*. IEEE, 2013, pp. 1–8.

[87] "Neurotechnology verieye sdk, version: 11.2," www.neurotechnology.com/verieye.html, Accessed: February 1, 2020.

[88] "Neurotechnology verilook sdk, version: 11.2," https://www.neurotechnology.com/verilook.html, Accessed: February 1, 2020.

[89] R. Raghavendra, K. B. Raja, and C. Busch, "Presentation attack detection for face recognition using light field camera," *IEEE Transactions on Image Processing*, vol. 24, no. 3, pp. 1060–1075, 2015.

[90] R. Raghavendra and C. Busch, "Robust scheme for iris presentation attack detection using multiscale binarized statistical image features," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 4, pp. 703–715, 2015.

[91] B. Toth, *Liveness Detection: Iris*. Boston, MA: Springer US, 2009, pp. 931–938. [Online]. Available: https://doi.org/10.1007/978-0-387-73003-5_179

[92] ISO, "Information technology - trusted platform module library part 1: Architecture," 2015.

[93] J. Hermans, B. Mennink, and R. Peeters, "When a bloom filter is a doom filter: security assessment of a novel iris biometric template protection system," in *Biometrics Special Interest Group (BIOSIG)*. IEEE, 2014, pp. 1–6.

[94] "Aadhaar authentication api specification, version: 2.0," https://uidai.gov.in/images/FrontPageUpdates/aadhaar_authentication_api_2_0.pdf, Accessed: March 5, 2020.

[95] J. Kornblum, "Identifying almost identical files using context triggered piecewise hashing," *Digital investigation*, vol. 3, pp. 91–97, 2006.

[96] D. Chang, S. Garg, M. Ghosh, and M. Hasan, "Biofuse: A framework for multi-biometric fusion on biocryptosystem level," *Information Sciences*, vol. 546, pp. 481–511.

[97] A. B. J. Teoh and J. Kim, "Error correction codes for biometric cryptosystem: An overview," *Information and Communications Magazine*, vol. 32, no. 6, pp. 39–49, 2015.

[98] S. Gao, "A new algorithm for decoding reed-solomon codes," in *Communications, Information and Network Security*. Springer, 2003, pp. 55–68.

[99] M. Malek, "Hadamard codes," *California State University*, vol. 112.

[100] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *Proceedings of ICC'93-IEEE International Conference on Communications*, vol. 2. IEEE, 1993, pp. 1064–1070.

[101] G. Mai, K. Cao, X. Lan, and P. C. Yuen, "Secureface: Face template protection," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 262–277, 2020.

[102] F. NL, "Uk," comparison bose-chaudhuri-hocquenghem bch and reed solomon," *CCITT SGXV, Doc.# 476, Working Party XV/4, Specialists Group on Coding for Visual Telephony*.

[103] K. P. Hollingsworth, K. W. Bowyer, and P. J. Flynn, "The best bits in an iris code," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 6, pp. 964–973, 2008.

[104] S. Kanade, D. Petrovska-Delacrétaz, and B. Dorizzi, "Multi-biometrics based cryptographic key regeneration scheme," in *2009 IEEE 3rd International Conference on Biometrics: Theory, Applications, and Systems*. IEEE, 2009, pp. 1–7.

[105] S. Kanade, D. Camara, D. Petrovska-Delacrtaz, and B. Dorizzi, "Application of biometrics to obtain high entropy cryptographic keys," *World Acad. Sci. Eng. Tech*, vol. 52, p. 330, 2009.

[106] A. Stoianov, "Security of error correcting code for biometric encryption," in *Privacy Security and Trust (PST), 2010 Eighth Annual International Conference on*. IEEE, 2010, pp. 231–235.

[107] F. Hao, R. Anderson, and J. Daugman, "Combining crypto with biometrics effectively," *IEEE transactions on computers*, vol. 55, no. 9, pp. 1081–1088, 2006.

[108] S. G. Kanade, D. Petrovska-Delacrétaz, and B. Dorizzi, "Enhancing information security and privacy by combining biometrics with cryptography," *Synthesis Lectures on Information Security, Privacy, and Trust*, vol. 3, no. 1, pp. 1–140, 2012.

[109] S. Chauhan and A. Sharma, "Improved fuzzy commitment scheme," *International Journal of Information Technology*, pp. 1–11, 2019.

[110] T. Hoang, D. Choi, and T. Nguyen, "Gait authentication on mobile phone using biometric cryptosystem and fuzzy commitment scheme," *International Journal of Information Security*, vol. 14, no. 6, pp. 549–560, 2015.

[111] P. Li, X. Yang, H. Qiao, K. Cao, E. Liu, and J. Tian, "An effective biometric cryptosystem combining fingerprints with error correction codes," *Expert Systems with Applications*, vol. 39, no. 7, pp. 6562–6574, 2012.

[112] C. G. Cullen, *Matrices and linear transformations*. Courier Corporation, 2012.

[113] J. Daugman, "How iris recognition works," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 21–30, 2004.

[114] S. Lin and D. J. Costello, *Error control coding*. Prentice hall, 2001, vol. 2, no. 4.

[115] H. Al-Assam and S. Jassim, "Security evaluation of biometric keys," *computers & security*, vol. 31, no. 2, pp. 151–163, 2012.