FUZZY VAULT SCHEME FOR FINGERPRINT VERIFICATION: IMPLEMENTATION, ANALYSIS AND IMPROVEMENTS

by Cengiz Örencik

Submitted to the Graduate School of Sabancı University
in partial fulfillment of the requirements for the degree of
Master of Science

Sabanci University

July, 2008

FUZZY VAULT SCHEME
FOR FINGERPRINT VERIFICATION:
IMPLEMENTATION, ANALYSIS AND IMPROVEMENTS

APPROVED BY:

Assoc. Prof. Dr. Erkay Savaş	....
(Thesis Supervisor)	
Assist. Prof. Dr. Thomas Brochmann Pedersen	....
Assoc. Prof. Dr. Mehmet Keskinöz	....
Assoc. Prof. Dr. Albert Levi	....
Assist. Prof. Dr. Cem Güneri	....

DATE OF APPROVAL:........................

FUZZY VAULT SCHEME FOR FINGERPRINT VERIFICATION: IMPLEMENTATION, ANALYSIS AND IMPROVEMENTS

Cengiz Örencik

CS, Master's Thesis, 2008

Thesis Supervisor: Erkay Savaş

Keywords: Fuzzy Vault, Biometrics, Fingerprint, Privacy

Abstract

Fuzzy vault is a well-known technique that is used in biometric authentication applications. This thesis handles the fuzzy vault scheme and improves it to strengthen against previously suggested attacks while analyzing the effects of these improvements on the performance.

We compare the performances of two different methods used in the implementation of fuzzy vault, namely brute force and Reed Solomon decoding with fingerprint biometric data. We show that the locations of fake (chaff) points leak some valuable information and propose a new chaff point placement technique that prevents that information leakage. A novel method for chaff point creation that decreases the success rate of the brute force attack from 100% to less than 3.3% is also proposed in this work.

Moreover, a special hash function that allows us to perform matching in the hash space which protects the biometric information against the 'correlation attack' is proposed. Security analysis of this method is also presented in this thesis. We implemented the scheme with and without the hash function to calculate false accept and false reject rates in different settings.

PARMAKİZİ İÇİN FUZZY VAULT SİSTEMİ: UYGULAMA, ANALİZ VE GELİŞTİRMELERİ

Cengiz Örencik

CS, Yüksek Lisans Tezi, 2008

Tez Danışmanı: Erkay Savaş

Anahtar Kelimeler: Fuzzy Vault, Biometrikler, Parmakizi, Gizlilik

Özet

Fuzzy vault sistemi, biyometrik tabanlı kimlik onaylama sistemlerinde kullanılan bilinen bir tekniktir. Bu tezde, fuzzy vault sistemini temel alarak, bu sistemin daha önce önerilmiş saldırılara karşı güvenliliğini arttıran yenilikler öneriyoruz ve bu yeniliklerin performansa etkilerini inceliyoruz.

Fuzzy vault uygulamalarında kullanılan, kaba kuvvet ve Reed Solomon kod çözme isimli iki metodu, parmakizi biyometrik verisini kullanarak karşılaştırdık. Vault üzerindeki taklit noktaların yerlerinin, nemli bilgi açığa çıkardığını gösterdik ve bu bilgi sızıntısını engelleyen yeni bir taklit nokta yerleştirme methodu önerdik. Ayrıca kaba kuvvet saldırısının başarı oranını %100'den %3.3'e düşüren yeni bir taklit nokta yaratma methodu önerdik.

Bunların haricinde, karşılaştırmayı hash alanında yapmayı mümkün kılan özel bir hash fonksiyonu önerdik. Bu methodla, biyometrik bilgisini 'ilişki kurma' saldırısına karşı güvenli hale getirdik ve bu methodun güvenlik analizlerini yaptık. Ek olarak, bu sistemin uygulamasını hash fonksiyonu içeren ve içermeyen değişik ayarlarda yaparak hatalı kabul ve hatalı ret oranlarını hesapladık.

Acknowledgements

I wish to express my gratitude to my supervisor Erkay Savaş, for his invaluable guidance, support and patience all through my work. I am also grateful to Thomas Pedersen, for his valuable contributions to this thesis.

I would like to thank Mehmet Keskinöz for all his support and guidance.

Special thanks to my friends Eren Çamlıkaya, for his help in the coding part and Alisher Kholmatov for his valuable remarks about fuzzy vaults.

I am indebted to the members of the jury of my thesis: Thomas Pedersen, Mehmet Keskinöz, Albert Levi and Cem Güneri for reviewing my thesis and for their useful remarks.

I am grateful to TÜBİTAK (The Scientific and Technical Research Council of Turkey), for the M.Sc. fellowship supports.

Especially, I would like to thank my family, for being there when I needed them to be. This work would never have been possible without their support.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

In this age of universal connectivity, with hackers and electronic fraud, user authentication has become a very crucial matter. The new developments in biometric technology provides us the tools for authentication that can protect our identity from being stolen. The aim of this research is to improve the *fuzzy vault* [3] scheme to strengthen it against previously proposed as well as new attacks while analyzing the effects of these improvements on the performance. The research also involves comparison of the efficiencies of the two decoding methods, namely brute force and Reed Solomon, that is used in the implementation of *fuzzy vault*.

1.1 Authentication Factors

Identification for access control and other purposes can be achieved by utilizing three factors:

1. What you know (e.g. passwords)
2. What you have (e.g. smartcards)
3. What you are (biometric data identifying a person)

These are called the *three pillars of authentication* [4]. Either these factors can be used alone or any combination of the three can be used together to increase security and compensate the weaknesses of one factor. The first factor can be anything that needs to be remembered to prove your identity such as passwords or PINs. Even though the passwords are the most common way of authentication, they have many drawbacks: They can be forgotten, stolen, shared or guessed. The second factor can be any unique token which is registered to the user who needs to possess it for authentication. There are two kinds of tokens:

1. Storage tokens
2. Dynamic tokens

Storage tokens have a unique information that identifies its user. They are usually used together with passwords. A common example for this kind of authentication is the ATMs where the card (storage token) and PIN (password) is used together for authentication. This system provides better security then something you know since they can not be shared or guessed, but still the token and the associated password can be stolen. Different from the storage tokens, dynamic tokens generate a one-time authentication code. This code is usually in the form of a challenge sent from the computer and the response from the token. The dynamic tokens are usually used together with passwords so this way of authentication still requires a password that can be forgotten and a token that can be stolen.

Biometric is used as the third factor for authentication. A biometric is inseparable from an individual and always accessible providing comparably high level of security. In addition, it can easily be combined with other factors to increase security further. Biometric identification, on the other hand, also suffers from two major drawbacks:

1. The noisy nature of biometrics measurement process
2. Privacy issues due to the fact that biometric data reveals private information about the individuals which is not intended to be revealed

otherwise

The latter concern is nowadays becoming more and more important and authorities are in the process of taking measures to protect the privacy of individuals (e.g. Australian Biometrics Institute Privacy Code).

1.2 Biometrics and Fingerprint

Biometrics have their own terminology that is used throughout this thesis. The basic terminology for biometrics, specifically fingerprint biometric is explained in this section. A biometric is a physical or psychological feature that can be measured and quantified. This quantified feature can be used to authenticate a person with a degree of certainty by comparing different measurements of this feature. Clearly the degree of certainty depends on the type and quality of the biometric and the authentication algorithm used.

Fingerprint biometrics was one of the first biometrics that is used for identification and authentication purposes. It is still widely used in many areas and people accept that fingerprints are unique and can be used for identification. Since it is widely used, it is crucial to have a secure fingerprint authentication system.

Generally macro and micro features are used to identify a fingerprint image [4]. Macro features can be seen with the naked eye but to see the micro features, a sensor device is necessary. The macro features are used as helper data [5] for fingerprint authentication but the minutia points that is mainly used in fingerprint authentication are identified by the micro features.

The most common macro features are ridge patterns as illustrated in Figure 1, core point (center point of a fingerprint) and maximum curvature points. On the other hand, common minutia points (i.e. micro features) are ridge ending, ridge bifurcation and dot (or island) as illustrated in Figure 2. Some of the main macro and micro features are marked in Figure 3.

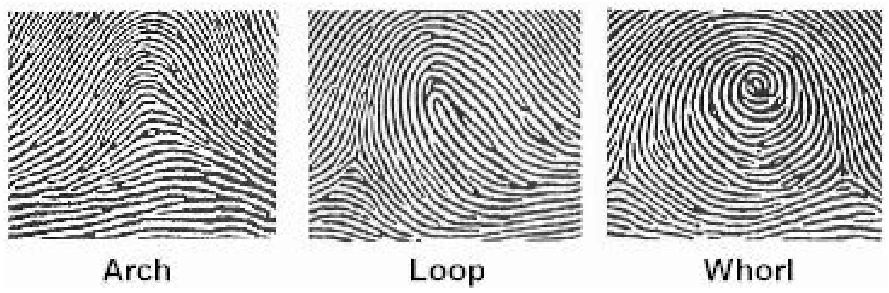


Figure 1: Ridge Patterns [1]

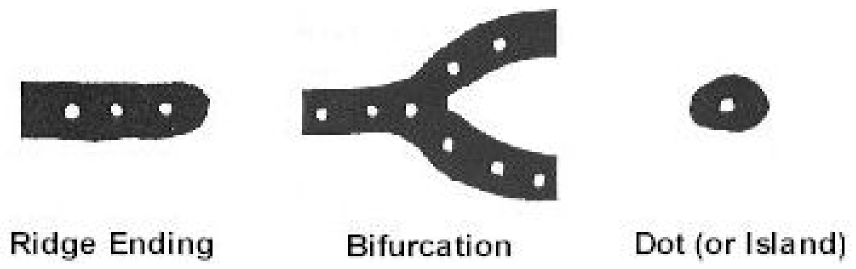


Figure 2: Micro Features [1]

1.3 Contributions of the Thesis

In this thesis, we focus on several issues involving *fuzzy vault* implementation for biometrics usage. The first issue is to compare the computational efficiencies of the two methods, namely brute-force and Reed Solomon (RS) decoding methods. Another issue we deal with is to provide a step-by-step guideline for the implementation details of *fuzzy vault* schemes, which has not

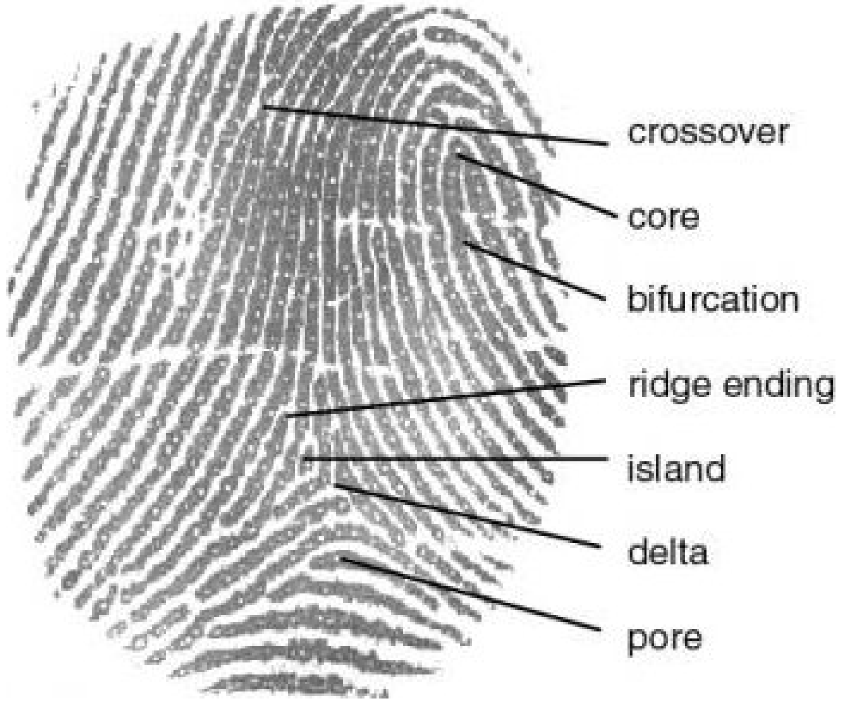


Figure 3: Macro Micro Features [2]

been given in previous works. We also analyze some security drawbacks of the *fuzzy vault* scheme and propose solutions to those weaknesses as outlined below:

- Kholmatov et al. [6] showed that it is possible to link an unknown vault to another vault that is constructed by the same biometric by applying the correlation attack which is explained in Section 6.3. We propose keeping hash values of the minutia points, instead of the minutia points themselves. The details of this proposed method is explained in Section 8.2 together with the security analysis.

- The locations of the points in the vault may reveal some information as to which points are genuine depending on the chaff point generation. We propose a method in Section 7.1 that makes distinguishing genuine points impossible.

- Mihailescu [7] pointed out that the *fuzzy vault* scheme is vulnerable to brute force attack. We propose a new method in Section 7.2 to decrease the success rate of this attack from 100% to less than 3.5%. This countermeasure proves to be useful in certain settings.

- We study the effects of distances between chaff points and between a chaff and a genuine point on the security and performance of the *fuzzy vault*.

- We also study limitations on the vault size and its effects on the security and performance of the *fuzzy vault*.

1.4 Organization of the Thesis

The rest of the thesis is organized as follows:

Section 2 presents the previous works on the *fuzzy vault* scheme and briefly explains the principles of the *fuzzy vault* scheme. Shamir's secret sharing system which has a crucial importance in the *fuzzy vault* scheme is also explained in this section.

In Section 3, a review of the *fuzzy vault* authentication scheme is given and the details of enrollment, verification and alignment stages of this scheme are explained in detail. We also mention different alignment methods where any of them can be used in this authentication system.

Section 4 explains the implementation details of the brute force and Reed Solomon decoding algorithms [8] used for reconstructing the authentication data hidden in the *fuzzy vault*. We define the Generalized Reed Solomon codes which are used for reconstructing the secret polynomial. The decoding algorithm for Generalized Reed Solomon codes that we used in our algorithm is also presented in this section.

In Section 5, a comparative analysis for the performance of two techniques used in polynomial reconstruction is provided.

In Section 6, we propose an attack called Location Based Attack that the original fuzzy vault system is vulnerable. Also, the two previously proposed attacks targeted on the *fuzzy vault* system, namely the brute force attack and the correlation attack are visited in this section.

Section 7 outlines two proposed modifications to the enrollment stage to increase the security of the *fuzzy vault* and summarizes the security analysis of the scheme against brute force attack. From a given vault, the first modification makes distinguishing genuine points impossible while the second modification strengthen the scheme against brute force attack.

In Section 8, we propose keeping hash values of the minutia points instead of the minutia points themselves. We introduce the requirements, a hash function should satisfy to be used in a secure *fuzzy vault* scheme. We propose a special hash function and present proofs that our proposed hash function satisfies the necessary requirements.

Section 9 explores the effects of the vault and threshold sizes and use of the proposed hash function on the security and fault rates of the scheme using experimental data. It also provides a timing comparison between brute force and RS decoding methods.

And finally Section 10 is devoted to our conclusions and the summary of the thesis.

2 Literature Survey

One of the first biometric authentication systems that uses cryptographic techniques is proposed by Juels and Wattenberg [9] called the *fuzzy commitment* scheme. Different from traditional cryptographic techniques, this scheme does not require an exact match with the decryption key but a reasonably close key is sufficient for decryption. In this method a secret is hidden under a key x and the user can reveal the secret given any key x' that is close to x in terms of Hamming distance. The minutia points of a fingerprint can be used to construct x. Although the method tolarates some errors in the information symbols of x, it can not tolarate re-ordering of the symbols which is called the *order-invariance* property. Soutar et al. [10] also proposed an algorithm that binds a large cryptographic key with the user's fingerprint image using enrollment. Given the same fingerprint, the key can be revealed by using correlation filter functions. This scheme overcomes the order invariance problem but with a highly inefficient method.

Juels and Sudan [3] proposed the so-called *fuzzy vault* scheme that overcomes the order invariance problem in an efficient way. The main idea is to exploit the relationship between error correction and secret sharing — the biometric data together with a *secret* defines a codeword from an appropriate error correction code. Given the fingerprint, the codeword can be corrected, and the secret is extracted. However, the secret does not reveal anything about the biometric data. If the secret is compromised, one can always choose another secret to combine with the same biometric. The main idea is that the biometric data is essentially used to extract a *secret* hidden in the coefficients of a secret polynomial. The method for reconstructing the secret polynomial is based on the Shamir's threshold secret sharing scheme [11] which utilizes polynomial evaluations at minutiae points. Shamir's secret sharing scheme is briefly explained in Section 2.1.

Later, Clancy et al. [12] used this *fuzzy vault* scheme in a secure smartcard system. They used Reed-Solomon decoding to construct the secret polynomial. The authors provide realistic expectations on the values of the security parameters and associated attack complexity. They claim that the scheme provides 69 bits security against a brute force attack but with the parameters that provides this security, the error rates increase between 20 to 30 percent. In 2004, Dodis et al. [13] propose a modification to the original *fuzzy vault* scheme. They used a second polynomial p' where the degree of p' is higher than p which overlaps with p only for the genuine minutia points. They represent the vault only using the coefficients of p' without using locations of chaff or genuine points.

The codeword can be corrected by using brute force; but using more sophisticated Reed-Solomon (RS) decoding method is usually assumed to be more efficient [3], [12]. Though it is well known that the RS decoder performs better than brute force *asymptotically*, it still remains to be verified whether the brute force method or the RS decoding method performs better for *fuzzy vaults* in practical implementations.

A successful application of *fuzzy vault* to fingerprint biometrics is due to [14] that basically uses the brute force approach. Different from Clancy's work they used alignment help-data which decreases the error rates, and also a Cyclic Redundancy Check (CRC) embedded in a coefficient of the secret polynomial is used to guarantee that the correct polynomial is found.

2.1 Shamir's Secret Sharing Scheme

In a (k, n) threshold secret sharing scheme, a secret S is divided among n people such that any coalition of k people can successfully reveal the secret S. Furthermore, a secret sharing mechanism is said to be perfect if a coalition of $k-1$ people cannot even reduce the candidate space to find the secret S. Shamir's method of interpolation of the secret polynomial is perfect since it satisfies this property [11].

The method, firstly, requires that a polynomial $f(x) = a_{k-1}x^{k-1}+a_{k-2}x^{k-2}+\ldots+a_0$ of degree $k-1$ be generated in $Z_q[x]$ where q is a prime number that satisfies $q > k$ and $\forall i\ a_i < q$. In the original Shamir's method the secret is the constant coefficient a_0 of the polynomial; however, in *fuzzy vault* implementations the secret is the concatenation of all coefficients of the polynomial (i.e. $S = a_{k-1}||a_{k-2}||\ldots||a_0$). The share of the i^{th} party is $y_i = f(x_i)$, for values $1 \leq i \leq n$ where n is the number of secret shares. If k parties come together, they can construct the polynomial and learn the secret; a coalition of less than k parties naturally cannot reveal the secret[1].

Let us assume that an attacker captures $k-1$ shares of the n secret shares. For each candidate value S' where $0 \leq S' < q$, the attacker can construct a different polynomial where each of these polynomials (i.e. the secret S) are equally likely. Therefore, the attacker can learn nothing about the actual value of the secret S from the $k-1$ shares he captured.

[1]In the original Shamir's secret sharing where the secret is the constant coefficients no information can be gathered about the secret by a coalition of less than k parties. Thus, the original scheme provides information theoretic security while the security properties of the *fuzzy vault* are yet to be determined.

This (k, n) threshold secret sharing scheme is quite efficient when it is used with good polynomial interpolation algorithms such as *Lagrange interpolation* [15]. Moreover, this scheme has other useful properties such as:

- When the value k (degree of the secret polynomial) is kept fixed, any number of new secret share can be added or deleted without effecting any of the other secret shares.
- By using this secret sharing scheme, a hierarchical scheme can be established where more important share holders have more secret shares according to their rank in the hierarchical structure. For example, the president of a company may have five shares, the vice president may have 3 shares and other workers may have a single share. Then a (6,n) threshold scheme can be enabled either by two workers, one of whom is the president or by four workers, one of whom is vice president or by any six workers. Although, this is a very important property, it is not useful in the context of *fuzzy vaults.*

2.2 Error Correction and Detection

An error correction code $\mathbf{C}$ over a finite alphabet $\mathbf{F}$ is called an (m, M, d) code where m is the code length, M is the code size (i.e. number of all possible codewords) and d is the minimum *Hamming distance* between two codewords $c \in \mathbf{C}$ [8].

Given an (m, M, d) code $\mathbf{C}$ over $\mathbf{F}$, let $c \in \mathbf{C}$ be the original codeword and y be the received word. An error is defined as the event of changing an entry in c and the error locations are the indexes of these entries. Error correction decoders find the error locations and error values as long as the number of errors is less than a threshold τ. In this work the error correction codes we use can recover up to $\tau = (d-1)/2$ errors. Different from error correction decoders, error detection decoders only indicate the error locations, without attempting to correct them.

3 Review of Fuzzy Vault

Juels and Sudan [3] proposed a scheme called *fuzzy vault* for secure biometric authentication. The identification process using the *fuzzy vault*, consists of two major stages: the enrollment and verification. The scheme is fuzzy since the secret polynomial can be reconstructed even when the list of minutia points of the enrolled and measured fingerprints are not exactly the same.

The biometric identification problem can be stated using an analogy with Alice and Bob as described in [3]. The famous example is that Bob wants to know Alice's phone number; but Alice will give him the number only if their taste of films matches to a certain amount. Let A denote the list of Alice's favorite films and B denote the list of Bob's favorite films. An important factor here is that the lists of favorite films are unordered sets. Alice publishes her set A along with other random films which are not in the set A resulting in a much bigger set A'. If Bob's list B matches certain number of films in the set A' which are also members of set A, Bob will correctly receive Alice's phone number.

The scheme presented above is a direct analogy of the biometric verification process with *fuzzy vault*. In this section we give a brief outline for the techniques used in the application of *fuzzy vault* scheme to fingerprint biometrics. As mentioned above, the identification process using the *fuzzy vault* consists of two major stages: the enrollment and verification. In the enrollment stage, the *fuzzy vault* is created by embedding a secret polynomial after the fingerprint of the user is obtained. The *fuzzy vault* hides the fingerprint and the secret polynomial which can be revealed if the same finger is used in verification. The verification stage contains two phases: 1) the alignment of the measured fingerprint to the points in the *fuzzy vault*, and 2) the reconstruction of the secret polynomial. The enrollment and alignment stages are the same for both brute force and RS decoding methods that differ in the polynomial reconstruction phase.

The two stages, namely enrollment and alignment, are described briefly in the following sections and the Section 4 is devoted to the details of polynomial reconstruction phase. Note that these stages outline our implementation and may differ from other *fuzzy vault* implementations.

3.1 Enrollment Stage

During the enrollment stage, expectedly n minutiae points from a fingerprint are presented to the system. Two coordinates of the n genuine minutiae points, (x_i, y_i) are concatenated to form integers $\overline{x_i} = x_i||y_i$. Each coordinate of the minutia point is a w-bit number and thus the resulting number $\overline{x_i}$ is of length $2w$-bit. The numbers $\overline{x_i}$s form the minutiae space in which random chaff points are also created. Then, chaff points are added to the vault such that there are a total of C points with inter-Euclidean distance greater than a threshold t. A *fuzzy vault* with C points is assumed to be accessible to anyone including an external attacker.

Now, a $2kw$-bit secret key S used for identification is equally divided into k parts and each part is embedded as one coefficient of a secret polynomial $p(x)$ over $Z_{q^2}[x]$ of degree $k-1$ where q is a w-bit integer (Figure 4(a)). Since the secret polynomial has degree $k-1$, k points that lie on this polynomial are sufficient to successfully reconstruct the polynomial .

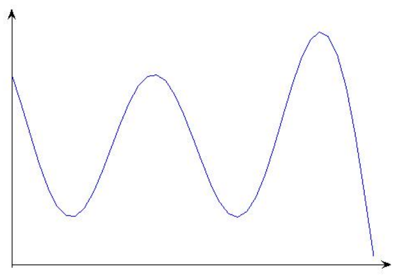
For each $2w$-bit number $\overline{x_i}$ formed from the concatenation of coordinates of a minutiae point, the secret polynomial $p(x)$ is evaluated in $\pmod{q^2}$ and $\overline{y_i} = p(\overline{x_i})$ is obtained for each genuine point (Figure 4(b)). Then chaff points are picked at random. And finally, these chaff points are placed in the *fuzzy vault* which is a two dimensional vector space (Figure 4(c)). In other words, for each randomly chosen $2w$-bit chaff point $\overline{x}$, a randomly chosen $\overline{y}$ coordinate of the same size is added. Naturally, while $\overline{y}$ coordinates of genuine points lie on the secret polynomial, $\overline{y}$ coordinates of chaff points do not.

Since the vault contains many more chaff points than genuine points it is computationally expensive to reconstruct the secret polynomial without knowing the original biometric data. The steps required for the enrollment stage are illustrated in Figure 4 and the block diagram of the original *fuzzy vault* enrolling scheme is shown in Figure 5

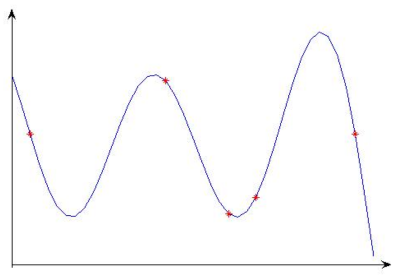
In the implementation of [14], cyclic redundancy check (CRC) of the secret S is added to the secret polynomial as a coefficient to guarantee that the correct polynomial is found in the verification stage, since the polynomial reconstruction methods may return an incorrect polynomial. However, in our implementation we instead check if there are at least $k + \mu$ vault points lie on the polynomial, for some $\mu > 1$, to guarantee that the correct polynomial is found. In our tests we see that both methods give the same False Accept Rate (FRR) and False Reject Rate (FAR) results (Section 9).

3.2 Verification Stage

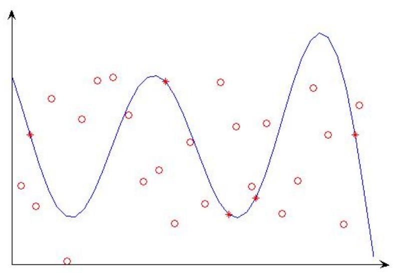
The goal of the verification stage is to reconstruct the secret polynomial from the genuine biometric data, which is used to recover the secret key



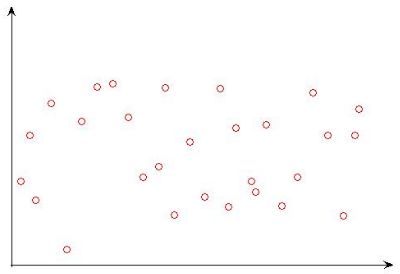
(a) Create secret polynomial



(b) Project elements x_i onto polynomial



(c) Create random chaff points



(d) Vault

Figure 4: Fuzzy Vault Scheme for Enrollment

S. The recovered secret key is then used for identification. When a user presents a genuine fingerprint for identification, an average of m minutiae points are expected to match the points in the vault, where $m \leq n$. The fuzziness comes from the fact that the person does not have to present the same set of minutiae points for each verification process. This is especially a useful feature since the fingerprint measurement is a noisy process and in each verification a different set of measured minutiae points match the points in the *fuzzy vault*. The block diagram of the original *fuzzy vault* verification scheme is shown in Figure 6.

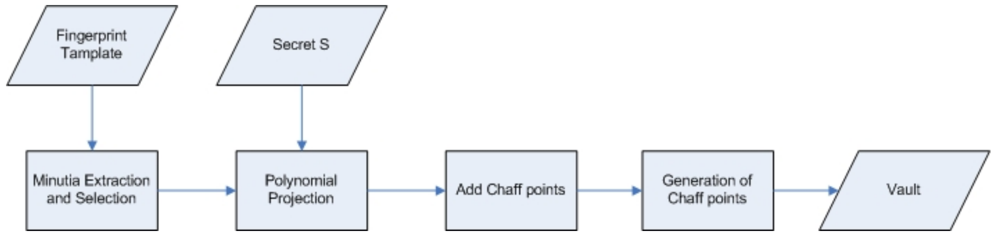


Figure 5: Block Diagram For Enrollment

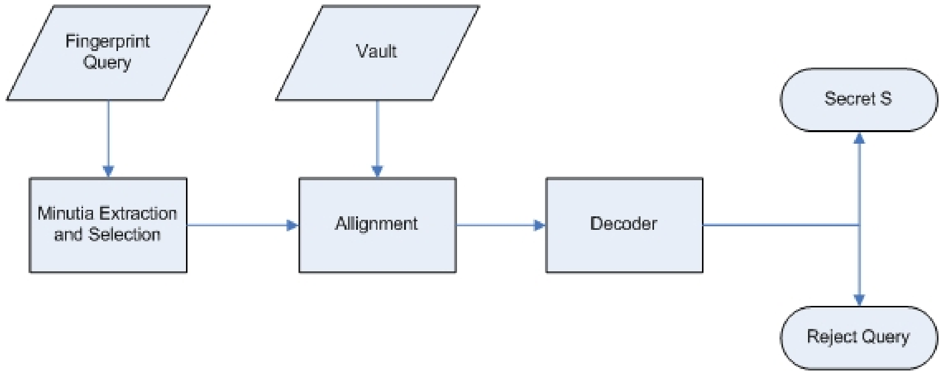


Figure 6: Vault Verification

3.3 Alignment Phase

The verification process of the fingerprint presented to the system should undergo some preprocessing before applying the polynomial reconstruction algorithm. Preprocessing stage is mainly the alignment of the query fingerprint to the enrolled fingerprint stored during the enrollment phase. There are different methods that do the alignment of the query fingerprint to the enrolled fingerprint and the most commonly used minutia alignment methods are explained in this section.

At the end of the alignment phase, matching points of the query fingerprint images, consisting of some genuine and some chaff points, are presented to the system for verification. This list is known as *verification list* and when it contains at least $k + \mu$ points that matches to genuine points, the secret polynomial can be reconstructed.

Without the alignment process, the false reject rates will be quite high since the biometric data varies greatly in different measurements due to imperfections of the process. If the query fingerprint is genuine, the verification list is mainly composed of genuine points from the enrollment phase with a small number of chaff points.

Aligning two fingerprints is a difficult task and errors in this phase could lead to false rejects. There are several different approaches for fingerprint alignment and the most commonly used ones are as follows:

- by using reference points
- by using helper data
- by exhaustive search

3.3.1 Alignment by Reference Points

Yang and Verbauwhede [16], constructed an automatic secure fingerprint verification system based on the *fuzzy vault* scheme where the most reliable reference points are chosen from the enrolled and query templates and aligned in the alignment phase. There is a high noise due to shifting and rotation on the position of minutia points that are obtained by a fingerprint sensor. Yang and Verbauwhede overcame this problem by observing the minutia points in the Polar coordinate system instead of observing in the Cartesian coordinate system. By choosing the origin of the Polar coordinate system correctly, they can obtain a system independent of the translation and rotation of the input fingerprint images. They used a rotation and translation invariant that is a function of r, θ and φ where r is the distance between two minutia, θ is the position angle, and φ is the direction difference between a minutia and the origin. Assume the local feature vectors of the i^{th} minutia of the fingerprint A and j^{th} minutia of the fingerprint B are given as $M_A(i)$ and $M_B(j)$ respectively. Their similarity level is calculated with the following formula:

$$s(i,j) = \begin{cases} 1 - \frac{|M_A(i)-M_B(j)|_W}{T(W)}, & if\ |M_A(i) - M_B(j)|_W < T(W) \\ & 0, \text{otherwise} \end{cases}$$

where $|M_A(i) - M_B(j)|_W$ is the weighted distance between two local feature vectors, W is a weight vector and $T(W)$ is a fixed threshold related to this weight vector.

The algorithm calculates all $s(i,j)$ values and choose the pair with the largest similarity level as referance pair. Then the minutia points are converted in a polar system. The polar coordinates of the query fingerprint is used in the verification phase. Though this alignment based on reference point is computationally efficient, finding a reliable point requires at least 3 templates during enrollment of a fingerprint and still errors may occur that leads to false rejects. To avoid that problem, an additional information from the fingerprint, called the helper data, can be used in the alignment phase.

3.3.2 Alignment by Helper Data

Uludag et al. [5] implemented a *fuzzy vault* system that uses helper data that is automatically extracted from the fingerprints, later Nandakumar et al. [17] used helper data that is constructed in the same way as Uludag for the alignment phase of *fuzzy vault*. They used the Orientation Field Flow Curves (OFFC) [18] since they are robust to noise. First an orientation field is set and a flow curve is found where an orientation field flow curve is a set of linear segments whose tangent direction is parallel to the orientation field direction at each point. The set of flow curves is found by calculating many flow curves each with a different starting point and from each curve, the point that has the maximum curvature value is found. The helper data is composed of these points with maximum curvature values. In this method the helper data must be constructed both for the enrolled fingerprint and for the query fingerprint.

After the helper data is extracted from the fingerprint, the points with very high and very low curvature are filtered out that gives the final version of the helper data. In the alignment phase, the helper data extracted from the enrolled fingerprint (H_E) and the one extracted from query fingerprint (H_Q) are aligned with each other by using an Iterative Closest Point (ICP) based algorithm [19]. The details of the alignment algorithm is presented in [17].

Note that the helper data is kept as public information, therefore, it should not reveal any information about the minutia points that might compromise the security. The maximum curvature points are macro features of a fingerprint that are independent from the minutia points which are micro features as previously explained in Section 1.2. Therefore, the authors claim that helper data should not leak any information about the minutia attributes of a fingerprint. However, this information can still be used to decrease the search area of the vault and make the system even more vulnerable to brute force attack which is explained in Section 6.2.

3.3.3 Alignment by Exhaustive Search

The exhaustive search method does not require any helper data or reference points but only uses the minutia location coordinates of the fingerprints. All the translation in x and y coordinates plus the rotation variants of the points of query fingerprint are compared with the points in the vault. If a point in the query fingerprint is closer to a template point than certain number of pixels in the image we assume that the two points are matched; therefore the tested point is stored as the matching point. This process is repeated for many different combinations of translated and rotated fingerprint images and the combination with maximum number of matching points are the output of the alignment phase.

The exhaustive search method is not an efficient method compared to the other two methods but it can make quite accurate alignment. The False Accept and False Reject Rates given in Section 9 are calculated by using this alignment method. However, all the contributions given in this thesis are independent of the alignment method and any alignment method (i.e. matching algorithm) can be used instead of the currently used exhaustive search method to obtain faster matching results.

We devote the next section to the polynomial reconstruction method, for which two methods (i.e. brute force and Reed-Solomon decoding) can be applied.

4 Polynomial Reconstruction Phase

There are two methods employed to reconstruct the secret polynomial, namely brute force and Reed-Solomon decoding. Both methods essentially apply the Shamir's secret sharing scheme [11] in the reconstruction of the secret polynomial. The shares of the secret S correspond to the genuine points in the vault which are not discernible among the many chaff points (fake shares). A genuine fingerprint reveals sufficient number of genuine shares and these shares are used in the polynomial reconstruction methods to recover the secret polynomial.

Note that the *fuzzy vault* does not exactly provide the same type of security as Shamir's secret sharing scheme. Shamir's scheme provides information theoretic security since the secret in this scheme is embedded as only a single coefficient of the secret polynomial while in the *fuzzy vault* the secret is the concatenation of all coefficients.

4.1 Brute Force Approach

To reconstruct the secret polynomial using brute force approach requires trying many of the combinations of k out of given m matching points. Note that some of the m matching minutiae points are the ones that actually match random chaff points in the *fuzzy vault.* When k minutiae points that match the real minutiae points are found during the exhaustive search, the scheme is said to be successful.

In brute force approach, first k pairs of $(\overline{x_i}, \overline{y_i})$ are chosen randomly from the verification list and the polynomial on which the selected k pairs lie is calculated using Lagrange interpolation method. Then whether μ of the remaining vault points satisfies $\overline{y_i} = p(\overline{x_i})$ is tested. If more points that lie on the same polynomial are found, the fingerprint is verified; otherwise rejected. If insufficient number of pairs satisfy $\overline{y_i} = p(\overline{x_i})$ condition, another random k pairs are taken as input and the process is repeated. The maximum number of trials is set to a high value, after which the program rejects the fingerprint if no polynomial satisfying the condition is found. The drawback of the brute force approach is high computation complexity when the query fingerprint is too noisy which cause lower genuine matches and higher chaff matches.

4.2 Reed Solomon Decoding Approach

Utilizing error correcting codes for the implementation of *fuzzy vault* is first proposed by [3]. The authors state that after matching minutiae points are obtained, use of Reed-Solomon (RS) decoder is a more efficient approach than the brute force. The Reed-Solomon decoders have an error correction capability of $\tau = \frac{m-k}{2}$ errors. Even though the Guruswami-Sudan list decoding algorithm [20] [21] can correct errors beyond this limit ($\tau_{GS} = \sqrt{mk}$), it is not suitable to our case due to efficiency reasons. The best choice to implement RS decoder is to use the Berlekamp-Massey (BM) algorithm as explained also in [12] since it is fast, easy to implement and widely studied. Moreover, for the parameters used in the *fuzzy vault* scheme the error correction capacity of Gruswami-Sudan's algorithm is very close to Berlekamp-Massey's algorithm.

However, details of the RS decoding method, are given neither in [3] nor in [12]. These papers mention the decoder, named UNLOCK function, as a black box that reconstructs the secret polynomial given the m matching points. The lack of detailed description of the method caused some misunderstanding in literature; for instance [22] claim that Reed-Solomon decoding is not applicable in the case of *fuzzy vault* scheme. To clarify the misunderstanding, the Reed Solomon codes and Reed Solomon decoding method used for the *fuzzy vault* scheme is explained here in detail.

4.2.1 Generalized Reed Solomon Codes

A *Generalized Reed-Solomon* (RS) code [8] over the finite field $F = GF(q^2)$, is a linear code C_{RS} over F with a parity check matrix

$$\mathbf{H_{RS}} = \begin{pmatrix} 1 & 1 & \dots & 1 \\ \overline{x_1} & \overline{x_2} & \dots & \overline{x_m} \\ \overline{x_1}^2 & \overline{x_2}^2 & \dots & \overline{x_m}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \overline{x_1}^{m-k-1} & \overline{x_2}^{m-k-1} & \dots & \overline{x_m}^{m-k-1} \end{pmatrix} \begin{pmatrix} v_1 & 0 & \dots & 0 \\ 0 & v_2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & v_m \end{pmatrix},$$

where $\overline{x_1}, \overline{x_2}, \dots, \overline{x_m}$ are distinct nonzero elements in F, and $v_1, v_2, \dots, v_m$ are nonzero elements in F that are not necessarily distinct. The elements $\overline{x_i}$ are called the code locators and v_i are column multipliers. The matrix H_{RS} is called the *canonical* parity check matrix of C_{RS}. A canonical parity check matrix is not unique, even up to scaling of column multipliers, due to the fact that the same RS code can be defined through more than one list of code locators. However, we give a method to calculate a canonical parity check matrix in Section 4.2.2.

The conventional Reed Solomon codes are not suitable for *fuzzy vault* decoding since they require an element $\overline{\alpha}$ where $\overline{\alpha}$ is an element of multiplicative order m in F to create the generator [23]. Note that m (number of minutia points matched) is not a predefined value and may differ for every user which makes conventional RS codes not suitable for *fuzzy vault*.

As explained in the enrollment stage, when we construct the *fuzzy vault* we evaluate the secret polynomial for all n minutiae points, i.e. $\overline{y_i} = p(\overline{x_i})$ for $i = 1, \ldots n$. This can be put into a matrix-vector formulation as follows:

$$\begin{bmatrix} \overline{y_1} & \overline{y_2} & \ldots & \overline{y_n} \end{bmatrix} = \begin{bmatrix} p_0 & p_1 & \ldots & p_{k-1} \end{bmatrix} \mathbf{G}$$

where the matrix $\mathbf{G}$ is given as

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \overline{x_1} & \overline{x_2} & \ldots & \overline{x_n} \\ \overline{x_1}^2 & \overline{x_2}^2 & \ldots & \overline{x_n}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \overline{x_1}^{k-1} & \overline{x_2}^{k-1} & \ldots & \overline{x_n}^{k-1} \end{pmatrix}$$

This is indeed a *shortened* RS encoding with the generator matrix $\mathbf{G}$. It is crucial to notice that the generator matrix changes for each user, which differ from the conventional application of RS encoding method. Since the enrollment stage essentially utilizes the RS encoding, the reconstruction of the secret polynomial in the verification stage can be achieved by employing an *RSDecoder*.

The codeword to decode in the *fuzzy vault* scheme is the evaluation of a polynomial of degree $k - 1$ over a set of m distinct points in field F. The codeword consists of m pairs $(\overline{x_i}, \overline{y_i})$ where $\overline{x_i} \in F$ is the minutiae point that matches either a genuine or chaff point in the *fuzzy vault*. If the codeword satisfies $\overline{y_i} \neq p(\overline{x_i})$ for less than τ of the given values of i, the decoder returns the secret polynomial $P(x)$ and thus the fingerprint will be verified. Otherwise, the decoder returns a false polynomial.

4.2.2 RS Decoding with BM Algorithm

The RS decoding with BM algorithm takes two vectors ($\mathbf{X} = [\overline{x_1}, \overline{x_2}, \ldots, \overline{x_m}]$ and $\mathbf{Y} = [\overline{y_1}, \overline{y_2}, \ldots, \overline{y_m}]$) where $\mathbf{X}$ is the code used to create the parity check matrix and $\mathbf{Y}$ is the codeword with errors. The method has four major steps [8]:

1. Computation of canonical parity-check matrix
2. Computation of syndromes
3. Computation of error locater polynomial and error locations
4. Computation of secret polynomial

Firstly, a canonical parity-check matrix $\mathbf{H}$ of the GRS code is an $(m-k) \times m$ matrix such that $\mathbf{H}\mathbf{G}^T = \mathbf{G}\mathbf{H}^T = \mathbf{0}$. $\mathbf{H}$ can be constructed as follows [8]:

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ \overline{x_1} & \overline{x_2} & \ldots & \overline{x_m} \\ \overline{x_1}^2 & \overline{x_2}^2 & \ldots & \overline{x_m}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \overline{x_1}^{m-k-1} & \overline{x_2}^{m-k-1} & \ldots & \overline{x_m}^{m-k-1} \end{pmatrix} \cdot \mathbf{V}$$

where $\mathbf{V}$ is a diagonal matrix with the vector $[v_1, v_2, \ldots, v_m]$ being on its diagonal and

$$v_j = -\left(\prod_{1 \le l \le m \text{ and } l \ne j} (\overline{x_j} - \overline{x_l}) \right)^{-1} \quad \text{for } 1 \le j \le m$$

As the second step, the syndromes of the vector $\mathbf{Y}$ with respect to $\mathbf{H}$ are computed as follows:

$$\begin{pmatrix} S_0 \\ S_1 \\ \vdots \\ S_{m-k} \end{pmatrix} = HY^T$$

If there are no errors in the vector $\mathbf{Y}$, the syndromes will be all zero vector. In this case the decoding algorithm will jump to step 4 since there are no errors to find. Otherwise, the BM algorithm is used to find the error locater polynomial (ELP) which is defined as $\Lambda(x) = \prod_{j \in J}(1 - \overline{x_j}x)$ where J is the set of error locations. Note that $\Lambda(\overline{x_i}^{-1}) = 0 \iff i \in J$.

The Berlekamp Massey algorithm (BM) given in Algorithm 1 computes ELP given the syndrome polynomial $S(x) = \sum_{i=0}^{m-k} S_i x^i$.

Here, given $S(x)$ and n, the algorithm computes i-recurrences ($\sigma_{-1}(x)$,$\omega_{-1}(x)$) of $S(x)$ iteratively up to n where n-recurrence of $S(x)$ is an ordered pair of polynomials ($\sigma(x)$,$\omega(x)$) such that $\sigma(0) = 1$ and $\sigma(x)S(x) \equiv \omega(x) \pmod{x^n}$.

After the ELP is obtained, the roots of the error locater polynomial can be calculated by substituting the inverse of each element of vector $\mathbf{X}$ for $\Lambda(x)$ and checking for zero. This method works since $\Lambda(\overline{x_i}^{-1}) = 0 \iff i \in J$ (i.e. there is an error in i^{th} location of the codeword). Since ELP shows the locations of all errors, the rest of the data must be correct, namely $\forall i \notin J \;\; \overline{y_i} = p(\overline{x_i})$.

Algorithm 1 Berlekamp Massey Algorithm

```
Require: $S(x)$ syndrome polynomial
Ensure: ELP $\Rightarrow \Lambda(x) = \prod_{j \in J}(1 - x_j x)$
 1: $n \leftarrow$ degree of $S(x) + 1$;
 2: $\sigma_{-1}(x) \leftarrow 0$; $\sigma_0(x) \leftarrow 1$;
 3: $\omega_{-1}(x) \leftarrow -x^{-1}$; $\omega_0(x) \leftarrow 0$;
 4: $\mu \leftarrow -1$; $\delta_{-1} \leftarrow -1$;
 5: for $i$ from 0 to $n-1$ by 1 do
 6:     $\delta_i \leftarrow$ coefficient of $x_i$ in $\sigma_i(x)S(x)$;
 7:     $\sigma_{i+1}(x) \leftarrow \sigma_i(x) - (\delta_i/\delta_\mu) \cdot x^{i-\mu} \cdot \sigma_\mu(x)$;
 8:     $\omega_{i+1}(x) \leftarrow \omega_i(x) - (\delta_i/\delta_\mu) \cdot x^{i-\mu} \cdot \omega_\mu(x)$;
 9:     if $(\delta_i \neq 0)$ AND $(\max(\sigma_i, \omega_i + 1) \leq i)$ then
10:         $\mu \leftarrow i$;
11:     end if
12: end for
13: return $\sigma_n(x)$
```

Finally, the secret polynomial can be reconstructed by using the Lagrange interpolation method with the correct minutiae points if the number of errors does not exceed τ. Otherwise, the function returns a wrong polynomial of degree $k-1$. Again we check if more points that lie on the same polynomial exist. If not, the function is called with fewer number of pairs. This process is repeated a few times with some of the different random pairs being removed from the list. If the algorithm still returns a wrong polynomial as output after these attempts, then the fingerprint is rejected.

5 Computational Complexity of Polynomial Reconstruction

In [12], Clancy et al. argue that using the RS decoder is a better approach than the brute force method if the attacker cannot eliminate some of the chaff points from the verification list. But the authors do not provide a comparison between the two approaches. In this thesis we try to clarify as to which method is optimal depending on the parameters of m and k where m is the number of matched points and $k-1$ is the degree of the secret polynomial.

For comparing the two approaches, we calculate the number of operations in the secret polynomial reconstruction phase for both methods. For the sake of simplicity, we ignore addition and assignment operations and only count multiplication and inverse operations in F_{q^2} since the latter two operations dominate the computation.

5.1 Complexity of the RS Decoder

The Reed Solomon decoder has four steps as explained in section 4.2 and the complexity of each step and the total complexity is given in Table 1. We assume that Step 3 always returns an error locater polynomial; i.e. the measured fingerprint always leads to matchings to chaff points.

From the perspective of complexity comparison, the main difference between the brute force and the RS decoding approaches is that RS decoder can distinguish a genuine fingerprint in only one trial if the number of incorrect matchings is less than the error correcting capability of the RS code τ. On

Table 1: Operational Complexity of RS Decoding Method.

Step	Multiplication	Inv.
1. Constructing H(total)	$3m^2 - 2mk$	m
2. Syndrome Computation	$m(m-k)$	-
3. Finding Error Locations	$m(k^2/3+2)$	-
4. Polynomial Construction	k^2	-
Total	$4m^2 + m(k^2/3)$ $-m(3k+2)+k^2$	m

the other hand, the brute force approach may have to perform excessively many trials to complete the verification process.

5.2 Complexity of the Brute Force Method

Complexity of the brute force method is given in Table 2. Selecting k random points out of m matched points (i.e. Step 1 in the table) involves a randomized algorithm, whose complexity we estimate as equivalent to k multiplication operations. The variable l in the last row of Table 2 stands for the number of trials needed on average, which naturally increases with the error in the query fingerprint. Without knowing the number of trials l in the brute force method it is not easy to compare two methods. Comparison is only possible with experiments on real and synthetic data, which we achieve in Section 9. However, it is important to note that one round of brute force is faster than Reed Solomon method.

Table 2: Operational Complexity of Brute Force Method

Phase	Multiplication	Inv.
1. Choosing k random points	k	-
2. Polynomial Construction	k^2	-
3. Verification of the result	$5m$	-
Total	$l(k^2 + 5m + k)$	-

6 Attacks on Fuzzy Vault

Although Juels and Sudan [3] proved that the *fuzzy vault* scheme satisfies some security properties, it is still vulnerable to some attacks. The attacks on the *fuzzy vault* scheme, mostly assume the interception of a vault from a database. There are several attacks targeted on the *fuzzy vault* scheme such as the brute force attack and the correlation attack which can be applied in reasonable amount of time. Therefore, the *fuzzy vault* scheme is insecure without additional security measures.

6.1 Location Based Attack

The vault involves the location of all the points, either chaff or genuine. Therefore, creation of random chaff points is crucial since they should be uniformly distributed in the minutiae space so that an attacker, having access to the vault, should not be able to distinguish between genuine minutiae points and random chaff points [24]. In the original scheme proposed by [3], the chaff points were created with the condition that every point in the vault should be at least t Euclidean distance apart to supply a uniform distribution. However, in this method the *fuzzy vault* could leak some information about the location of genuine points. We have no control over the locations of genuine points, as some of them might be very close to each other as exemplified in Figure 7 where chaff points are represented as circles and genuine points as circles with crosses.

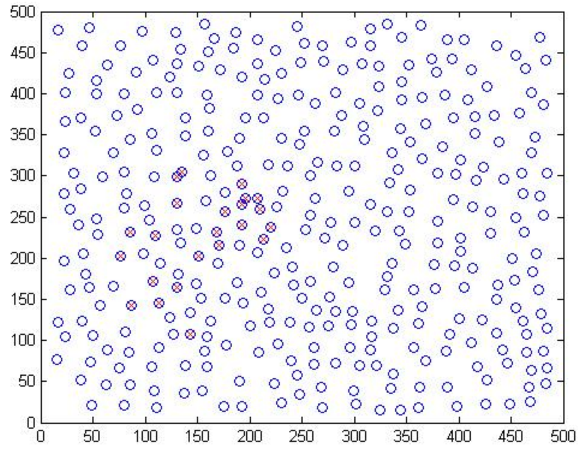


Figure 7: Original Fuzzy Vault where genuine points are marked

If an attacker intercepts a vault, he can locate some of the genuine points correctly by checking the distances between the points; i.e. if the distance between two points is closer than the threshold t, then these points are genuine. Although this attack may not be sufficient to find the secret polynomial since the number of identified genuine points will probably less then k, it will highly reduce the complexity of the brute force attack that is explained in Section 6.2.

6.2 Brute Force Attack

If an attacker intercepts a vault, but has no other information about the locations of the genuine points, the best method to recover the secret polynomial is brute force trial [12][7]. Mihailescu provides a strong brute force attack in [7], which finds the secret polynomial in less than $8(Ck)(C/n)^k$ operations where C is the number of points in the vault, n is the number of genuine points in the vault and the degree of the secret polynomial is $k-1$. The idea of the attack relies on the established fact [3] that when there are more than D vault points on a polynomial of degree $k-1$ for a fixed threshold $D \in (k-1, n)$, this polynomial is the secret polynomial with a very high probability. In this attack, the intruder chooses random k points from the vault, where $k-1$ is the degree of the secret polynomial and finds the polynomial that these k points lie on. Later he tests how many other vault points lie on that polynomial, if it is greater or equal to D, he claims this is the secret polynomial and all the points that lie on that polynomial are genuine minutia points of the enrolled fingerprint. Otherwise, the attacker choose another random k points and repeats the operation until he finds a polynomial that has more than D points that lie on it.

6.3 Correlation Attack

Scheirer et al. [25] suggested another kind of attack called attack via record multiplicity. This kind of attack assumes that the attacker intercepts at least two *fuzzy vaults* that belongs to the same user. Note that these vaults may be created by different secret polynomials and different chaff points but the genuine points should highly overlap since they are the minutia points of the same fingerprint. Scheirer claimed that correlating these two vaults may reveal the biometric data (i.e. minutia points). Later, Kholmatov et al. [6] showed that by using this property of the *fuzzy vault* scheme, it is possible to link an unknown vault to another vault that is constructed by the same biometric in a reasonable amount of time with high probability. Kholmatov et al. calculated that given two matching vaults, the secret polynomial can be revealed 59% of the time.

Due to this reason, some additional security measures are necessary for a secure *fuzzy vault* scheme. Recently Nandakumar et al. [26] implemented the *fuzzy vault* scheme by combining it with passwords. This scheme successfully overcomes the vulnerability of the scheme against correlation attack without increasing the false reject rates. However, this system is a two factor authentication scheme where the user has to provide both the password and the biometric during authentication and that maybe inconvenient in certain applications.

In this thesis we propose methods that improves the security of the scheme against all three attacks, namely location based attack, brute force attack and correlation attack. We changed the threshold settings as explained in Section 7.1 for securing the scheme against location based attack. A novel chaff point placement method in Section 7.2 is proposed as a remedy for brute force attack. For a remedy against correlation attack, we keep distorted versions of the biometric that preserves the invariants of the biometric image. The details of this method is explained in Section 8.2.

7 New Enrollment Stage

In this section, we explain two proposed modifications to the enrollment stage in order to strengthen the *fuzzy vault* against possible attacks.

7.1 Distribution of Chaff Points

As previously explained in Section 6.1, some of the genuine points can be identified by just examining the locations of points relative to their neighboring points. As a remedy, we generate the chaff points with the condition that every chaff point in the vault should be at least at Euclidean distance t from a genuine point and should be at least at Euclidean distance t' from any other chaff point. Note that $t' < t$ since having chaff points far from the genuine points is necessary and have a positive effect on false reject rate (FRR) as demonstrated in Section 9. Smaller threshold t', on the other hand, for inter-chaff point distance is necessary to imitate the distribution of genuine points where close genuine points occasionally occur in the vault. While t value depends on the fingerprint image size and the total number of points in the vault, t' should be chosen depending on the distribution of genuine points. For fingerprint image of 500×500 pixels, Figure 8 shows the *fuzzy vault* constructed with the new chaff point placement strategy, where the threshold values t and t' are chosen as 18 and 8, respectively. As seen in Figure 8, the distribution of chaff points in the *fuzzy vault* closely resembles the distribution of genuine points, hence an attacker cannot easily distinguish the genuine points from the chaff points.

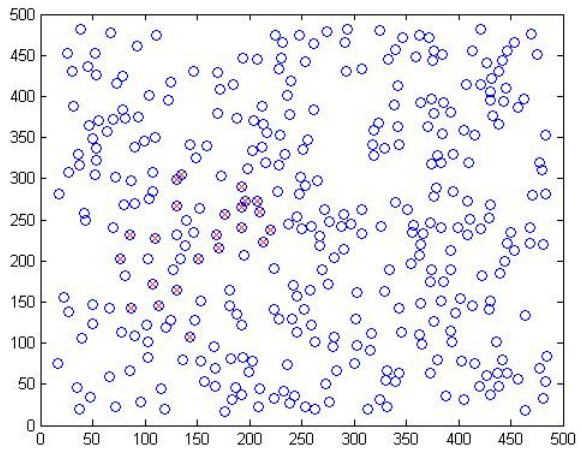


Figure 8: Fuzzy vault with new scheme where genuine points are marked

7.2 A Novel Method for Chaff Point Placement

As explained in Section 6.2, Mihailescu proved that in less than $8Ck(C/n)^k$ operations[2], the intruder can recover the secret polynomial [7]. Note that there are C points in the vault where n of them are genuine and the secret polynomial has degree $k-1$.

Our proposed method to improve the security involves the idea that, by choosing the chaff points at random, but in a more clever way, we can embed some other (randomly chosen) polynomials of degree $k-1$ other than the secret polynomial in the vault. If we guarantee that the number of chaff points that lie on these (fake) polynomials, is around n — the same number of genuine points on the secret polynomial on average — the attacker cannot distinguish the secret polynomial from the fake ones. Otherwise the attacker

[2]Operation means atomic arithmetic operations such as additions and multiplications.

who succeeds to construct a polynomial can discard it if there are fewer points. One way of choosing chaff points is described in Algorithm 2.

Algorithm 2 Algorithm for chaff point selection

```
1: Place the genuine points in the vault
2: Keep the unassigned chaff points in a pool
3: Keep a list of fake polynomials which is initially empty
4: repeat
5:    Pick a random number $r$ close to $n$
6:    For the first fake polynomial, take random $k-1$ points from the vault
      and take one random point from the pool. For others take random $k$
      points from the vault.
7:    Find the $(k-1)^{st}$ degree polynomial that passes through the selected
      points. Add the polynomial to the list if it is not already in it.
8:    Check the vault if there are any other points that lie on the polynomial.
      Decrement $r$ by the number of points on this polynomial.
9:    Pick $r$ points from the pool (or the remaining points if their number
      is less than $r$) and place them on the fake polynomial and place the
      resulting values in the fuzzy vault.
10: until the pool is empty
```

With the proposed chaff placement method, we allow each polynomial to intersect with other polynomials in at least k vault points which increases the maximum number of polynomials we can embed into the vault. Note that no two polynomials can intersect with each other in more than $k-1$ points. Since each of the embedded polynomial is of the same degree and has similar number of vault points that lies on it, they are equally likely to be the secret polynomial. Therefore by increasing the number of embedded fake polynomials, we reduce the probability that the attacker successfully guess the secret polynomial. As a result of our experiments in our setting described above, we are able to hide around 30 fake polynomials in the vault. Therefore, this method decreases the probability of finding the secret polynomial using Mihailescu's attack from 100% to approximately 3.3% after the brute force attack is applied. Due to the fact that most of the identification applications only allow a limited number of trials, the proposed method enhances the security considerably. Moreover, the method does not affect the false accept or false reject rates since the matching algorithm considers only the x coordinates of the points and this method changes only the y coordinates.

7.3 Security against Brute Force Attacks

As explained in Section 6.2, there is an efficient brute force attack that can find the secret polynomial and the biometric information in less than $8(Ck)(C/n)^k$ operations where C is the number of points in the vault, n is the number of genuine points in the vault and the degree of the secret polynomial is $k-1$.

In our tests the parameter n is on average 35 and k is constant 10. For $C = 300$, which gives a better FRR, breaking the system requires $8 \times 300 \times 10 \times (300/35)^{10} \approx 2^{46}$ operations. For $C = 350$, which gives a worse FRR, the system provides a better security; breaking the system requires this time $8 \times 350 \times 10 \times (350/35)^{10} \approx 2^{48}$ operations.

Without the use of the proposed method in Section 7.2, the secret polynomial is found with probability 1 after this attack. However, our proposed method decrease the probability to approximately 0.03 since the polynomial found as a result of brute force attack, is not guaranteed to be the secret polynomial.

8 Preventing Correlation Attacks

As explained in Section 6.3, the *fuzzy vault* system is vulnerable against the correlation attack and some additional security measures are necessary for a secure *fuzzy vault* system. We propose a special hash function for keeping the hash values of the minutia points and perform the matching in hash space. In this section, we first define the requirements a hash function should satisfy to be used in a secure *fuzzy vault* scheme and propose a hash function that overcomes the vulnerability against correlation attack.

8.1 Requirements of the Hash Function

Due to the correlation attack against *fuzzy vault*, we should randomize the minutia points. One well-known method for randomization is encryption but this requires the safeguarding of the private keys which is another problem. Use of hash values of the minutia points, instead of the minutia points themselves is an efficient method for randomization and this method does not require safeguarding of keys since everything is public. In this section, we define the key properties that a hash function should satisfy to be used in a secure *fuzzy vault* implementation. We define a family of 3D-hash-functions as a set $H = \{h_i : Z_{q^2} \rightarrow Z_{p^3}\}_i$ where $p < q$. We represent the hash of a point with $h(x, y)$ and $h_V(x, y)$ represents the hash of all the minutia points together with the chaff points (i.e. the vault). We use Chebyshev distance [27] (also called infinity norm distance) to measure the distance between two points in a vault. Here, the distance between two vectors is the greatest of their differences along x and y coordinates. The Chebyshev distance between two vectors or points v and w, with standard coordinates v_i and w_i, respectively, is:

$$||v - w||_\infty = \max(|v_i - w_i|).$$

Any hash function that is used for biometrics should satisfy the following properties:

1. Verifying a legitimate user should be possible (Robustness)
2. It should be secure against correlation attack (Non Linkability)
3. It should not be possible to learn the original biometric from the hashed version of it (Non-Invertability)

The formal definitions for these properties are explained below.

Definition 1. *Robustness*

Given two very similar fingerprint images (i.e. one is the noisy version of the other one), for any hash function $h \in H$ the possibility that their hash values are also very similar should be large. The formal definition is as follows:

$$||(x, y) - (x', y')||_\infty < \delta \Rightarrow P_{h \in_R H}[||h(x, y) - h(x', y')||_\infty < \epsilon] > 1 - \sigma \quad (1)$$

where $\sigma = p(\epsilon)$ for some polynomial p.

Given a vault, V, and a set of vaults, S, where exactly one vault $V' \in S$, is created from the same fingerprint as V, with a different hash function from the set H, the probability of matching V' to V in polynomial time should be at most $1/|S| + \epsilon$, for some security parameter ϵ. This requirement is formalized in the following definition.

Definition 2. *Non-Linkability*

Let d be the average distance between a point and its nearest neighbor in the vaults (under the assumption that each vault has the same number of points, we can further assume that the average distance is the same for all vaults). Here we consider the worst case and assumed that there is no noise between the two enrolled fingerprints. We can give the formula of the definition as follows:

$$P_{x,y \in Z_q, h, h' \in H}[||h(x,y) - h'(x,y)||_\infty \leq d/2] < \epsilon \tag{2}$$

Intuitively the definition says that, given a minutia point hashed by different functions, the possibility to correlate the two points is very low.

Definition 3. *Non-Invertability*

Given a hash value $v = h(x,y)$ there is no unique point in the pre-images of v.

$$P_{h \in_R H}[x' = x, y' = y | h(x,y) = h(x',y')] < \phi \text{ for some } \phi < 1/2 \tag{3}$$

8.2 Our Hash Function

Let $R_{\alpha,\beta,\gamma}$ be a 3×3 rotation matrix which rotate real-valued vectors by angles of α, β, γ, around x, y and z axis respectively which is calculated as:

$$\mathbf{R}_{\alpha,\beta,\gamma} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \text{-}\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ \text{-}\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} \cos\gamma & \text{-}\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and M^i represents the i^{th} row of some matrix M. For integers q and p_i, where $q > p_i$ for $i \in \{1, 2, 3\}$ and p_i are prime numbers, we define a family of hash functions $H = \{h_{\alpha,\beta,\gamma} : Z_{q^2} \rightarrow Z_{p_i{}^3}\}_{\alpha,\beta,\gamma}$ as:

$$[h_{\alpha,\beta,\gamma}(x, y)]^i = round(R^i_{\alpha,\beta,\gamma}(x, y, 1)^T) \bmod p_i, \text{ for } i = 1, 2, 3 \qquad (4)$$

where *round* function maps a real number to the closest integer and n is the number of minutia points. In this work we choose the primes between $q/2$ and $q/5$, which achieves low FRR values while satisfying the necessary security. According to the prime number theorem [28], there are approximately $x/log(x)$ primes not exceeding x. This gives us around $(\frac{q/2}{log(q/2)}) - (\frac{q/5}{log(q/5)})$ possible candidates for p_i.

The block diagram of the proposed *fuzzy vault* enrolling and verification scheme is shown in Figure 9 and Figure 10, respectively.

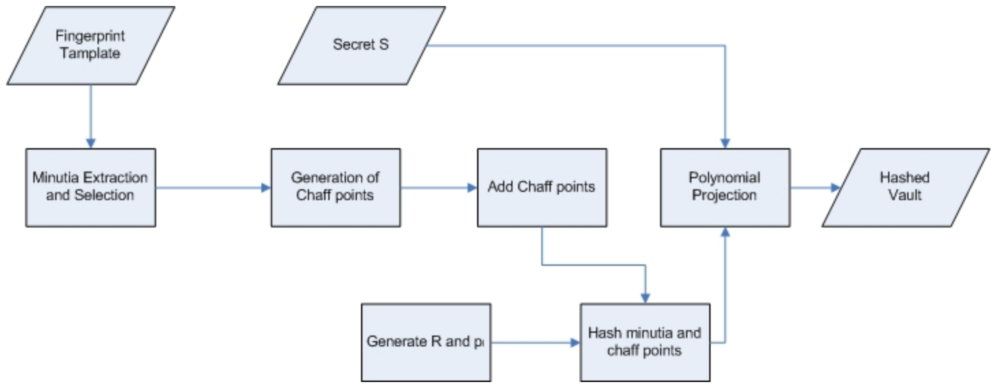


Figure 9: Proposed Vault Enrolling

Note that the proposed verification is different from the original scheme only in the alignment phase, where the proposed alignment phase first hashes

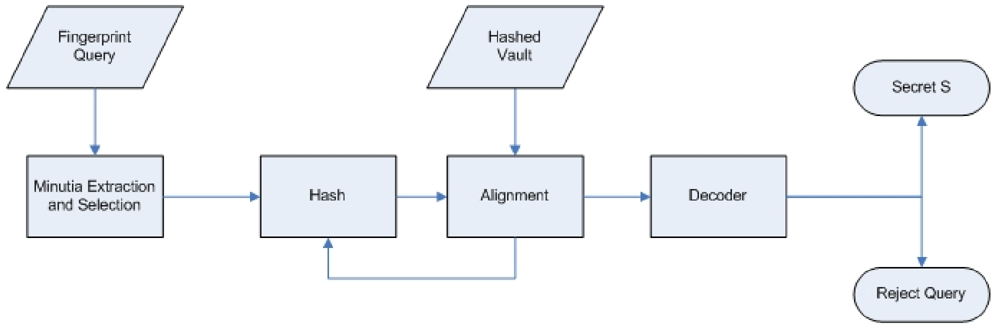


Figure 10: Proposed Vault Verification

the query points and then aligns them with the vault. The details of the alignment phase are given in Algorithm 3.

Algorithm 3 Algorithm of the alignment phase

```
1: for all rotations and translations do
2:     Apply a rotation and translation to the query minutia points
3:     Hash them with the public hash function
4:     Compare the distance with Hashed Vault points
5:     if #matched points is larger than maximum matched then
6:         Update maximum matched and keep this set of points
7:     end if
8: end for
```

8.3 Analysis of The Hash Function

A hash function should satisfy the three properties given in Section 8.1. We claim that the hash function we proposed in Section 8.2 satisfies these properties.

Lemma 1. (*Robustness)* Given two points where the Chebyshev distance between them is less than δ, our proposed hash function is robust according to Definition 1.

Proof. Let the hashing matrix be:

$$\mathbf{R}_{\alpha,\beta,\gamma} = \begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{pmatrix}, \tag{5}$$

where all elements of R are real numbers between [-1,1].

Assume (x, y) and (x', y') are two points where the Chebyshev distance between them is less than δ. This implies that $|x - x'| < \delta$ and $|y - y'| < \delta$. From this we can trivially derive the following inequalities:

$|x - x'| < \delta \Rightarrow -\delta < a_i|x - x'| < \delta$ since $-1 < a_i < 1$

$|y - y'| < \delta \Rightarrow -\delta < b_i|y - y'| < \delta$ since $-1 < b_i < 1$

Recall that $||h(x, y) - h(x', y')||_\infty$ =max($|(a_ix + b_iy + c_i) \bmod p_i - (a_ix' + b_iy' + c_i) \bmod p_i|$) for $i \in \{1, 2, 3\}$

Let σ be:

$\lfloor \frac{q}{p_i} \rfloor 2\delta/q < \frac{2\delta}{p_i}$ (i.e. σ is an approximation to the probability $(a_ix + b_iy + c_i) \geq kp_i$ and $(a_ix' + b_iy' + c_i) < kp_i$ or vice versa for some $k \leq \lfloor \frac{q}{p_i} \rfloor$.) Note that since we assume x and y are uniformly distributed and p_i is a prime number, $(a_ix + b_iy + c_i) \bmod p_i$ should also be uniformly distributed. There are $\lfloor \frac{q}{p_i} \rfloor$ points r where $r = kp_i$ for some positive integer $k \leq \lfloor \frac{q}{p_i} \rfloor$ and given the value $(a_ix + b_iy + c_i)$, the distance between this value and the value $(a_ix' + b_iy' + c_i)$ can at most be 2δ.

With probability $1-\sigma$:

$$\begin{aligned}||h(x,y)-h(x',y')||_\infty &= \max(|(a_i|x-x'|+b_i|y-y'|)| \bmod p_i) \\ &< a_i\delta + b_i\delta < 2\delta \text{ for } i \in \{1,2,3\}\end{aligned}$$

Given that, $||(x,y)-(x',y')||_\infty < \delta$, $P_{h\in_R H}[||h(x,y)-h(x',y')||_\infty < \epsilon] > 1-\sigma$ for some $\epsilon < 2\delta$ and $\sigma < \frac{2\delta}{p_i}$ □

Lemma 2. (*Non-Linkability*) Given a minutia point, the two hashed vault points created by our proposed hash function, that uses the same minutia point but different α, β, γ values, are not linkable according to Definition 2 given above.

Proof. Let d be given, and let

$$\mathbf{R}_{\alpha,\beta,\gamma} = \begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{pmatrix} \text{ and } \mathbf{R}'_{\alpha',\beta',\gamma'} = \begin{pmatrix} a'_1 & b'_1 & c'_1 \\ a'_2 & b'_2 & c'_2 \\ a'_3 & b'_3 & c'_3 \end{pmatrix}$$

be random variables describing the choice of hashing matrices for h and h' respectively.

We let $\mathcal{C}$ be the event $\|h(x,y)-h'(x,y)\|_\infty \leq d/2$, and let $\mathcal{D}_\tau$ be the event that the maximum of the angle differences: $|\alpha-\alpha'|, |\beta-\beta'|$, or $|\gamma-\gamma'|$ is larger than τ or $p_i \neq p'_i$ and also let $\overline{\mathcal{D}_\tau}$ be the converse event of $\mathcal{D}_\tau$.

$$\begin{aligned}P[\mathcal{C}] &= P[\mathcal{C}|\mathcal{D}_\tau]P[\mathcal{D}_\tau] + P[\mathcal{C}|\overline{\mathcal{D}_\tau}]P[\overline{\mathcal{D}_\tau}] \\ &\leq P[\mathcal{C}|\mathcal{D}_\tau] + P[\overline{\mathcal{D}_\tau}] \qquad (6)\end{aligned}$$

If we fix $R_{\alpha,\beta,\gamma}$, there are $(360/2\tau)^3 = (180/\tau)^3$ possible rotation matrices, $R'_{\alpha',\beta',\gamma'}$ and $([(\frac{q/2}{log(q/2)}) - (\frac{q/5}{log(q/5)})])^3$ p_i triplets which fall in the event $\overline{\mathcal{D}_\tau}$. So the probability $P[\overline{\mathcal{D}_\tau}] = ((\frac{180}{\tau}) \times [(\frac{q/2}{log(q/2)}) - (\frac{q/5}{log(q/5)})])^{-3}$.

$P[\mathcal{C}|\mathcal{D}_\tau]$ is equal to the probability that a randomly chosen point in $Z_{p_i{}^3}$ is in the δ range (i.e. matched) to a fixed point since the hashed minutia points are uniformly distributed over $Z_{p_i{}^3}$.

$P[\mathcal{C}|\mathcal{D}_\tau] = (\frac{2\delta}{p_i})^3$.

Putting it all together, we can write (6) as

$$\begin{aligned} P[\mathcal{C}] &< P[\mathcal{C}|\mathcal{D}_\tau] + P[\overline{\mathcal{D}_\tau}] \\ &< \left(\frac{2\delta}{p_i}\right)^3 + \left(\left(\frac{180}{\tau}\right)\left[\left(\frac{q/2}{log(\frac{q}{2})}\right) - \left(\frac{q/5}{log(\frac{q}{5})}\right)\right]\right)^{-3} \end{aligned} \tag{7}$$

Setting ϵ equal to the upper bound of $P[\mathcal{C}]$ shown in (7) completes the proof. $\square$

Lemma 3. *(Non-Invertability)* Given a point (x, y),there is no unique point in the pre-images of the hash value $v = h(x, y)$ if $h \in H$, therefore our hash function is not invertible according to Definition 3 above.

Proof. While hashing the vault, we are multiplying the vector $(x, y, 1)^T$ with a 3×3 rotation matrix in modulo p_i. The adversary can apply the following attack to guess the x and y values where x and y are coordinates of one point (i.e. either chaff or genuine) in the vault.

Assume:

$round(R_{\alpha,\beta,\gamma}(x, y, 1)^T) = (x', y', z')$ and

$x' = a \bmod p_1, y' = b \bmod p_2, z' = c \bmod p_3$ (i.e. $h_{\alpha,\beta,\gamma}(x, y) = (a, b, c)$)

The adversary can try all possible values of x', y' and z' by adding or subtracting p_i values and find which of these triplets satisfy the following condition:

$R^{-1}(x', y', z') = (x, y, z)$ where $0 < x, y < q$ and $z = 1$

However, due to the rounding operation, satisfying $z = 1$ exactly is not possible so the attacker must consider the x, y values that satisfies $1-E < z < 1+E$ where E is an error threshold which do not give a unique solution. $\square$

Our empirical results show that a reasonable value for E is 2, and we tested that the number of x, y pairs that satisfies this condition is on the average 6 and never less than 4. Therefore, we expect that a hash value of each point has at least $f = 4$ pre-images. In the worst case the adversary has to perform f^C matching to verify that two *fuzzy vaults* match (where C is the number of points in the vault). This is clearly infeasible, since the complexity grows exponentially in the size of the vault.

8.4 Security Analysis against Correlation Attack

The correlation attack that is suggested by Scheirer et al. [25] depends on identifying the genuine points that are common in both vaults by correlating the x values in two vaults. The correlation of x values in two vaults is done by exhaustive matching [6] where the matching algorithm tries many possible rotations and translations and chooses the one that maximizes the number of matching points in two vaults. This algorithm can only be applied if the vaults can be rotated and translated without loosing precision, for performing exhaustive matching algorithm. However, this is not possible in our proposed *fuzzy vault* scheme with hashing.

Assume that an attacker captures two *fuzzy vaults* (v_1, v_2) that are created by using the same biometric data but with different hash functions (R_1, R_2). In the worst case there will be no noise (i.e. rotation and translation) between two impressions of the same fingerprint and the randomly chosen modulo values will be equal.

$v_1 = round(R_1(x, y, 1)^T_{n\times 3}) \bmod p$

$v_2 = round(R_2(x, y, 1)^T_{n\times 3}) \bmod p$

Since rotation matrices are invertible there exist some matrix R_3 such that $R_1 = R_3 * R_2$. If *fuzzy vaults* are rotatable without loosing precision then the equation $v_1 \approx round(R_3 * v_2) \bmod p$ should hold with some small error due to rounding. Although modulo is a congruence relation (respecting addition, subtraction, and multiplication) on the integers, this relation is not valid in real numbers.

Note that if $a \equiv a_2 \bmod p$ and $b \equiv b_2 \bmod p$, where $a, b \in Z$ then:

$(ab) \bmod p \equiv (a \bmod p \times b \bmod p) \bmod p \equiv (a_2 b_2) \bmod p$

However, this equivalence relation does not hold where $a, b \in R$. Assume that $a = pa_1 + a_2 + a_3$ and $b = pb_1 + b_2 + b_3$, where $a_1, a_2, b_1, b_2 \in Z$ and $0 < a_3, b_3 < 1$

$$\begin{aligned}(ab) \bmod p &= (a_1 b_1 p^2 + a_1 b_2 p + a_2 b_1 p + a_1 b_3 p + b_1 a_3 p + a_2 b_2 + a_2 b_3 + b_2 a_3 + a_3 b_3) \bmod p \\ &= (a_1 b_3 p + b_1 a_3 p) \bmod p + (a_2 b_2 + a_2 b_3 + b_2 a_3 + a_3 b_3) \bmod p \\ &= (a_1 b_3 p + b_1 a_3 p) \bmod p + (a \bmod p)(b \bmod p) \bmod p\end{aligned}$$

Since a_3 and b_3 are not integer, the term $a_1 b_3 p + b_1 a_3 p$ may not be an integer multiple of p, therefore it does not equal to 0 when taking modulo p. The error term Δ where $\Delta = [(ab) \bmod p - (a \bmod p)(b \bmod p) \bmod p]$ can be calculated as:

$\Delta = (a_1 b_3 p + b_1 a_3 p) \bmod p$

Due to this error term Δ our proposed scheme does not satisfy $v_1 \approx round(R_3 * v_2) \bmod p$.

Another method to correlate the two vaults is to try to match them without applying any rotation. As proven in Lemma 2, the probability to link the two hashed vault points created by our proposed hash function, that uses the same minutia point but different α, β, γ values, is lower than a security parameter ϵ. This lemma can be generalized for the probability to link the two hashed vaults created by our proposed hash function, that uses the same fingerprint but different α, β, γ values. Recall that for linking two vaults, at least k of the genuine points should match.

Our empirical results show that δ is a 2 bit number and p_i is a 7 bit number. Therefore, $P[\mathcal{C}|\mathcal{D}_\tau] = (\frac{2\delta}{p_i})^3 = (2^3/2^7)^3 = 2^{-12}$. Also the empirical value τ is 3 degrees, and the number of modulus triplets we can use is around 30^3, where $q = 500$. So the probability $P[\overline{\mathcal{D}_\tau}] = ((\frac{180}{3}) \times 30)^{-3} \approx 2^{-32}$.

Note that $P[\mathcal{C}|\mathcal{D}_\tau]$ is the probability of one random point matched to a fixed point. Let $\mathcal{L}$ be the event of linking two vaults, $P[\mathcal{L}|\mathcal{D}_\tau]$ is the probability that at least k of the n genuine points are randomly matched. This probability is less than two times of the probability of exactly k of the n genuine points matched since the probability exponentially decreases when the number of points matched increase.

$$P[\mathcal{L}|\mathcal{D}_\tau] \leq 2 \times \left(\begin{array}{c} n \\ k+1 \end{array} \right) (2^{-12})^k \tag{8}$$

Note that $P[\mathcal{L}|\mathcal{D}_\tau]$ is a lot smaller than $P[\overline{\mathcal{D}_\tau}]$ where $k = 10$ is a constant.

$$\begin{aligned} P[\mathcal{L}] &= P[\mathcal{L}|\mathcal{D}_\tau]P[\mathcal{D}_\tau] + P[\mathcal{L}|\overline{\mathcal{D}_\tau}]P[\overline{\mathcal{D}_\tau}] \\ &\leq P[\mathcal{L}|\mathcal{D}_\tau] + P[\overline{\mathcal{D}_\tau}] \\ &< 2 \times P[\overline{\mathcal{D}_\tau}] < 2^{-31} \end{aligned} \tag{9}$$

The result above shows that if a person affiliates to two different databases with the same finger, the probability that these two enrollments can be correlated is less than ϵ where $\epsilon = 2^{-31}$. This result can be used to answer the question: "What is the minimum number of databases one should affiliate such that the expected probability for finding one collusion is at least 50%?". This problem can be adapted to the birthday paradox [28], where the number of databases is the number of people and $1/\epsilon$ is the number of possible birthdays. According to the birthday paradox, if one person affiliates to $\sqrt{1/\epsilon} = 2^{15,5}$ different databases with the same finger, at least one pair of templates can be corrolated with probability 1/2.

Now let us assume that there are only two databases and there are N people that are affiliated to both databases with the same finger. The probability that at least one person's two enrollments can be correlated is $1 - (1 - \epsilon)^N$. Note that $1 - \epsilon$ is the probability of two templates of one person is not correlated and $(1 - \epsilon)^N$ is the probability of none of the two templates of N people is not correlated.

The birthday attack can also be adapted to the case with N people, again assigning a 0.5 probability of random collusion. The minimum number of databases one should affiliate such that the expected probability for finding one collusion for at least one pair of templates of one person is at least 50% will be $\sqrt{\frac{1}{1-(1-\epsilon)^N}}$

Moreover, suppose a smart attacker found a way to rotate the vault without loosing too much precision. The correlation can only be possible if both vaults are in the same space. Two vaults are in the same space if they are created by using the same modulo (i.e. p_1, p_2, p_3 are equal to p_1', p_2', p_3' respectively). The *fuzzy vault* scheme is still secure against that attack since the probability that the two matching vaults are hashed with the same modulus is $1/30^3$ for our empirical values. Moreover Kholmatov et al. [6] calculated that, for two dimensional classical vaults, linking an unknown vault to a set with 400 vaults is possible for only 40% of all the cases. Since in this new method we are loosing some precision, this probability will be lower. Therefore, this attack can successfully link two matching vaults with some probability less than 0.000037% if some smart rotation method that we do not know, is found.

9 Test Results

We implement polynomial reconstruction phase with both of two previously discussed approaches: 1) brute force method and 2) RS decoding.

For the tests we use a database of 180 people where there are two fingerprint images for each finger, for a total of 360 fingerprints. The first 180 fingerprint images are used for enrollment and the second 180 images are used for verification of the corresponding fingerprints. Later, all fingerprints are cross-tested for false accept rates. In the experimental setting bitmap images of 500×500 pixels are created for each fingerprint.

All computations and tests are performed on a computer with 1.7 GHz Intel Celeron M processor and 448MB of RAM. The codes are developed in either MATLAB or C++ (Microsoft Visual Studio) depending on the nature of the problem.

We investigate two issues; firstly, the effects of the vault and threshold sizes on the performance and security of the *fuzzy vault*, and secondly, time efficiencies of the two methods used in the polynomial reconstruction phase of the verification stage.

9.1 Effects of Vault and Threshold Sizes and Usage of Hash Function

The false reject rates (FRR) and false accept rates (FAR) are calculated in six settings where different values for vault size and minimum distance threshold are used for our database of fingerprints. We use vault sizes of 200, 250 and 300 points and minimum distance threshold between a genuine and a chaff point is tested for ($t = 15$) and ($t = 18$). The minimum distance between any two chaff points is taken as 8. We calculate the FAR and FRR results for both the brute force and the RS decoding methods.

The FAR rates is 0% in all settings after cross testing all fingerprint images with different fingers in the six settings.

Table 3: FRR without hashing in four different settings

	Vault Size		
Threshold (t)	200	250	300
15	2.78%	4.45%	6.65%
18	1.12%	1.67%	2.78%

Table 3 shows the FRRs for different vault sizes and threshold values. The results clearly demonstrate that as the minimum distance between two points increases, the possibility of a genuine point matching to a chaff point decreases resulting in lower FRRs. Although the two values of threshold used in the experiments have the same security level, increasing it further may become impossible after a certain point since we cannot place as many points as needed. Similarly, when more chaff points are added to the vault, the possibility of a genuine point matching to a chaff point increases. Larger vault size results in higher security. The security impact of vault size was analyzed in Section 7.3.

We also calculated the false reject rates (FRR) and false accept rates (FAR) in six settings when the proposed hash function is used, where different values for vault size (C) and modulus (p) is used in the same database of fingerprints. We use vault sizes of 200, 250 and 300 points, random modulus triplet of ($p = (p_1, p_2, p_3)$) where p_i is a random prime between 100 and 250 and we fix the minimum distance threshold to $t = 18$.

The FAR rates are still 0% in all settings after cross testing all fingerprint images with different fingers in the six settings. The FRRs for the different vault sizes and modulus values when the hash function is used, are presented in Table 4.

	Vault Size(C)		
Modulus (p)	200	250	300
$(197, 163, 181)$	2.78%	4.45%	6.1%
$(191, 157, 173)$	3.33%	4.45%	6.65%

Table 4: FRR with hashing in six different settings

Table 4 shows the FRRs for different vault sizes and modulus values. The results clearly demonstrates that as larger modulus values are used, the precision loss from the original data decreases resulting in lower FRR. Similarly, when more chaff points are added to the vault, the possibility of a genuine point matching to a chaff point increases resulting in higher FRR. Although larger vault size results in higher security, due to higher FRR, increasing the vault size further is not feasible.

9.2 Timing Results for Polynomial Reconstruction Phase

As explained before, the verification phase is the main part where two approaches are compared in terms of timing and success performance. The "Number Theory Library" (NTL) [29] is used for all the operations in the polynomial reconstruction. For both approaches, the polynomial reconstruction algorithms take two vectors $\mathbf{x} = \{x_1, x_2, \ldots, x_m\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_m\}$ where $y_i = P(x_i)$ for the points matched to a genuine point and $y_i \neq P(x_i)$ for the points matched to a chaff point.

We first compare the timing results of the two approaches using our database of real fingerprints. In case of true fingerprints we find that for 15% of the data, the RS decoding approach is faster. For non-matching fingerprints, the RS decoding method is naturally always faster since concluding that a fingerprint is non-matching takes 1000 trials which is a fixed value in our experiments.

Considering that the database used in the experiments may not fully represent all cases, especially the ones with high levels of noise in the measurement process, we use synthetic data to control the number of matched points in the alignment process. We generate synthetic fingerprints and corresponding *fuzzy vaults* of 350 total points, where the number of matched points are in the range of $[21, 35]$. Timing results of the experiment for changing number of genuine matched points where the total number of matched points is fixed are shown in Figure 11. As clearly observed in the figure, the brute force method takes much longer when the number of genuine matching points are low due to excessive number of trials to find the correct matching points. As the number of matching points increases, the brute force becomes faster. We also provide the same graphic for number of operations in Figure 12, that we obtain in our theoretical analysis. The similarity of the two figures show that the experimental results are in line with the theoretical analysis. It is important to note that the reason for timing results of RS decoding is worse than the operational complexity of RS decoding is that we did not consider the cost of function calls while calculating the operational complexites of the method.

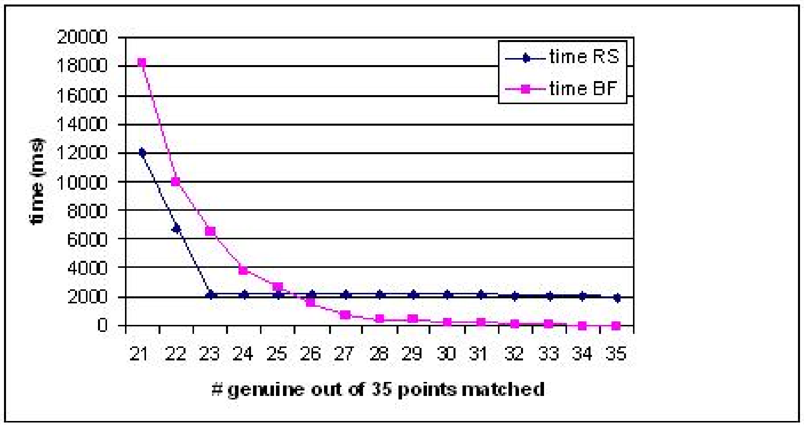


Figure 11: Timing results of two methods with varying number of matched points

The experiments demonstrate that the optimal method changes depending on the number of genuine points matched. The brute force approach is faster if the matching is very good (i.e. most of the minutiae points are matched to genuine points) since it will require very few trials to find the polynomial. On the other hand, the RS decoder is very fast to reject a forgery since brute force will require excessive number of trials to decide on a reject. Also for a weak matching of a valid fingerprint, the RS decoder is again faster.

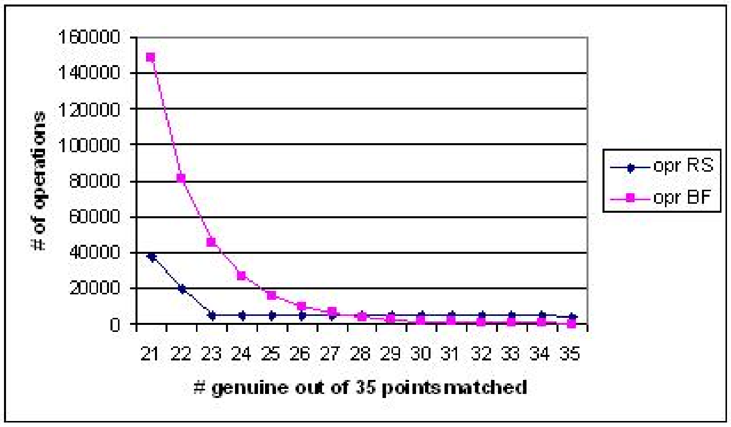


Figure 12: Operational complexities of two methods with varying number of matched points

10 Conclusion

In this thesis, we addressed the implementation, security, and performance issues of *fuzzy vault* for biometric identification. We first provided a guideline that explains the implementation steps of decoding using two methods: brute force and Reed-Solomon (RS) in a detailed way. We then compared the efficiencies of the two methods. The results shows that for weak matching, the RS decoding method is more efficient; but for good matching, the brute force method works faster. For the success rate of verification, they both give the same false accept rate (FAR) and false reject rate (FRR).

We proposed a new chaff point placement method to prevent some inferences on the location of genuine points. By adjusting the distance between points (genuine-to-chaff and chaff-to-chaff) we showed that it is possible to increase security of the *fuzzy vault* implementation.

We explored the effects and limitations of the vault size on the security and performance of the *fuzzy vault*. The higher number of chaff points in the vault is demonstrated to strengthen the method against the most successful attack. However, placing more chaff points takes more time and becomes impossible after a certain number of points and has an adverse effect on the FRR rate.

We proposed to embed a number of fake polynomials in the *fuzzy vault* along with the secret polynomial to reduce the success rate of the attacker. We succeeded in placing only limited number of fake polynomials in the vault. We believe that further research will reveal more efficient methods to place a higher number of fake polynomials in the vault, that will further increase the security.

We also proposed a special hash function that allows us to perform the matching in the hash space. This method makes the correlation attack inapplicable to our proposed *fuzzy vault* system which is shown by the security analysis of this hashing method.

As some suggestions for future work, a matcher with helper data can be implemented to observe the effects of this matcher to the efficiency and successful authentication rates. Another issue is considering a hybrid solution instead of brute force and Reed Solomon. This hybrid version might run brute force a small predefined number of times, say ℓ, at most; if the fingerprint is accepted the program terminates; otherwise run the *RSDecoder* to decide. Moreover, whether this public helper data leaks any information about the biometric should also be considered and detailed analysis should be provided.

References

[1] N. Z. Police, "Fingerprint sections," 2007. http://www.police.govt.nz/service/fingerprint/.

[2] J.-F. Mainguet, "Fingerprint,palmprint, pores," 2008. http://pagesperso-orange.fr/fingerchip/biometrics/types/fingerprint.htm.

[3] Juels and M. Sudan, "Fuzzy vault scheme," in *IEEE International Symposium on Information Theory*, p. 408, 2002.

[4] P. Reid, *Biometrics for Network Security.* Prentice Hall, 2004.

[5] U. Uludag and A. Jain, "Securing fingerprint template: Fuzzy vault with helper data," in *Proc. IEEE Workshop on Privacy Research in Vision (PRIV)*, (NY, USA), June 2005.

[6] A. Kholmatov and B. Yanikoglu, "Realization of correlation attack against fuzzy vault," in *Security, Forensics, Steganography and Watermarking of Multimedia Contents X, Electronic Imaging*, (San Jose CA, USA), 2008.

[7] P. Mihailescu, "The fuzzy vault for fingerprints is vulnerable to brute force attack." http://arxiv.org/abs/0708.2974v1, 2007.

[8] R. M. Roth, *Introduction to Coding Theory.* Cambridge University Press, 2006.

[9] Juels and M. Wattenberg, "A fuzzy commitment scheme," in *ACM Conferance on Computer and Communications Security*, (New York, USA), pp. 28–36, 1999.

[10] C. Soutar, D. Roberge, S. A. Stojanov, R. Gilroy, and B. V. K. V. Kumar, "Biometric encryption," *ICSA Guide to Cryptography*, 1999.

[11] Shamir, "How to share a secret," *Communications of the ACM 22*, pp. 612–613, 1979.

[12] C. Clancy, N. Kiyavash, and D. Lin, "Secure smartcard - based fingerprint authentication," in *ACM Workshop on biometric methods and applications, (WBMA)*, November 2003.

[13] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: how to generate strong keys from biometrics and other noisy data," in *Int. Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pp. 523–540, 2004.

[14] U. Uludag, S. Pankanti, and A. Jain, "Fuzzy vault for fingerprints," in *Proceeding of International Conference on Audio- and Video-Based Biometric Person Authentication*, pp. 310–319, 2005.

[15] K. H. Rosen, *Elementary Number Theory and its applications.* Pearson Addison Wesley, 2005.

[16] S. Yang and I. Verbauwhede, "Automatic secure fingerprint verification system based on fuzzy vault scheme," in *Proceeding of IEEE International Conferance on Acoustics, Speech, and Signal Processing*, pp. 609–612, 2005.

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

[18] S. Dass and A. K. Jain, "Fingerprint classification using orientation field flow curves," in *Indian Conferance Computer Vision, Graphics and Image Processing*, pp. 650–655, 2004.

[19] P. Besl and N. McKay, "A method for registration of 3-d shapes," *IEEE Trans. PAMI*, pp. 239–256, 2004.

[20] V. Guruswami and M. Sudan, "Improved decoding of reed-solomon and algebraic-geometric codes," in *FOCS 1998, Symposium on Foundations of Computer Science*, 1998.

[21] M. Sudan, "Decoding of reed solomon codes beyond the error correction bound," *Journal of Complexity*, pp. 180–193, 1997.

[22] Q. Li, Z. Liu, and X. Niu, "Analysis and problems on fuzzy vault scheme," in *2006 International Conference on Intelligent Information Hiding and Multimedia*, pp. 244–250, 2006.

[23] J. Justesen and T. Hoholdt, *A Course in Error-Correcting Codes.* European Mathematical Society, 2004.

[24] A. Kholmatov, B. A. Yanikoglu, E. Savas, and A. Levi, "Secret sharing using biometric traits," in *Biometric Technology For Human Identification III*, vol. 62022006, (Orlando, Florida USA), 2006. In Proceedings of SPIE.

[25] W. J. Scheirer and T. E. Boult, "Cracking fuzzy vaults and biometric encryption," in *IEEE Biometrics Research Symposium at the National Biometrics Consortium Conferance*, 2007.

[26] K. Nandakumar, A. Nagar, and A. K. Jain, "Hardening fingerprint fuzzy vault using password," in *International Conferance on Biometrics*, pp. 927–937, 2007.

[27] K. E. Atkinson, *An Introduction to Numerical Analysis.* John Willey & Sons, 1988.

[28] W. Trappe, *Introduction to Cryptography with Coding Theory.* Pearson Education, 2006.

[29] V. Shoup, "Ntl: A library for doing number theory," 2008. http://www.shoup.net/ntl.