Privacy Preserving Technique for Set-Based Biometric Authentication using Reed-Solomon Decoding

Jesse Hartloff*[1], Avradip Mandal[2], and Arnab Roy[2]

[1]SUNY at Buffalo, Buffalo, NY, USA
hartloff@buffalo.edu
[2]Fujitsu Laboratories of America, Sunnyvale, CA, USA
{amandal, aroy}@us.fujitsu.com

Abstract: In this work, we present a single-factor biometric authentication system that provides template security against an adversarial server while allowing error-tolerant matching. Our approach is to secure templates represented as sets using error-correcting codes and Reed-Solomon decoding. To accomplish this, each element in the set is combined with a random codeword and a secret share is computed using the codeword and a Reed-Solomon based secret sharing scheme. These random codewords provide uncertainty for an attacker, while the genuine user can decode to the correct values for verification. Without a reading from the enrolling biometric the shares will appear random, thus protecting the users biometric. We show implementation results for this system on fingerprints using pairs of minutia points. Our system overcomes many common weaknesses for template security systems including replay attacks, malicious servers, eavesdroppers, and record multiplicity attacks.

1 Introduction

The appeal of using biometrics has led to an increase in their use as a means of identification. With biometric-based authentication, users are not required to remember extraneous passwords or carry tokens such as smartcards. All users effortlessly bring their biometrics with them wherever they go making it an ideal candidate for user-friendly authentication.

However, this increase in use leads to privacy concerns when sharing biometric information with various service providers since it can be difficult to tell if they are trustworthy. The problem becomes severe when using the same biometric to enroll in several different systems. If one of them is not using proper privacy protocols, it can allow your biometric to be revealed and used to access other systems.

To achieve security against malicious servers, we construct a client-based system which is an instantiation of a secure sketch [DORS08] based on Reed-Solomon decoding for error-tolerant secure matching and an authentication protocol that prevents replay attacks. We use the secure sketch as part of a fuzzy extractor [DORS08] to bind a secret to the enrolling biometric reading creating a secure template which is sent to the server. We

*Work done while Jesse Hartloff visited Fujitsu Laboratories of America.

note that our primitives do not satisfy the stringent cryptographic requirements outlined in [Boy04]. In Section 4, we independently argue why our scheme still should be secure based on reasonable assumptions.

By having a client-centered system, we gain some valuable security properties. Since the only biometric related information that leaves the client is in the form of a secure template, there is no need to trust the server or to secure the communication channels for enrollment or verification. Also, since the client controls the generation of the template they can alter the protocol if they wish. Changing the system to utilize different modalities or adding a user-specific key can be done by the client without the server even being aware of the clients protocol, thus adding security and flexibility to the system. We prevent replay attacks by utilizing a secure signature of random values for each authentication.

We implemented this system on fingerprints using the publicly available FVC2002-DB1 [MMC$^+$02] dataset and report the results in Section 5. To construct a template from a fingerprint, we extract a set of pairs of minutia points and quantize each pair. This provides a set of features that is used to construct the secure templates. These features are combined in a secret sharing scheme such that an attacker cannot attack individual templates points but must correctly match, or guess, a subset of features with size depending on the degree of the underlying polynomial. Since all matching occurs at the client where the fingerprint is read, we have access to the full fingerprint image of the test reading and utilize this during verification.

2 Related Work

There have been various systems proposed that provide template security for fingerprints. Possibly the most popular of them is the fuzzy vault construct of Juels and Sudan [JS06]. Similar to our current system, the fuzzy vault binds a secret to a set of values and releases the secret using Reed-Solomon decoding given a set that is sufficiently similar to the one used during enrollment. The security of the fuzzy vault relies on adding many randomly generated chaff points to obfuscate the enrolling data. The fuzzy vault has been implemented in various forms to secure fingerprint templates [BCF12, JA07, NNJ08], however the fuzzy vault is vulnerable to various attacks including record multiplicity, chaff injection, and replay attacks [KY08, MNS$^+$10, MMT09, PM09, SB07] and can have very large template sizes due to the addition of enough chaff points for sufficient security. Our current system overcomes all of these shortcomings.

Many protocols combine cryptographic techniques with biometrics to form a secure biometric cryptosystem [BBCdS08, BCI$^+$07, Sto10]. However, these schemes have been shown to have vulnerabilities especially when a malicious server is considered instead of the common honest but curious server [SBCS12].

Our work develops a new method that falls under the category of secure sketch [DORS08] which we use as part of a fuzzy extractor [DORS08] to bind a secret to the enrolling fingerprint. A general theoretical framework for an authentication scheme using secure sketch is given in [BDK$^+$05]. Our scheme can be considered as a concrete instance of a fingerprint matching scheme that generalizes to the abstract scheme from [BDK$^+$05] which utilizes a secure sketch and an authentication scheme, both of which are treated as black boxes. The scheme also utilizes client-based computation to further protect the biometric data. Another instance of this scheme has been implemented for face biometrics [SLM07].

There are many other proposed systems for fingerprint template security including [BSW07] and [FMC12], both of which report more accurate matching performance that ours. We note that in [BSW07], the system stores partial template information in the clear to compute a robust distance measure that improves matching accuracy. A security analysis shows that the remaining entropy in the template is still sufficient for security, though an attacker is given some information of the enrolling fingerprint. Our proposed system does not reveal any biometric information. The security of [FMC12] is based on two attacks implemented by the authors that attempt to recover information of the enrolling fingerprint from the stored template. Results are show with parameters for which these attacks have a success rate of 0.

We show that in the proposed system, the enrolling fingerprint information is secure against any attack, including an adversarial server, under the assumption of uniformity of the template points. Addressing this assumption is a source of continuous research.

3 System Protocol

Our system consists of two phases - an Enrollment phase which is one-time for a single user and server pair and an Authentication phase which can be executed for every session of a user with the server. There are three parties involved in the Enrollment phase - the user, the server and a Trusted Third Party, while in the Authentication phase only the user and the server are involved. We describe the phases in detail below. The only role of the Trusted Third Party is to verify that the server stores the correct template in tact, thus preventing a man-in-the-middle attack.

3.1 Enrollment Phase

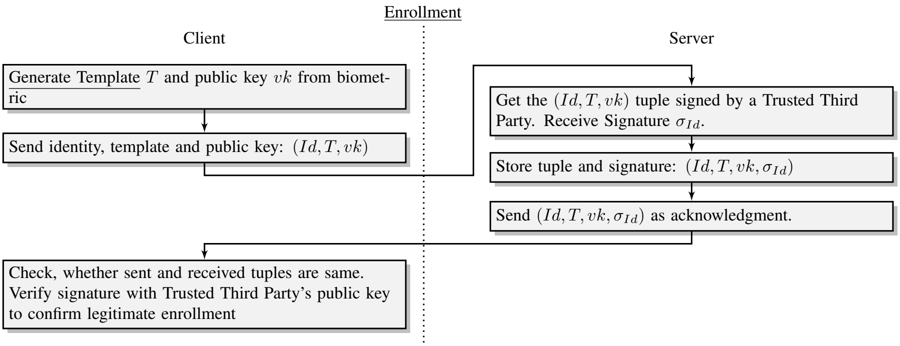

Enrollment
Client
Server
Generate Template T and public key vk from biometric
Send identity, template and public key: (Id, T, vk)
Get the (Id, T, vk) tuple signed by a Trusted Third Party. Receive Signature σ_{Id}.
Store tuple and signature: (Id, T, vk, σ_{Id})
Send (Id, T, vk, σ_{Id}) as acknowledgment.
Check, whether sent and received tuples are same. Verify signature with Trusted Third Party's public key to confirm legitimate enrollment

Figure 1: Enrollment process

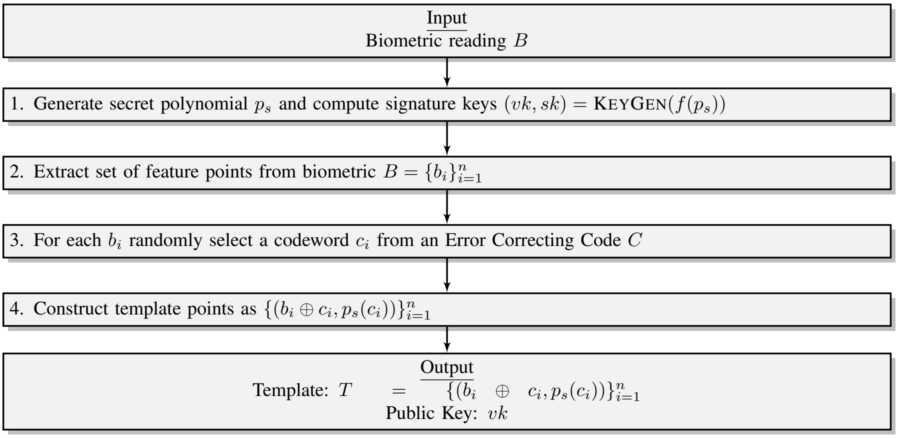

Input
Biometric reading B
1. Generate secret polynomial p_s and compute signature keys $(vk, sk) = \textsc{KeyGen}(f(p_s))$
2. Extract set of feature points from biometric $B = \{b_i\}_{i=1}^n$
3. For each b_i randomly select a codeword c_i from an Error Correcting Code C
4. Construct template points as $\{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^n$
Output
Template: $T = \{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^n$
Public Key: vk

Figure 2: Generate template

Enrollment consists of a user generating a random secret and binding it to her biometric to create a secure template. The secret is used as a seed to generate a public verification key using a signature scheme and will be used to generate the corresponding private signature key during verification. The template and public key are then sent to server for storage.

The template and public key together with the user's identity are then digitally signed by a Trusted Third Party. This signature is sent back to the client for verification. See Figure 1.

Generating Template. As part of the enrollment process, the user will generate a secure template using a random secret and a biometric reading by following the steps in Figure 2 which are described below. We assume that the reading is a set of bit-vectors b_i of identical length, where i runs from 1 to n, which is true in our implementation in Section 5. To tolerate errors in reading each bit vector, we use an Error Correcting Code (ECC) C, regarded as a set of codewords. The length of codewords is chosen to be the same as an individual bit-vector b_i.

1. User generates a random secret that will be used as a seed to generate a key pair of a secure signature scheme. For use in our system, this secret is encoded as a polynomial p_s. We use a cryptographic hash function (e.g. SHA-3) f and a secure signature scheme (e.g. PKCS #1) $(\textsc{KeyGen}, \textsc{Sign}, \textsc{Verify})$ in our system. The key pairs are generated as follows, by using $f(p_s)$ as source of randomness for $\textsc{KeyGen}$.
$$(vk, sk) = \textsc{KeyGen}(f(p_s))$$

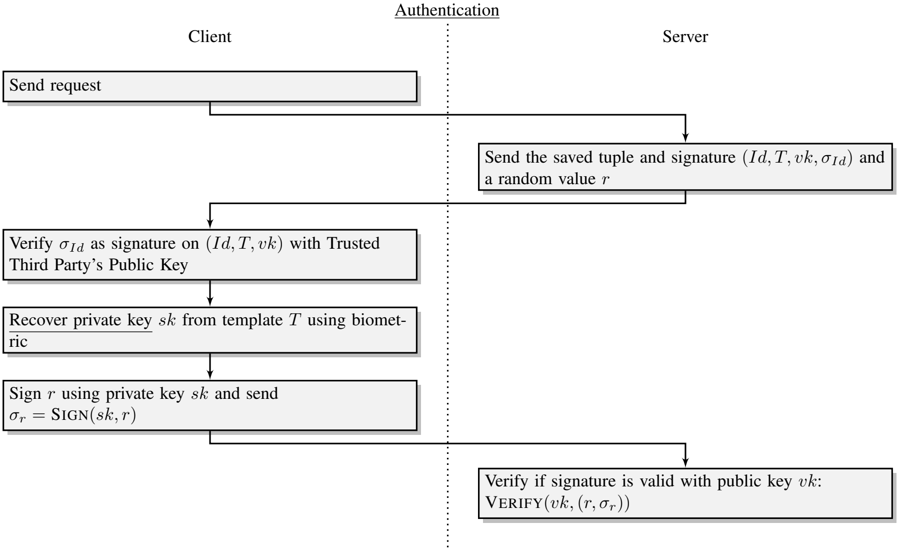
2. Extract a set (B) of feature points from a biometric reading, which is a set of bit-vectors b_i of identical length, where i runs from 1 to n. Please see Section 5 for a concrete example.

3. Choose a random codeword c_i from a code C for each $b_i \in B$. The c_i values

will be used to hide the template data while allowing some error-correction during verification.

4. Compute $y_i = b_i \oplus c_i$ and $\gamma_i = p_s(c_i)$ for each $i \in [n]$ and store as the secure template $T = \{(y_1, \gamma_1), \dots, (y_n, \gamma_n)\}$. The polynomial evaluations will be used as input in a Reed-Solomon decoder to correct for errors.

3.2 Authentication Phase



Send request

Send the saved tuple and signature (Id, T, vk, σ_{Id}) and a random value r

Verify σ_{Id} as signature on (Id, T, vk) with Trusted Third Party's Public Key

Recover private key sk from template T using biometric

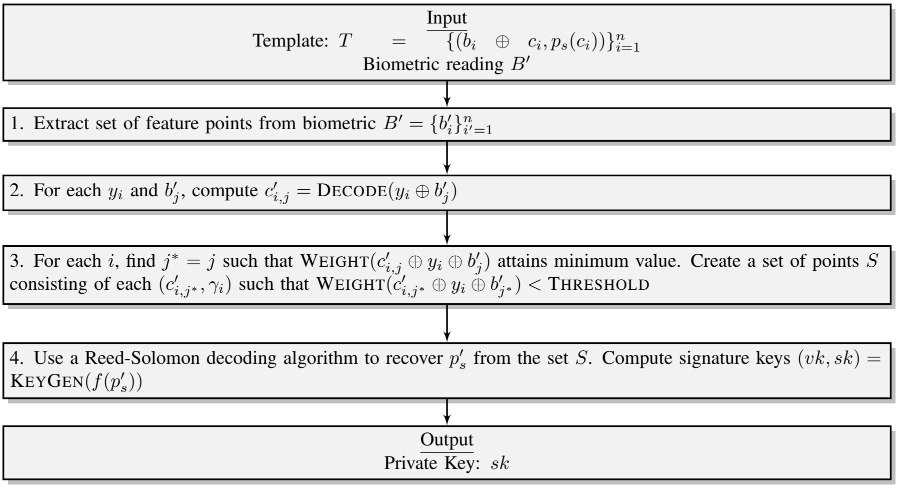
Sign r using private key sk and send $\sigma_r = \textsc{Sign}(sk, r)$

Verify if signature is valid with public key vk: $\textsc{Verify}(vk, (r, \sigma_r))$

Figure 3: Authentication

To authenticate, the server sends the signed tuple back to the user along with a random value r. If the user is legitimate, she will be able to recover the secret from the template using her biometric and generate the private signature key to be used to sign r. The signature on r is sent back to the server where it is verified using the public verification key. See Figure 3.

Key Recovery. During verification, the client must recover the secret signature key from the secure template and use it to sign the random value r sent by the server. Since the key is recovered at each verification, the user is not required to remember it making this a single-factor system. This signature is sent back to the server to complete the verification process. Below are descriptions of the steps outlined in Figure 4.



Input

Template: $T = \{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^n$

Biometric reading B'

1. Extract set of feature points from biometric $B' = \{b'_i\}_{i'=1}^n$

2. For each y_i and b'_j, compute $c'_{i,j} = \textsc{Decode}(y_i \oplus b'_j)$

3. For each i, find $j^* = j$ such that $\textsc{Weight}(c'_{i,j} \oplus y_i \oplus b'_j)$ attains minimum value. Create a set of points S consisting of each (c'_{i,j^*}, γ_i) such that $\textsc{Weight}(c'_{i,j^*} \oplus y_i \oplus b'_{j^*}) < \textsc{Threshold}$

4. Use a Reed-Solomon decoding algorithm to recover p'_s from the set S. Compute signature keys $(vk, sk) = \textsc{KeyGen}(f(p'_s))$

Output

Private Key: sk

Figure 4: Recover private key

1. Extract set of feature points $\{b'_i\}_{i=1}^n$ from biometric reading B' using the same method as for enrollment.

2. Client computes $c'_{i,j} = \textsc{Decode}(y_i \oplus b'_j)$ for each (y_i, b'_j) pair. Here $\textsc{Decode}$ outputs the nearest codeword from $(y_i \oplus b'_j)$.

3. For each i, client chooses $j^* = j$, such that $\textsc{Weight}(c'_{i,j} \oplus y_i \oplus b'_j)$ attains minimum value for j between 1 to n. Create a set of points S consisting of each (c'_{i,j^*}, γ_i) such that $\textsc{Weight}(c'_{i,j^*} \oplus y_i \oplus b'_{j^*}) < \textsc{Threshold}$.

4. Use Reed-Solomon decoding on S to recover p'. By the properties of the Reed-Solomon decoder, if the number of genuine points in the S minus the number of false points in S is greater than the degree of p_s, then $p' = p_s$.

4 Security

4.1 Template privacy against Brute Force Attacker

We first show that the proposed system is secure against a basic brute force attack before proceeding to formally show its security. A brute force attacker against our protocol from the previous section has access to the template $T = \{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^n$ and public key vk, where $(vk, sk) = \textsc{KeyGen}(f(p_s))$. The goal of the attacker is to recover the biometric template $B = \{b_i\}_{i=1}^n$ (or another biometric template close to B). This problem is equivalent to guessing the random codewords $\{c_i\}_{i=1}^n$ and testing the correctness of guess from $\{p_s(c_i)\}_{i=1}^n$ and vk. If an attacker can correctly guess the random polynomial p_s, it can easily find the codewords from $p_s(c_i)$ values. If the polynomial p_s is of degree t, then to find out the polynomial p_s, the brute force attacker has to make correct guesses for c_i values simultaneously for at least $(t+1)$ points. These random guesses can be interpolated to a possible guess for the polynomial as p'_s. The only way the attacker can check whether the random guesses for c_i values (equivalently, random guess for the polynomial p_s) are correct or not, is by running the KeyGen algorithm on $f(p'_s)$ and checking whether the resultant verification key is same as the published verification key vk or not. If we sample the codewords from a (n, k, d) error correcting code, then each codeword c_i has k bits of entropy and the attacker has to make simultaneous guesses to at least $(t+1)$ codewords. Hence our protocol has $k(t+1)$ bits of security against brute force attackers.

4.2 Formal Security Guarantees

The protocol described in Section 3 provides the following security guarantees:

- **Type-I Security:** It is a secure authentication protocol, i.e., only legitimate users can get authenticated to the server.
- **Type-II Security:** It protects user's biometric data against malicious servers. If multiple servers are authenticating the users using our protocol and one of them is acting maliciously, even then the malicious server cannot authenticate to another server on behalf of any common user (a person who has enrolled to both servers).

To show that our scheme is secure in both the cases, we consider a powerful adversary $\mathcal{A}$, which

1. Has access to the template
$$T = \{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^{n}$$
2. Has access to the public verification key vk, such that $(vk, sk) = \textsc{KeyGen}(f(p_s))$
3. Can send any r of its choice[1] to the client and receive σ_r, which is a valid signature of r (gets verified by the verification key vk).

The goal of the above adversary is to come up with a valid message, signature pair $(\hat{r}, \hat{\sigma_r})$ which would get verified by the verification key vk, without querying $\hat{r}$ to the client[2]. Such an adversary mimics a dishonest server trying to authenticate to another server (Type-II attacker), as well as a powerful man in the middle attacker (Type-I attacker).

If the signature scheme is chosen message secure, then security of our protocol can be based upon the following assumption, which we later argue to be reasonable.

[1] In the actual protocol the server also sends the enrolled template T along with its signature signed by a trusted third party. Client verifies integrity of the template by verifying the signature with trusted third parties public key, before reconstructing its private key based on the template. This forces the attacker to send the same T to evoke a response.

[2] This actually provides a stronger security guarantee. In the actual protocol execution, the attacker has to come up with a valid signature of some $\hat{r}$, chosen by the honest server, not an $\hat{r}$ chosen by the attacker herself.

Assumption 1. *If f is a cryptographic hash function, $B = \{b_1, \cdots, b_n\}$ and $B' = \{b'_1, \cdots, b'_n\}$ are two sets of feature points corresponding to fingerprints of two different individuals (Id and Id'), $(c_1, \cdots, c_n)$ and $(c'_1, \cdots, c'_n)$ are random code-words, and p_s, p'_s are two randomly selected polynomials, then the following two tuples are indistinguishable to a computationally bounded adversary:*

$$(Id, f(p_s), \{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^{n}) \\ \approx (Id, f(p_s), \{(b'_i \oplus c'_i, p'_s(c'_i))\}_{i=1}^{n})$$

The above assumption says that biometric templates corresponding to two different individuals are indistinguishable, as well as it is infeasible to correlate the output of the hash function $f(p_s)$ and the biometric template T. We now prove that, under this assumption, the only way the adversary $\mathcal{A}$ can be successful is to break the security of the signature scheme.

Theorem 1. *If* $(\text{KeyGen}, \text{Sign}, \text{Verify})$ *is a signature scheme which is* existentially unforgeable under chosen message attack *(EU-CMA), and Assumption 1 holds true, then adversary $\mathcal{A}$ can win only with negligible probability.*

Proof. Using adversary $\mathcal{A}$, we construct an EU-CMAadversary $\mathcal{B}$ against the signature scheme. The EU-CMAchallenger will generate verification key vk and signing key sk by running KeyGen. Adversary $\mathcal{B}$ will receive the verification key vk from the EU-CMAchallenger. The EU-CMAchallenger will also provide access to the signing oracle $\text{Sign}(sk, \cdot)$ to the adversary $\mathcal{B}$. Adversary $\mathcal{B}$ works as follows:

1. Sample fingerprint $B' = \{b_1, \cdots, b'_n\}$ from a random individual. Sample random codewords $\{c'_1, \cdots, c'_n\}$ and random polynomial p'_s. Send vk and $T' = \{(b'_i \oplus c'_i, p'_s(c'_i))\}_{i=1}^{n}$ to the adversary $\mathcal{A}$. Assumption 1 says, adversary $\mathcal{A}$ would not be able to distinguish the simulated (vk, T') from (public key of $\text{KeyGen}(f(p_s))$, $\{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^{n}$) received in the real protocol.

2. For every signature query r sent by adversary $\mathcal{A}$, $\mathcal{B}$ can forward the query to the EU-CMAchallenger.

3. In the end, a successful $\mathcal{A}$ would provide a valid message signature tuple $(\hat{r}, \hat{\sigma_r})$ which was not obtained as the response to a query. $\mathcal{B}$ can send the same tuple to the EU-CMAchallenger and provide a valid forgery.

□

Justification of Assumption 1. Assumption 1 plays a key role in our security proof. Assuming the hash function f behaves as a random oracle, Theorem 2 stated below reduces Assumption 1 to the following simpler one (independent of the hash function f).

Assumption 2. *If $B = \{b_1, \cdots, b_n\}$ and $B' = \{b'_1, \cdots, b'_n\}$ are two sets of feature points corresponding to fingerprints of two different individuals (Id and Id'), $(c_1, \cdots, c_n)$ and $(c'_1, \cdots, c'_n)$ are random code-words, and p_s, p'_s are two randomly selected polynomials, then*

- *The following two tuples are indistinguishable:*

$$(Id, \{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^{n}) \approx (Id, \{(b_i' \oplus c_i', p_s'(c_i'))\}_{i=1}^{n})$$

- *Given* $\{(b_i \oplus c_i, p_s(c_i))\}_{i=1}^{n}$*, it is hard to output* p_s

Assumption 2 consists of two parts, of which the first one says that if we XOR biometric feature bit-vectors with random codewords, the xored bit-vectors corresponding to two different individuals become indistinguishable. Moreover, indistinguishability continues to hold when we additionally provide random polynomial evaluations corresponding to those random codewords. As a quick sanity check, if we assume the b_i's are coming from a uniform distribution and p_s is a linear polynomial our assumption holds provably. We claim that even if p_s is a higher degree polynomial and the b_i's are coming from an actual fingerprint distribution, our assumption still holds. The second part of the assumption says xoring the random codewords with biometric minutia points, hides the codewords to sufficient degree that it is impossible to recover p_s given the evaluations $\{p_s(c_i)\}_{i=1}^{n}$.

Theorem 2. *If f is a random oracle, then Assumption 2 implies Assumption 1.*

Proof. We show that if there is an Assumption 1 adversary $\mathcal{A}_1$, which succeeds with non negligible advantage then we can construct an Assumption 2 adversary $\mathcal{A}_2$ succeeding with non negligible advantage. From the Assumption 1 challenger, $\mathcal{A}_2$ receives the identity of an individual Id and a biometric template T (using a coin flip, T was generated by the biometric corresponding to either the individual Id or from another different individual). $\mathcal{A}_2$ wins if

1. It can correctly guess whether T was generated using the biometric corresponding to individual Id, or
2. It can output the polynomial p_s used during the generation of T

$\mathcal{A}_2$ works as follows:

1. Sample random r from the range of f. Send (Id, r, T) to $\mathcal{A}_1$
2. For each different random oracle query $f(p_i)$ made by $\mathcal{A}_1$, answer by sampling a random r_i from the range of f. Save (p_i, r_i) in a table.
3. In the end $\mathcal{A}_1$ will return its guess, whether T was generated using Id's biometric or not. Send the same guess to Assumption 2 challenger along with a random p out of $\{p_i\}$ (random oracle queries made by $\mathcal{A}_1$) as a guess for p_s.

f being a random oracle, (Id, r, T) is a valid Assumption 1 challenge, as long as the p_s used in generation of T does not belong to the set $\{p_i\}$. In that case, whenever $\mathcal{A}_1$ makes a successful guess, $\mathcal{A}_2$ also makes a successful guess and wins against the Assumption 2 challenger. In the other case, $\mathcal{A}_1$ being an efficient adversary can only make polynomially many $f(p_i)$ queries and one of those p_i's is actually p_s. Hence, $\mathcal{A}_2$ can successfully guess p_s with $1/\mathtt{poly}()$ probability, which is non-negligible. □

5 Experimental Results

As a proof of concept of the feasibility of our system, we implemented a simple fingerprint matching method using pairs of minutia points. These template values consist of the concatenation of the distance between the points, the difference in their orientations, the angle of the line defined by the points, the angle between the orientation and the connecting line, the number of ridges between the points, and the type of each minutia in the pair. Each value is then quantized and the gray code is applied to the quantized values. This encoding has the property that any two consecutive integers will have hamming distance 1. This allows us to treat the integers as bit strings and enables the use of hamming distance to compare similar minutia pairs. After quantization, these template values are 22 bits in length. Since the angle of the line connecting the points is rotation variant, we consider several different rotations of the test fingerprint during verification. We only compute the enrolling template at a single rotation to limit the amount of information and correlation in the template.

We use a (22, 6, 4) randomly generated code to protect the template points for this implementation. This code has 64 codewords which provides 6-bits of entropy that an attacker would have to guess in order to recover the template point. Since there is no feedback for an attack on a single point, an attacker would have to simultaneously correctly guess enough points to recover the secret polynomial. At the ZeroFAR, this polynomial has degree 16 meaning an attacker must simultaneously guess at least 17 points each with 6 bits of entropy resulting in $6 * 17 = 102$ bits of security against a brute force attack as described in Section 4.1. To decode this polynomial we use the Welch-Berlekamp decoder for the Reed-Solomon code.

The choice of code provides a tradeoff between security and matching accuracy. For this implementation, we only utilize 1 bit of error correction, meaning we could use a more efficient code with more than 64 codewords to increase entropy while still being able to correct from 1 bit errors. However, increasing the number of codewords also increases the number of false matches on the template points since it is more likely that a random value will be within 1 bit of a codeword. Thus, the tradeoff between accuracy and security can be adjusted by altering the size of the code.

To match a template with a fingerprint reading, we first extract a set of pairs from the enrolled template using the method from Section 3.2. To increase the accuracy of genuine matches, we filter out some of the extracted template points by only considering sets of points that form a complete sub-graph of at least 4 minutia points. This reduces the chance that a false match will be considered.

We use the FVC (Fingerprint Verification Competition) style of measuring results with 2800 genuine and 4950 impostor tests. Results are reported for FVC2002-DB1. Minutiae points were obtained using the open-source minutiae extraction method MINDTCT [WGT$^+$] published by NIST.

A summary of our matching results can be found in table 1. Since security is a focus of our system, we are concerned with large polynomials that lead to no false accepts.

Degree of Secret Polynomial	FAR (%)	FRR (%)
14	0.08	19.0
15	0.04	20.3
16	0.0	21.4

Table 1: Matching results for FVC2002-DB1. We note that the system should not be used when the FAR is not 0.0 as this compromises security. We include additional values to give more information on the matching performance of the system.

6 Conclusion

In this work, we presented a fingerprint matching system that provides template security against an adversarial server by utilizing the entropy of random codewords in conjunction with polynomial based secret sharing. We accomplish this in part by shifting the matching responsibility to the client instead of the server. We also include a signature scheme for verification that prevents replay attacks from potential eavesdroppers. In addition to presenting this novel theoretical system, we provide the results of an implementation based on pairs of minutiae points to show the feasibility of this system in practice.

References

[BBCdS08] Manuel Barbosa, Thierry Brouard, Stphane Cauchie, and SimoMelo de Sousa. Secure Biometric Authentication with Improved Accuracy. In Yi Mu, Willy Susilo, and Jennifer Seberry, editors, *Information Security and Privacy*, volume 5107 of *Lecture Notes in Computer Science*, pages 21–36. Springer Berlin Heidelberg, 2008.

[BCF12] Julien Bringer, Herv Chabanne, and Mlanie Favre. Fuzzy Vault for Multiple Users. In Aikaterini Mitrokotsa and Serge Vaudenay, editors, *Progress in Cryptology - AFRICACRYPT 2012*, volume 7374 of *Lecture Notes in Computer Science*, pages 67–81. Springer Berlin Heidelberg, 2012.

[BCI$^+$07] Julien Bringer, Herv Chabanne, Malika Izabachne, David Pointcheval, Qiang Tang, and Sbastien Zimmer. An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication. In Josef Pieprzyk, Hossein Ghodosi, and Ed Dawson, editors, *Information Security and Privacy*, volume 4586 of *Lecture Notes in Computer Science*, pages 96–106. Springer Berlin Heidelberg, 2007.

[BDK$^+$05] Xavier Boyen, Yevgeniy Dodis, Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Secure Remote Authentication Using Biometric Data. In Ronald Cramer, editor, *Advances in Cryptology EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 147–163. Springer Berlin Heidelberg, 2005.

[Boy04] Xavier Boyen. Reusable cryptographic fuzzy extractors. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick Drew McDaniel, editors, *Proceedings of the 11th ACM Conference on Computer and Communications Security, CCS 2004, Washington, DC, USA, October 25-29, 2004*, pages 82–91. ACM, 2004.

[BSW07] Terrance E. Boult, Walter J. Scheirer, and Robert Woodworth. Revocable Fingerprint Biotokens: Accuracy and Security Analysis. In *CVPR*, 2007.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[FMC12] M. Ferrara, D. Maltoni, and R. Cappelli. Noninvertible Minutia Cylinder-Code Representation. *Information Forensics and Security, IEEE Transactions on*, 7(6):1727–1737, Dec 2012.

[JA07] Jason Jeffers and Arathi Arakala. Fingerprint Alignment for A Minutiae-Based Fuzzy Vault. In Arathi Arakala, editor, *Biometrics Symposium, 2007*, pages 1–6, 2007.

[JS06] Ari Juels and Madhu Sudan. A Fuzzy Vault Scheme. *Des. Codes Cryptography*, 38(2):237–257, 2006.

[KY08] Alisher Kholmatov and Berrin Yanikoglu. Realization of correlation attack against the fuzzy vault scheme. In *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, volume SPIE 6819, pages 68190O–68190O–7, 2008.

[MMC+02] D. Maio, D. Maltoni, R. Cappelli, J.L. Wayman, and A.K. Jain. FVC2002: Second Fingerprint Verification Competition. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 3, pages 811–814 vol.3, 2002.

[MMT09] Preda Mihailescu, Axel Munk, and Benjamin Tams. The Fuzzy Vault for Fingerprints is Vulnerable to Brute Force Attack. In Arslan Brömme, Christoph Busch, and Detlef Hühnlein, editors, *BIOSIG*, volume 155 of *LNI*, pages 43–54. GI, 2009.

[MNS+10] Johannes Merkle, Matthias Niesing, Michael Schwaiger, Heinrich Ihmor, and Ulrike Korte. Security Capacity of the Fuzzy Fingerprint Vault. *International Journal on Advances in Security*, 3(3 & 4):146–168, 2010.

[NNJ08] Abhishek Nagar, Karthik Nandakumar, and Anil K. Jain. Securing fingerprint template: Fuzzy vault with minutiae descriptors. In *ICPR*, pages 1–4, 2008.

[PM09] Hoi Ting Poon and Ali Miri. A Collusion Attack on the Fuzzy Vault Scheme. *ISeCure, The ISC International Journal of Information Security*, 1(1):27–34, 2009.

[SB07] W.J. Scheirer and T.E. Boult. Cracking Fuzzy Vaults and Biometric Encryption. In *Biometrics Symposium, 2007*, pages 1 –6, sept. 2007.

[SBCS12] K. Simoens, J. Bringer, H. Chabanne, and S. Seys. A Framework for Analyzing Template Security and Privacy in Biometric Authentication Systems. *Information Forensics and Security, IEEE Transactions on*, 7(2):833–841, April 2012.

[SLM07] Y. Sutcu, Qiming Li, and N. Memon. Protecting Biometric Templates With Sketch: Theory and Practice. *Information Forensics and Security, IEEE Transactions on*, 2(3):503–512, Sept 2007.

[Sto10] A. Stoianov. Cryptographically secure biometrics. volume 7667, pages 76670C–76670C–12, 2010.

[WGT+] Craig I. Watson, Michael D. Garris, Elham Tabassi, Charles L. Wilson, R. Michael Mccabe, Stanley Janet, and Kenneth Ko. User's Guide to NIST Biometric Image Software (NBIS).