Fuzzy Extractors for Minutiae-Based Fingerprint Authentication

Arathi Arakala*, Jason Jeffers, and K.J. Horadam

School of Mathematical and Geospatial Sciences
RMIT University
368-374 Swanston Street, Melbourne 3000
{arathi.arakala,jason.jeffers,Kathy.Horadam}@ems.rmit.edu.au

Abstract. We propose an authentication scheme using fingerprint biometrics, protected by a construct called a Fuzzy Extractor. We look at a new way of quantizing and digitally representing the minutiae measurements so that a construct called PinSketch can be applied to the minutiae. This is converted to a Fuzzy Extractor by tying some random information to the minutiae measurements. We run a matching algorithm at chosen quantization parameters and show that the authentication accuracy is within acceptable limits. We demonstrate that our authentication system succeeds in protecting the users' identity.

1 Introduction

Storage of an un-encrypted enrolled biometric template in a database for authentication is a privacy and security threat (demonstrated by [1] for the fingerprint biometric). Several primitives have been proposed to protect biometric templates by performing template comparison in the encrypted domain ([2], [3], [4], [5] and [6]). Prior work in building practical fingerprint authentication systems using the above ideas ([7] and [8]) use a frequency domain approach, either partially or completely to handle pre-alignment issues . Jeffers and Arakala [9] have studied the feasibility of using translation and rotation invariant structures to build a pre-alignment free Fuzzy Vault using minutiae information alone. In this paper, we extend that work to perform pre-alignment free matching and use Fuzzy Extractors to execute the matching in the encrypted domain.

2 Fuzzy Extractors

A Fuzzy Extractor is a cryptographic construct defined in [6], which enables two inputs to be compared in the encrypted domain and returns a successful match if the two inputs are within a limited distance from each other in a specific metric space. It is a combination of a primitive called a *Secure Sketch* and a *Strong*

* This paper is for the BBSPA competition. It forms part of the PhD thesis of the first author, taken under the supervision of the other two authors.

S.-W. Lee and S.Z. Li (Eds.): ICB 2007, LNCS 4642, pp. 760–769, 2007.

Randomness Extractor. During the enrolment of an error prone, non uniformly distributed input, the Secure Sketch generates some public information related to the input called a 'Sketch' which by itself cannot be used to recover the input. The Randomness Extractor is used to map the non uniform input to a uniformly distributed string. The seed for the Randomness Extractor along with the Sketch will be stored publicly. When a query must be matched to the input, the Fuzzy Extractor uses the public Sketch of the input along with the query to reconstruct the input exactly. The Fuzzy Extractor's reconstruction procedure will be designed such that if the query is within a specified distance from the input, the reconstruction will succeed. The reconstructed input will then be mapped to the same string using the same seed stored along with the Sketch. The public storage of the Sketch and the seed do not substantially compromise the security of the input as they cannot be used to recover the input without a query which is 'close' to the input. In this paper, we build our Fuzzy Extractor using a Secure Sketch construct called PinSketch defined in [6], which is based on the set difference metric in a large universe setting. We convert this to a Fuzzy Extractor by using a a simple pairwise independent hash function.

PinSketch performs error correction using binary $[n, k, \delta]$ BCH codes where n is the number of bits in the codeword, k is the number of bits in the message and δ is the minimum distance of the codeword. Each input X will be a word of length $n = 2^m - 1$. During enrollment, the Syndrome of X denoted by $\text{Synd}(X)$ will be computed and will be stored as the public Sketch of X.When a query X' is to be compared with X, $\text{Synd}(X)$ is retrieved and the following recovery procedure takes place:

- Compute $\text{Synd}(X')$
- Set $\sigma = \text{Synd}(X) - \text{Synd}(X')$
- Find a vector v such that $\text{Synd}(v) = \sigma$ and $|\text{Supp}(v)| \leq ((\delta - 1)/2)$[1]
- If X and X' differ in no greater than $((\delta - 1)/2)$ positions, then $v = X - X'$. We can get back X exactly by computing $X' + v$

3 Our Scheme

The novelty of our scheme is that we perform translation and rotation invariant minutiae-based matching in the encrypted domain. In our scheme, every fingerprint will be defined by the *Descriptor* of each minutia extracted from it. Each minutia's *Descriptor* is comprised of two parts: a vector describing the minutia's global position in a polar coordinate system centered on the core and 5 vectors representing the local positions of the five nearest neighboring minutiae in a polar coordinate system centered on the minutia in consideration. The components of each descriptor will be encrypted using an instance of a Fuzzy Extractor.

[1] A binary vector v can be represented by a list of its non zero positions, called its Support Set, denoted by $\text{Supp}(v)$.

Quantization of the Global Reference Frame. Every minutia, M_i's, global position will be described by its radial distance from the core, R_i and the angle Θ_i made by M_i with respect to the 0° axis, which is aligned with the orientation of the core (see Fig. 1(a)). The polar coordinate frame is quantized with the radial width of each quantum as R_{tol} and the angular range of each quantum as Θ_{tol}. Let the maximum radial measure of the frame be R_{max}.The quanta will be numbered from 1 to GQ_{max} where $GQ_{max} = ((R_{max}/R_{tol}) \cdot (360°/\Theta_{tol}))$ is the total number of quanta in the global reference frame.

Quantization of the Local Reference Frame. Each Minutia M_i will be described by the five distances and five relative angles between a neighbor $n_i^j, 1 \leq j \leq 5$ and itself, in a polar coordinate system with M_i at the origin and M_i's ridge orientation aligned to the 0° axis (see Fig. 1(b)). This reference frame is quantized with each quantum having radial width d_{tol} and angular range θ_{tol}. If d_{max} is the maximum radial distance in this reference frame, the quanta will be numbered from 1 to LQ_{max} where $LQ_{max} = ((d_{max}/d_{tol}) \cdot (360°/\theta_{tol}))$ is the maximum number of quanta in this local reference frame.

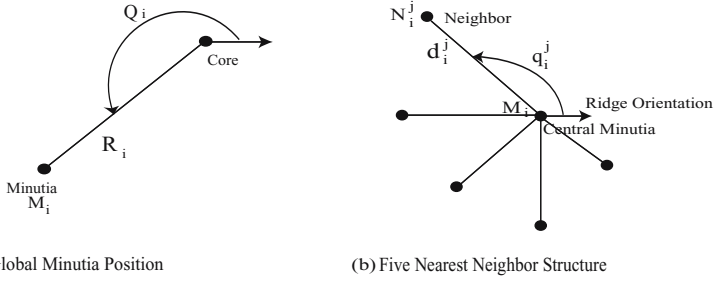


Fig. 1. Global Position and Local Structure of a Minutia

Compensating for Non-Linear Distortion in Minutiae Positions. Each minutia, by virtue of its position in a global or local reference frame, will lie in some quantum q. Non-linear distortions in minutia position may cause it to move to a different quantum when compared to its corresponding minutia in the enrolled fingerprint. We propose a novel scheme to match corresponding minutiae positions when such movement occurs. Every local or global position will be described by a set of quanta. This set, which includes q where a minutia lies and the 8 quanta that surround it is called a *Position Set*. The shaded regions in 2(a) give an example of the elements of the Position Set for a Minutia's global position. If non-linear distortion causes a minutia to move to any one of the eight surrounding quanta, the resulting Position Set will differ from the correct Position set by a maximum of 10 elements. An illustration of this is given in Fig. 2(b), (c) and (d). Hence two minutiae positions are considered to match if the set difference between their Position Sets is less than or equal to 10. For minutiae that lie in the first tier (core is one of the corners) of the quantized frame, a

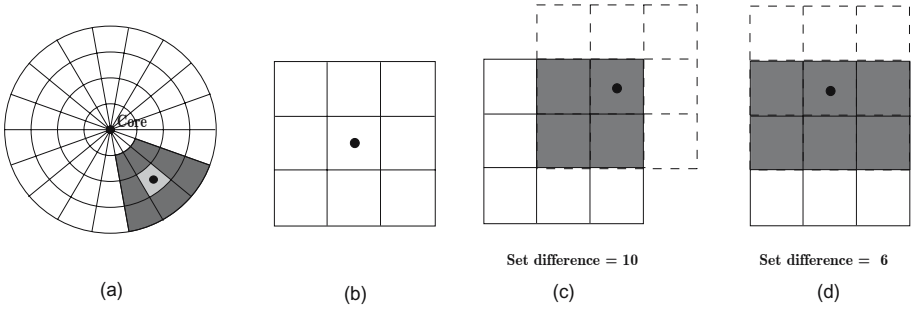


Fig. 2. Error tolerance in minutia position

similar compensation for movement results in a maximum set difference of 7. Thus, by setting the maximum allowable set difference to 10 we permit slightly more position variation in minutiae lying in the inner quanta.

Binary Representation and Matching Condition. The global Position Set of a Minutia M_i will be represented by a binary vector P_i, called its *Position Vector*, having length equal to GQ_{max} and with 1 in every position except those corresponding to the elements in its Position Set. Similarly, each of M_i's neighbors will be represented by the binary Position Vector of length LQ_{max} corresponding to each of their local Position Sets. A minutia in the enrolled fingerprint matches a minutia in the query fingerprint if:

1. The global Position Sets (Position Vectors) of the two minutiae have a set difference (Hamming distance) of not more than 10.
2. At least four out of the five neighbors in their corresponding Five Nearest Neighbor structures will have a set difference (Hamming distance) not more than 10 , when local Position Sets (Position Vectors) of corresponding neighbors are compared.

3.1 Encrypted Matching of Minutiae Points

If N minutiae were extracted from a fingerprint, the un-encrypted fingerprint template will consist of N *Descriptors*, one for each minutia. For a minutia M_i, its Descriptor comprises of a binary global Position Vector P_i and five binary local Position vectors $N_i^j, 1 \leq j \leq 5$. To perform matching in the encrypted domain, each global Position Vector is protected by using a Fuzzy Extractor realization called FE_{global} . Each set of five local Position Vectors is also protected by a Fuzzy Extractor realization called FE_{local}. The block diagrams of FE_{global} and FE_{local} are depicted in Fig. 3 and Fig. 4 respectively.

FE_{global} **Enrolment:** The enrolment stage consists of 3 modules - An XOR module, the PinSketch encoding module and a pairwise independent hash function module. In order to encrypt the global position of minutia M_i, P_i is given as input to FE_{global}. A codeword C is randomly chosen from a BCH Code

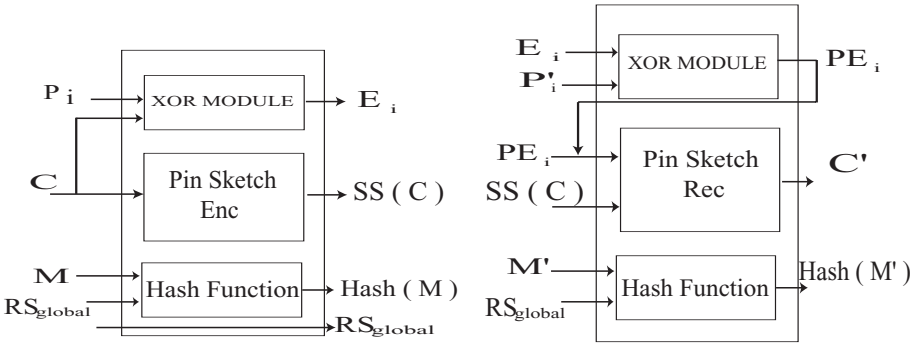
BCH_1 whose codeword length is at least GQ_{max}. P_i and C are XORed to give $PE_i = P_i \oplus C$. C is subjected to the encoding process of a PinSketch module and the result obtained is denoted by SS(C). The PinSketch module consists of the syndrome computation procedure for a BCH code with codeword length GQ_{max} and minimum distance of 21. The choice of minimum distance is due to the fact that the PinSketch recovery procedure must compensate for a a maximum of 10 bit errors between position vectors. Using the generator of BCH_1, the message M_C corresponding to C is computed. Finally, a random seed RS_{global} is selected and is used to generate a Hash of M_C denoted by Hash(M_C), using the pairwise independent hash function module. For each minutia's global position, PE_i, SS(C), Hash(M_C) and RS_{global} form part of the publicly stored template.

FE_{global} **Verification:** The Verification stage also consists of 3 modules - An XOR module, the Recovery module of PinSketch and a pairwise independent hash function module. When a minutia in the query fingerprint is compared with minutia M_i of the enrolled finger, the global Position Vector of the query minutia, P_i', is first compared. To perform the comparison, the public template information corresponding to P_i is retrieved and the following procedure takes place:

1. Compute $P_i' \oplus PE_i$. Let this be denoted as E_i.
2. E_i is passed through the recovery procedure of the PinSketch module. This consists of a BCH decoder which will decode to the nearest valid codeword of BCH_1. If P_i' was within a Hamming distance of 10 bits to P_i, the decoder would correctly decode to the codeword C. This is because $P_i' \oplus PE_i = P_i' \oplus P_i \oplus C$. Let $P_i' \oplus P_i = e_i$. When $|e_i| \leq 10$, the BCH decoder will successfully decode $E_i = C + e_i$ to C. Let the decoded word be denoted by C'.
3. Using the generator of BCH_1, $M_{C'}$, the message corresponding to C' is computed and passed through the pairwise independent hash function module using the same random seed RS_{global} to get Hash($M_{C'}$). If Hash(M_C)= Hash($M_{C'}$), the minutiae are said to match in their global position.

FE_{local} **Enrolment:** For each local Position vector of a neighbor $N_i^j, 1 \leq j \leq 5$ of minutia M_i, a random codeword CC_i^j is picked from a BCH code BCH_2 whose codeword length is at least LQ_{max} and minimum distance is 21. Each pair of N_i^j and CC_i^j are successively passed through the XOR module to get five words $NE_i^j, 1 \leq j \leq 5$. Each CC_i^j is passed through the encoding module of the PinSketch$_1$ to get five syndromes denoted by SS(CC_i^j),$1 \leq j \leq 5$. Using the generator of BCH_2, the five messages corresponding to the five random codewords are computed $mm_i^j, 1 \leq j \leq 5$.

The five messages are considered as a set L=$\{mm_i^j, 1 \leq j \leq 5\}$. L can now be considered as a binary string of length equal to $2^{|mm_i^j|}$, with ones occurring at positions corresponding to the elements in the set. L is passed through the enrolment procedure of PinSketch$_2$ which works over a BCH code BCH_3 of length $2^{|mm_i^j|}$ and minimum distance 5, to get SS(L). The minimum distance of 5 for



(a) FE_global Enrolment Modules (b) FE_global Verification Modules

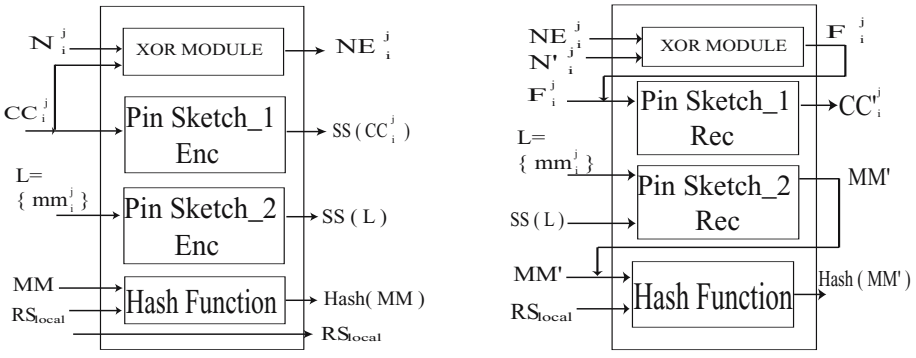
Fig. 3. Block diagram for FE_{global}

BCH_3 is chosen because we seek to correct 2 set difference errors between the set L and the set that will be obtained from the minutia in the query fingerprint(and hence 2 bit errors between their binary characteristic vectors).

The five messages are ordered and concatenated to form a single word MM. A random seed RS_{local} is used to generate the Hash of MM using the pairwise independent hash function. Thus, the pattern of the local Five Nearest Neighbor Structure for minutia M_i is defined by $NE_i^j, 1 \leq j \leq 5$, SS(CC_i^j),$1 \leq j \leq 5$, SS(L),Hash(MM) and RS_{local} in the publicly stored template.

FE_{local} **Verification**: Our criteria for matching two local structures, requires four out of the five neighbor positions to match. This allows for one insertion and one deletion of minutiae between two structures that are compared. There are 5! ways to map the structure of the minutia in the enrolled fingerprint with that of the minutia in the query fingerprint. We call each such mapping as a *Permutation*. For each Permutation, there will be a one to one correspondence between the neighbors in the two structures that are being compared. In a certain Permutation, let $N_i^{j'}$ be the query structure's neighbor corresponding to N_i^j. The following procedure is performed:

1. Compute $N_i^{j'} \oplus NE_i^j, 1 \leq j \leq 5$. Let each be denoted as $F_i^j, 1 \leq j \leq 5$.
2. Each F_i^j is passed through the recovery module of PinSketch$_1$ to correct 10 bit position errors. On decoding, we will have five estimates of the codewords $CC_i^{j'}, 1 \leq j \leq 5$.
3. Using the generator of BCH_2 , we obtain the corresponding message words $mm_i^{j'}, 1 \leq j \leq 5$. They form the set L'.
4. L' is passed through the recovery procedure of PinSketch$_2$, which will correct 2 set difference errors between L and L'. The resulting set of 5 messages are ordered in the same way as during enrolment and concatenated to form MM'.
5. Finally the random seed RS_{local} is used to compute Hash(MM') using the pairwise independent hash function module.If Hash(MM') equals Hash(MM), the



(a) FE_local Enrolment Modules (b) FE_local Verification Modules

Fig. 4. Block Diagram for FE_{local}

local structures are said to match. All possible Permutations are tried until a match is found. If no Permutation yields a match, then the local structures of the corresponding minutiae do not match.

When two fingerprints are compared, the global positions of corresponding minutiae are compared first. Only on obtaining a match in global position are the minutiae tested for match in the shape of the local structure. If sufficient number of minutiae match in both global position and local structure, the two fingerprints are considered as successful match.

4 Experiment Description

Our algorithm was run separately on two fingerprint databases. One was a privately collected database consisting of 74 distinct fingerprint impressions with 5 samples of each, weakly aligned[2]. Weak alignment means that a horizontal line through the core dividing the fingerprint in half, does not vary by more than a specified angle from the scanner's horizontal. For this database, this angle was 20°. The fingerprints were rolled, off-line, inked impressions scanned using a Mitsubishi Diamond View DV648U flatbed scanner set at 600 dpi. The core and minutiae (roughly 30 per fingerprint impression) were identified by eye and their Cartesian coordinates and orientation were recorded. The second database was the public FVC2000 Database 1a comprising of 100 fingerprints and 8 samples of each. We used Neurotechnologija's feature extraction SDK to determine the coordinates and orientation of the minutiae and core.

Quantization Parameters: The quantization parameters for each database had to be separately tuned. For the privately collected database, we chose R_{tol} and d_{tol} to be 40 pixels, Θ_{tol} and θ_{tol} were taken to be 20° and R_{max} and d_{max}

[2] Many fingerprint scanners enforce weak alignment by virtue of their fingerprint capture technique.

were set to 300 pixels. With these parameters, the maximum number of quanta in both the local and global frames was 126 i.e $GQ_{max} = LQ_{max} = 126$. For the public database, we chose R_{tol} and d_{tol} to be 20 pixels and kept Θ_{tol} and θ_{tol} at 20°. This gave $GQ_{max} = LQ_{max} = 270$.

BCH Codes: For the privately collected database, the smallest BCH code length covering 126 quanta was 127. As we needed to correct 10 errors (Section 3), we chose both BCH_1 and BCH_2 to be BCH(127,64,21). BCH_3 was chosen to have a codeword length of 2^{64} and minimum distance of 5. For FVC2000 Database 1a, we chose BCH_1 and BCH_2 to be BCH(511,421,21) and BCH_3 to have a codeword length of 2^{421} and minimum distance of 5.

Hash function: We used a simple pairwise independent hash function described as follows: If P is the Message Space, K is the Key Space and T is the Tag Space, the Hash Function $Hash$ is defined as $Hash : K \times P \rightarrow T$. For our implementation, the message p, tag t and a key pair (k_1,k_2), each in GF(2^m), where m is a positive integer, are related by the function t = Hash(p) = $k_1 \times p + k_2$ [10]. The multiplication occurs over GF(2^m) and p corresponds to m_C or MM in the FE_{global} or FE_{local} respectively. The key pair (k_1,k_2) corresponds to RS_{global} or RS_{local} respectively. For the privately collected database, we worked over GF(2^{64}) and for FVC2000 Database 1a we used GF(2^{421}) as the fields from which p, k_1 and k_2 were drawn.

Software. The software has been written in C++ and run on the Windows platform on a Pentium 4, 3 GHz machine. We used the public domain software written by Harmon and Reyzin [11] for syndrome computation and syndrome decoding of BCH codes. Victor Shoup's NTL library [12] was used for performing Finite Field computations.

5 Results and Analysis

Applying the above parameters to the fingerprints in our database, we evaluated the *False Non Match Rate(FNMR)* and *False Match Rate(FMR)* at different match Thresholds. The graph is shown in Fig. 5. For the privately collected database, we observe that Equal Error occurs at a Threshold of 18 and the error is roughly 10%. However, for FVCDatabase 1a we found the error at Equal Error Point was closer to 15%. We observed that the low Genuine Acceptance Rate and higher error in the FVC2000 database compared to our privately collected database was due to the error introduced by the feature extractor in extracting minutiae and the core location. False non matches particularly occurred when the extractor found double cores in the fingerprint.

Security Analysis: In the privately collected Database, the quantization frame used for global and local minutia positions gave exactly 126 possibilities each for the global and local binary position vectors, which an intruder could easily list. There are 126^4 possibilities for a FNN structure shape, as the intruder needs to get four out of the five neighbor positions correctly. If operating at the equal error

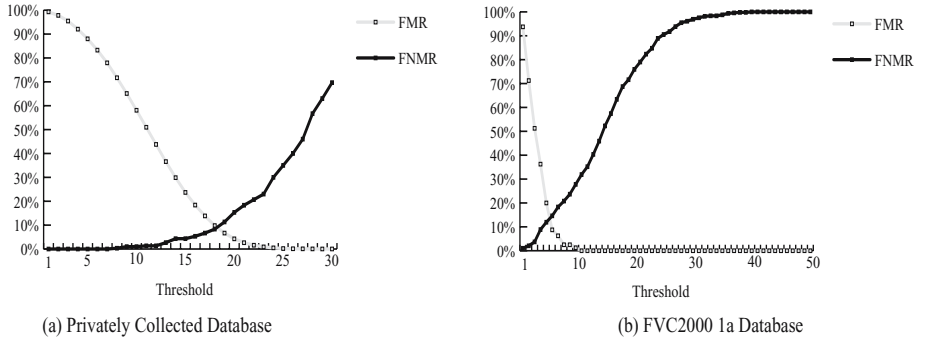


Fig. 5. FMR and FNMR curves

threshold, a minimum of 19 such structures need to be found to have a successful authentication. Thus the entropy of the fingerprint with such a representation scheme is $log_2(126^4 \times 19) = 32$ bits approx. For the FVC2000 database, the stricter quantization resulted in 270 possibilities for the local or global position vectors. Using a similar argument as before, the entropy of a fingerprint in this representation method would be $log_2(270^4 \times 6) = 34$ bits approx. Recovering an approximate reconstruction of the point pattern without the ridge orientations will be insufficient to reconstruct the fingerprint as described in [1] and trace a person's identity. Hence the identity of a user is protected under our scheme. However, an attacker who could hack into the feature extractor module could use this estimated point pattern to masquerade as the genuine enrolled user by overriding the feature extractor. A safeguard against such an attack would be to ensure that the feature extraction and matching algorithms be done at a secure site.

6 Conclusion and Future Work

We have demonstrated a working proof of concept of a secure minutia-based authentication system. The equal error rate is promising enough to encourage further research into faster matching, better quantization parameters and minutiae representation schemes. We will perform a full security analysis on this system and study methods (finer quantization and codes with larger error correcting capability) to improve its security. We will also test the system with alternate minutia structures to obtain better performances at equal error.

Acknowledgements. This research was supported by a grant from the Australian Department of Defence, which includes a scholarship for the first author. We are deeply grateful to Assoc. Prof. Serdar Boztas for useful discussions on pairwise independent hash functions. We also thank the anonymous reviewers for their comments which helped us to significantly improve our paper.

References

1. Ross, A., Shah, J., Jain, A.K.: Towards reconstructing fingerprints from minutiae points. In: Proc. SPIE, Biometric Technology for Human Identification II, vol. 5779, pp. 68–80 (March 2005)
2. Juels, A., Wattenberg, M.: A fuzzy commitment scheme. In: Tsudik, G. (ed.) Sixth ACM Conference on Computer and Communications Security, pp. 28–36. ACM Press, New York (1999)
3. Juels, A., Sudan, M.: A Fuzzy vault scheme. In: Proc. of IEEE ISIT, Lausanne, Switzerland, June 30-July 5, 2002, p. 408. IEEE Press, Los Alamitos (2002)
4. Linnartz, J.-P., Tuyls, P.: New shielding functions to enhance privacy and prevent misuse of biometric templates. In: Kittler, J., Nixon, M.S. (eds.) AVBPA 2003. LNCS, vol. 2688, pp. 393–402. Springer, Heidelberg (2003)
5. Tuyls, P., Goseling, J.: Capacity and Examples of Template-Protecting Biometric Authentication Systems. In: Maltoni, D., Jain, A.K. (eds.) BioAW 2004. LNCS, vol. 3087, pp. 158–170. Springer, Heidelberg (2004)
6. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data, Cryptology ePrint Archive, Report, 2003/235 (2003), Available `http://eprint.iacr.org`
7. Uludag, U., Jain, A.K.: Securing fingerprint template: fuzzy vault with helper data. In: Proc. IEEE Workshop on Privacy Research In Vision, NY, June 22, 2006, p. 163 (2006)
8. Tuyls, P., Akkermans, A.H.M., Kevenaar, T.A.M., Schrijen, G.-J., Bazen, A.M., Veldhuis, R.N.J.: Practical Biometric Authentication with Template Protection. In: Kanade, T., Jain, A., Ratha, N.K. (eds.) AVBPA 2005. LNCS, vol. 3546, pp. 436–446. Springer, Heidelberg (2005)
9. Jeffers, J., Arakala, A.: Minutiae-based Structures for a Fuzzy Vault. In: Proc. of 2006 Biometrics Symposium, MD, USA, September 19-21, 2006 (2006)
10. Shoup, V.: A Computational Introduction to Number Theory and Algebra, p. 125. Cambridge University Press, Cambridge (2005)
11. Harmon, K., Reyzin, L.: Implementation of algorithms from the paper Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data[Accessed October 3, 2006], [Online] Available: `http://www.cs.bu.edu/~reyzin/code/fuzzy.html`
12. Shoup, V.: NTL: A Library for doing Number Theory [Accessed October 9, 2006], [Online] Available: `http://shoup.net/ntl`