Biometric Fuzzy Extractors Made Practical: A Proposal Based on FingerCodes*

Valérie Viet Triem Tong[1], Hervé Sibert[2], Jérémy Lecœur[3], and Marc Girault[4]

[1] Supelec, Campus de Rennes, Avenue de la Boulaie F-35576 Cesson-Sévigné Cedex, France
[2] NXP Semiconductors, 9 rue Maurice Trintignant, F-72081 Le Mans Cedex 9, France
[3] Irisa-Inria, Campus de Beaulieu, F-35042 Rennes Cedex
[4] France Telecom Research and Development, 42 rue des Coutures, BP6243, F-14066 Caen Cedex 4, France

Abstract. Recent techniques based on error-correction enable the derivation of a secret key for the (varying) measured biometric data. Such techniques are opening the way towards broader uses of biometrics for security, beyond identification. In this paper, we propose a method based on fingerprints to associate, and further retrieve, a committed value which can be used as a secret for security applications. Unlike previous work, this method uses a stable and ordered representation of biometric data, which makes it of practical use.

1 Introduction

Biometric authentication refers to verifying physiological or behavioral features such as voiceprint, fingerprint or iris scan [9,2]. From personal device unlock to e-passports, biometry-based authentication schemes have spread into our lives. Such schemes usually run as follows: the user presents his current biometric data, and the verifier checks if the measured data match the biometric reference data acquired during the enrollment.

Biometric data have several properties which make them natural candidates for security applications: they are hard to forge, they are unique to each person, and they are a good source of entropy. However, they also have several drawbacks which slow down their adoption: one cannot change his biometric data, biometric data are often easy to steal, and acquisition of biometric data is subject to variations.

As the measured data can vary, they cannot be directly used as a password or as a cryptographic secret. Moreover, they have to be sent across a possibly insecure network for matching against reference data stored in a database, whose loss would be disastrous. Several research directions address these concerns. First, recent schemes avoid the storage of the whole reference data. Second, new techniques based on error-correction enable the derivation of a constant secret key

* Most of the research work that this article stems from was carried out by the authors at France Telecom Research and Development in Caen (France).

S.-W. Lee and S.Z. Li (Eds.): ICB 2007, LNCS 4642, pp. 604–613, 2007.

from the (varying) measured biometric data. Such techniques can also naturally address the biometric data storage problem.

Still, stolen biometric data are stolen for life. Unlike a cryptographic key, they cannot be updated or destroyed. To solve this last problem, it remains to be able to use biometric data to retrieve a secret value linked to each user, instead of them directly as a secret. Few such proposals have been made, but they are not practical. In this paper, we propose a practical method based on fingerprints to associate, and further retrieve, a committed value which can be used as a secret for security applications.

1.1 Related Works

The fact that storing biometric reference data should be avoided for distant authentication system is commonly accepted, as mentioned by Jain *et al.* in [16]. In an attempt to do so [8], Jain and Uludag propose to use watermarking to secure the transmission of biometric data across a network. However, they still use a biometric database.

In [11], Juels and Wattenberg present the first *crypto-biometric system* called *fuzzy commitment*, in which a cryptographic key is decommitted using biometric data. *Fuzziness* means that a value close to the original (under some suitable metrics) is sufficient to extract the committed value. The scheme is based on the use of error-correcting codes, and runs as follows. Let $C \subset \{0,1\}^n$ a set of codewords for a suitable error-correcting code. The user chooses a secret codeword $c \in C$, the decommitment key is the enrolled fingerprint x and the commitment is the pair $(c - x, \mathcal{H}(c))$, where $\mathcal{H}$ is a one-way function. When a user tries to decommit a pair $(c - x, \mathcal{H}(c))$ using a decommitment key x', he attempts to decode $c - x + x'$ to the closest codeword c'. Decommitment is successful if $\mathcal{H}(c') = \mathcal{H}(c)$, which means the user has retrieved his secret c. In this scheme, the key x is built from a fingerprint, as a set of minutiae positions. This yields two shortcomings. First, it does not allow modifications of x, such as re-ordering, and addition/deletion of an element in x, although such modifications are frequent in real life. Second, the security proof of this scheme holds only if x is uniformly distributed, which is not the case in reality.

In order to overcome these drawbacks, Juels and Sudan propose a *fuzzy vault* scheme [10]. This scheme may be thought of as an order-invariant version of the fuzzy commitment scheme, obtained by using a generalization of Reed-Solomon codes in which they think of a codeword as an evaluation of a polynomial over a set of points. The idea is to encode a secret s as a polynomial p of degree d using the Reed-Solomon encoding scheme. The codeword consists of a set of pairs $R_1 = \{(x, p(x)\}_{1 \leq i \leq n}$, where the x-coordinates represent the position of minutiae in the reference fingerprint. A set R_2 of *chaff* points that are not on p is added to R_1 to form the *vault* V. To rebuild the codeword using the Reed-Solomon decoding scheme, the user needs $d{+}1$ pairs $(x, y) \in R_1$ from the original n points. The method has been successfully tested by Uludag and Jain with the IBM-GTDB database [15]. However, the fuzzy vault has one main drawback, which is its restricted use. Indeed, if the polynomial p is compromised, then the fingerprint itself is, as all the other minutiae in the fingerprint are the points on p in the vault V. Thus, if this method was used for different applications, with different vaults each time, as the x-coordinates correspond to the minutiae, disclosure of different vaults for the same user would reveal his minutiae.

Next, Clancy *et al.* considered a practical implementation of the fuzzy vault in a secure smartcard [4]. Their experimentation showed that a sufficient security level for the fuzzy vault cannot be obtained with real-life parameters. They thus defined a modified scheme called the *fingerprint vault*, and proposed a way to find the optimal vault parameters.

However, the choice of chaff points also yields uniformity problems, that can be exploited by an attacker to distinguish between chaff points and real points in the vault. Chang and Li have analyzed this problem [3] in a general setting. They show that, since secret points are not uniformly distributed, the proper way of choosing chaff point is far from being trivial, and there is no known non-trivial bound on the entropy loss.

1.2 Contents of the Paper

In this paper, we propose a new method to associate, and further retrieve, a committed value using a fingerprint. Our method is based on a special fingerprint data representation measure called FingerCode [14]. The committed value is rebuilt from the FingerCode biometric data and public data we call **FingerKey**. The committed value cannot be recovered from the public data, and we show that no illegitimate user is able to forge it. Moreover, using the FingerCode, we avoid the minutiae set modifications concerns. At last, our method does not require storage of any biometric reference, and therefore, when the committed value is used as a (seed for generation of a) secret key in a secure cryptographic protocol, an attacker cannot learn the biometric data.

The paper is organized as follows: after the introduction, we present the tools we use and, more particularly, error-correcting codes, the FingerCode definition and its extraction method in Section 2. In Section 3, we describe our construction and give our experimental results. In Section 4, we give some advice on possible applications and security parameters. At last, we give directions for future work and we conclude.

2 Preliminaries

2.1 Error-Correcting Codes

Like the *fuzzy* schemes of Juels *et al.*, our scheme relies on using error-correcting codes. We refer the reader to [12,13] for more details on error-correcting codes.

The goal of error-correcting codes is to prevent loss of information during the transmission of a message over a noisy channel by adding redundancy to the original message m to form a message m', which is transmitted instead of m. If some bits are corrupted during the transmission, it remains possible for the receiver to reconstruct m', and therefore m. In our scheme, the noise is the result of the use of different measures of the fingerprint.

More formally, an error-correction code over a message space $M = \{0,1\}^k$ consists of a set of codewords $C \subset \{0,1\}^n$ $(n > k)$ associated with a coding function denoted by `encode`, and a decoding function denoted by `decode`, such that `encode`$: M \rightarrow C$ injects messages into the set of codewords and `decode`$: \{0,1\}^n \rightarrow C \cup \emptyset$ maps an n-bits string to the nearest codeword if the string does not have too many errors, otherwise it outputs $\emptyset$.

We recall that the Hamming distance between binary strings is the number of bit locations where they differ. The *distance* Δ of a code is the minimum Hamming distance between its codewords. Therefore, to be properly decoded, a string must have at most $\frac{\Delta-1}{2}$ errors. In the sequel, we have chosen to use Reed-Solomon codes for their correction power.

2.2 FingerCode

Our purpose is to retrieve constant secret data from a varying measures of biometric data. This measure should have be reasonably stable, which means slight modifications of the acquisition should result in a low distance from the reference. Therefore, we cannot use a minutiae-based matching method.

We propose to use a texture-based method, called *FingerCode*, which is stable in size, and founded on the localization of the morphological centre of the fingertip and the use of a well-known pattern analysis method to characterize a fingerprint. It was introduced by Jain and Prabhakar in [14,6]. A FingerCode is a 640-component vector of numbers that range from 0 to 7, which is ordered and stable in size. The matching is then handled by a simple Euclidean distance. Here is a summary of this method.

Using Bazen and Guerez method [1], an estimation of the block orientation field is computed. Using this orientation, a very simple curvature estimator can be designed for each pixel. The maximum value of this estimator is the morphological center that we are searching. We extract a circular region of interest around this point in which we consider 5 concentric bands and 16 angular portions making a total of 80 sectors. The width of the band is related to the image resolution, e.g. for a 500-dpi image, the band is 20-pixels wide. Although we avoid the problem of translation and rotation of the image by using the morphological center and a circular area, we still have to handle the finger pressure differences. Hence, we normalize the image to given values of mean and variance.

If normalization is performed on the entire image at once, then the intensity variations due to the finger pressure difference remain. Therefore, it is necessary to normalize separately each sector of the image, in order to obtain an image where the average intensity at a local scale is about the same as that of the whole image.

Ridges and valleys can be characterized by their local frequency and orientation. So, by using a properly tuned Gabor filter, we can remove noise and catch this piece of information. An even symmetric Gabor filter has the following general form in the spatial domain:

$$G(x,y,f,\theta) = \exp\left\{-\frac{1}{2}\left[\frac{x'^2}{\delta_x^2} + \frac{y'^2}{\delta_y^2}\right]\right\} \times \cos(2\pi \times f \times x')$$
$$x' = x\sin\theta + y\cos\theta$$
$$y' = x\cos\theta - y\sin\theta$$

where f is the frequency of the sinusoidal plane along direction t from the x-axis, and δ_x and δ_y are the space constants of the Gaussian envelope along the corresponding axis. The filtering in the spatial domain is performed with a 19×19 mask in 8 directions from 0 to $157,5\,^\circ$, resulting in 8 filtered images of the region of interest.

Finally, we compute the FingerCode as the average absolute deviation from the mean (AAD) of each sector of each image. The feature vector is organized as follows: values from 0 to 79 correspond to the $0\,^\circ$ filter, the 80 following values to the $22,5\,^\circ$ filter and so on. In these 80 values, the 16 first are the innermost band values, the first one being the top sector and the other values corresponding to the sectors counted clockwise. The matching is based on Euclidean distance between FingerCodes. We also use cyclic rotation of the FingerCode, up to 2 steps, to handle rotations up to $45\,^\circ$.

3 Our Construction

The idea behind our proposal is to use the FingerCode, which offers ordered and stable biometric data, in the fuzzy commitment scheme [11], in order to avoid minutiae-related drawbacks. However, the FingerCode has a too high FRR rate to be used directly as the decommitment key: error-correction would never be efficient enough to recognize a legitimate user. Therefore, our scheme slightly differs from the fuzzy commitment. It consists of a fuzzy extractor and of a secure sketch according to the definitions of Dodis *et al.* [5]. In our construction, the secret is represented as a word of $d+1$ letters, which correspond to the $d+1$ integer coefficients of a polynomial $p \in \mathbb{Z}[X]$ of degree d.

The registration step consists in extracting public data called *FingerKey* from the pair (F, p), where F is a FingerCode. To this end, n points of p are randomly chosen, with $n > d$, and we hide these points using n stable subparts of the FingerCode like in the fuzzy commitment scheme. To retrieve the secret polynomial p, we use the usual decoding procedure for each point. If at least $d+1$ points can be decommitted, then p is obtained by Lagrange interpolation.

3.1 Encoding

The encoding part, or *fuzzy extractor* as defined in [5], is a set of fuzzy commitments where the committed values are points of on a secret polynomial. First, the system chooses a secret polynomial p of degree d (for instance, using a bijection between a secret key chosen by user U and the polynomials space), and n random points $p_0 \dots p_{n-1}$ on this polynomial ($n > d$). These points are represented by $l - bit$ strings, and encoded with an encoding function $\mathcal{RS}$-$\texttt{encode}$ which outputs n codewords $c_0, \dots, c_{n-1} \in \{0,1\}^{l+e}$. In our experimentation,

$\mathcal{RS}$-`encode` is the encoding function for Reed-Solomon codes. The FingerCode F_U of the user is then divided into n parts $F_U = f_0 || \ldots || f_{n-1}$. The system then computes $\{\delta_i = c_i - f_i, 0 \leq i \leq n-1\}$. The FingerKey K_{F_U} for user U consists of the set of pairs $(\delta_i, \mathcal{H}(c_i))$ where $\mathcal{H}$ is a one-way function and $0 \leq i \leq n-1$. This algorithm is described in Figure 1. The FingerKey is then made public for future reference, and may be stored in a server or in a user's smartcard.

```
Inputs      polynomial $p$ of degree $d$
            integer $n$ with $n > d$
            family $(p_i)_{0 \leq i \leq n}$ of randomly chosen points of $p$
            FingerCode $F_U = f_0 || \ldots || f_{n-1}$

Outputs     FingerKey $K_{F_U}$

Algorithm   $K_{F_U} \leftarrow \emptyset$
            For $i$ going from 0 to $n-1$ do
              $c_i \leftarrow \mathcal{RS}\texttt{-encode}(x_i, p(xi))$
              $\delta_i \leftarrow c_i - f_i$
              $K_{F_U} \leftarrow K_{F_U} || (\delta_i, \mathcal{H}(c_i))$
            Return $K_{F_U}$
```

Fig. 1. Encoding algorithm: FingerKey extractor for user U

3.2 Decoding

In order to retrieve his secret, user U has his FingerCode $F'_U = f'_0 || \ldots || f'_n$ measured by the system. The system looks up the FingerKey K_{F_U} for U, and uses the $\mathcal{RS}$-`decoding` function to compute, for each value $\delta_i + f'_i$, the closest codeword c'_i. In our experimentation, as we use the Reed Solomon code for encoding, we use the Reed Solomon decoding function $\mathcal{RS}$-`decoding`. If equality $\mathcal{H}(c_i) = \mathcal{H}(c'_i)$ holds, then, the decommitment of the codeword is successful. If at least $d+1$ values f'_i $(i \leq 0 \leq n)$ yield a successful decommitment, the user is able to rebuild p using Lagrange interpolation. The user is thus authenticated as U if he succeeds in decommitting at least $d+1$ points and thus rebuild the secret polynomial p. The system can then rebuild the user-chosen secret. The algorithm is given in detail in Figure 2.

3.3 Experimentation

We have experimented this method on a fingerprint database containing 1000 pictures. In a first attempt, we used FingerCode values rounded to the nearest integer. It turned out that the FingerCode differed too much from the original value to be corrected by error-correction. We then changed our rounding method as follows: values from 0 to 2 were replaced by 0, those from 2 to 4 were replaced by 1, and the others (from 4 to 7) were replaced by 2. Thus, we used transformed FingerCodes whose vector components values were 0, 1 or 2, which yields 3^{640}

```
Inputs     degree $d$ of the committed polynomial
           integer $n$ with $n > d$
           User FingerCode $F'_U = f'_0 || \dots || f'_{n-1}$
           FingerKey for user $U$ $K_{F'_U} = (\delta_0, \mathcal{H}(c_0)) || \dots || (\delta_{n-1}, \mathcal{H}(c_{n-1}))$

Outputs    polynomial $p$ or Failure

Algorithm  $P \leftarrow \emptyset$
           For $i$ going from 0 to $n-1$ do
             $c'_i \leftarrow \mathcal{RS}\texttt{-decode}(\delta_i + f'_i)$
             If $\mathcal{H}(c_i) = \mathcal{H}(c'_i)$
               $P \leftarrow P \cup \{c_i\}$
           If $|P| > d$,
             $p \leftarrow$ Lagrange Interpolation of elements in $P$
             Return $p$
           otherwise return Failure.
```

Fig. 2. Decoding algorithm: Secure sketch for a user pretending to be U

possible values for an entire FingerCode. We divided each FingerCode into sixteen parts, used the Reed-Solomon code $RS(2^5, 10)$ defined by his generator polynomial $X^6 + X^5 + 1$, and we chose secrets as being polynomials of degree 8. Then, no FingerCode coming from a fingerprint of another user could lead to a successful decommitment. For a FingerCode coming from another fingerprint of the legitimate user, the decommitment success rate was more than 60%.

As our system is more flexible than the standard procedure and as it allows the regeneration of a user-chosen secret, with public data(FingerKey), we may have expected worse results than standard matching. However, using the standard FingerCode matching on the same picture database, we obtained a FRR of 78% for a FAR of $0,1\%$. Therefore, it appears that splitting the FingerCode into several parts and using our construction allows for better results than the standard FingerCode matching. In the literature, better results are given for usual FingerCode matching: Prabhakar obtains a 19,32% FRR for a 0,1% FAR. Therefore, it is likely that our results can also be improved, and, using FingerCode enhancements (picture optimization...), we can reasonably expect results for our construction similar to those obtained for standard FingerCode matching.

4 Applications and Security

4.1 Security

The scheme we have presented enables users to retrieve a committed secret, without biometric data storage. Just like other biometric systems, our system should be used in a safe manner. Indeed, measuring biometric data and then using these data to impersonate the legitimate user is still possible. Retrieving a user's secret is always possible given his FingerKey and a measure of his

FingerCode. Nevertheless, the concealment property ensures that retrieving the secret is unfeasible given the FingerKey only.

More formally, the security of the scheme relies on three properties:

1. an attacker who knows only a published **FingerKey** cannot retrieve the biometric data which it was computed from,
2. *(concealment)* an attacker who knows only a published **FingerKey** cannot retrieve the corresponding secret polynomial p,
3. an attacker who knows both a published **FingerKey** and the polynomial p cannot retrieve the corresponding biometric data.

First, let us notice that retrieving the secret polynomial p from the biometric data (FingerCode) associated with a **FingerKey** is straightforward. Property 1 arises from the security of the *fuzzy commitment* scheme [11]. We will show how it extends to Property 3. In the following Theorem, we show that the computational complexity of retrieving p is at least equal to the computational complexity of the inversion of the hash function used. Therefore, Property 2 holds as long as inverting the hash function is not computationally feasible.

Theorem 1. *Suppose that an attacker knows the value K_F of a* **FingerKey** *, and that he has access neither to the biometric data of the user corresponding to K_F, nor to a view of an execution of the scheme.*

Consider the following parameters of our system: ℓ is the length of the FingerCode subcomponents f_i, n is the number of FingerCode subcomponents, Δ is the distance of the error-correcting code used, L is the output length of the one-way function $\mathcal{H}$, d is the degree of the secret polynomial p, and $\mathcal{C}(\mathcal{H})$ the complexity of inverting $\mathcal{H}$ for a random input.

Given the FingerKey K_F, the complexity of retrieving p is equal to

$$\min(2^L, (d+1) * \mathcal{C}(\mathcal{H}), (d+1) * \frac{2^\ell}{\binom{\ell}{\frac{\Delta-1}{2}}}).$$

Sketch of the proof. We suppose p is chosen in a set whose cardinality is beyond exhaustive search. In order to retrieve p, the attacker has to find $d+1$ points on p. As he knows only the FingerKey K_F, this means he has to find $d+1$ elements in $\{c_i \in C\}_{1 \leq i \leq n}$. As p is independent from the FingerCode, the values δ_i do not reveal any information about the c_i's. Therefore, the attacker has either to revert $\mathcal{H}$, or to find at least $d+1$ values f such that $\mathcal{H}(RS - decode(\delta_i + f)) = \mathcal{H}(c_i)$ for different i's.

In order to find some f such that the distance between c_i and $\delta_i + f$ is at most $\frac{\Delta-1}{2}$, the attacker has to try an average of $\frac{2^\ell}{\binom{\ell}{\frac{\Delta-1}{2}}}$ values, and to test each value by computing $\mathcal{H}(RS - decode(\delta_i + f))$ for a given i. Trying several i's simultaneously would just multiply the complexity by the number of i's targeted, thus it does not reduce the global complexity of the attack, which is thus $(d+1) * \frac{2^\ell}{\binom{\ell}{\frac{\Delta-1}{2}}}$.

If the attacker chooses to revert $\mathcal{H}$, there should be no algorithm better than brute force for a proper choice of $\mathcal{H}$, so the complexity of inversion of $\mathcal{H}$ would be 2^L. In the case when there exists an algorithm better than exhaustive search to invert $\mathcal{H}$, the complexity of the attack is at most $(d+1) * \mathcal{C}(\mathcal{H})$. □

The idea behind the proof is that, as all the values $c_i - f_i$ are given, and that knowing the c_i's yields p, the gap between Properties 1 and 2 lies in the given values $\mathcal{H}(c_i)$.

At last, Property 3 holds thanks to the fact that the choice of the points on the polynomial is random. Hence, providing p does not give information on the c_i's, the knowledge of which is equivalent to knowing the biometric data.

Any implementation of the system should ensure that the FingerCode is not stored, and the user should only present his fingerprint to a trusted device. If this is not the case, then, the system could still be secure if the FingerKeys of the user are stored in a user-owned device. For instance, a FingerKey may be stored in a smartcard, which would, on input of the FingerCode value, regenerate the user's secret and perform cryptographic operations using this secret internally.

4.2 Applications

Our system addresses two usual drawbacks of biometrics: it enables the regeneration of constant data, and it also allows to change these data in case of loss or theft.

This system enables applications beyond the reach of usual biometric identification. For instance, the regenerated value can be used as the seed for obtaining a constant RSA keypair (or any other cryptographic key), used for applications such as digital signature, or encryption/decryption. Moreover, a user can choose different secrets for different applications, and also change every secret that would be compromised. Therefore, this system also offers scalability, and decreases the risks linked to the theft of private data.

5 Conclusion

We have presented a system in which no biometric data has to be stored, and the current biometric data is used to decommit a secret value. This enables the regeneration of several secret values used for various applications for each user. By reducing the need for transmission of biometric data, our system also reduces the risks of biometric data theft. This proposal is the first to combine error-correcting codes with stable and ordered biometric templates. Our experiments, based on the FingerCode, provide encouraging results. We now wish to improve our results by finding more suitable biometric measures. Hybrid methods that mix minutia and texture features, as introduced in as in [7], provide more stable biometric measures than the one we have used. One promising research direction is thus to adapt such methods to our construction, in order to improve our experimental results.

References

1. Bazen, A.M., Guerez, H.: Directional field computation for fingerprints based on the principal component analysis of local gradients. In: Proc. of 11th annual Workshop on Circuits, Systems and Signal Processing, 2000 (2000)
2. Bolle, R., Pankanti, S.: Biometrics, Personal Identification in Networked Society: Personal Identification in Networked Society. Kluwer Academic Publishers, Norwell, MA, USA (1998)
3. Chang, E.-C., Li, Q.: Hiding secret points amidst chaff. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 59–72. Springer, Heidelberg (2006)
4. Clancy, T.C., Kiyavash, N., Lin, D.J.: Secure smartcardbased fingerprint authentication. In: WBMA '03: Proceedings of the 2003 ACM SIGMM workshop on Biometrics methods and applications, pp. 45–52. ACM Press, New York (2003)
5. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 523–540. Springer, Heidelberg (2004)
6. Jain, A., Prabhakar, S., Hong, L., Pankanti, S.: Filterbank-based fingerprint matching. In: Proc. of IEEE Transactions on Image Processing, vol. 9(5), pp. 846–859. IEEE Computer Society Press, Los Alamitos (2000)
7. Jain, A., Ross, A., Prabhakar, S.: Fingerprint matching using minutiae and texture features. In: Proc. of International Conference on Image Processing (ICIP), Thessaloniki, Greece, pp. 282–285 (October 2001)
8. Jain, A., Uludag, U.: Hiding biometric data. IEEE Transactions on Pattern Analysis and Machine Intelligence (2003)
9. Jain, A.K., Maltoni, D.: Handbook of Fingerprint Recognition. Springer, New York (2003)
10. Juels, A., Sudan, M.: A fuzzy vault scheme. In: Proceedings of IEEE International Symposium on Information Theory, 2002. IEEE Computer Society Press, Los Alamitos (2002)
11. Juels, A., Wattenberg, M.: A fuzzy commitment scheme. In: CCS '99: Proceedings of the 6th ACM conference on Computer and communications security, pp. 28–36. ACM Press, New York (1999)
12. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes, North-Holland (1977)
13. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes, Part II, North-Holland (1977)
14. Prabhakar, S.: Fingerprint Classification and Matching Using a Filterbank. PhD thesis
15. Uludag, U., Jain, A.K.: Fuzzy fingerprint vault. In: Proc. Workshop: Biometrics: Challenges Arising from Theory to Practice, pp. 13–16 (2004)
16. Uludag, U., Pankanti, S., Prabhakar, S., Jain, A.: Biometric cryptosystems: Issues and challenges (2004)