Performance of the Fuzzy Vault for Multiple Fingerprints (Short Version)*

Johannes Merkle, Matthias Niesing, and Michael Schwaiger

secunet Security Networks AG,
D-45128 Essen,
Germany

Heinrich Ihmor and Ulrike Korte

Bundesamt für Sicherheit in der Informationstechnik (BSI),
D-53175 Bonn,
Germany

Abstract: The fuzzy vault is an error tolerant authentication method that ensures the privacy of the stored reference data. Several publications have proposed the application of the fuzzy vault to fingerprints, but the results of subsequent analyses indicate that a single finger does not contain sufficient information for a secure implementation.

In this contribution, we present an implementation of a fuzzy vault based on minutiae information in several fingerprints aiming at a security level comparable to current cryptographic applications. We analyze and empirically evaluate the security, efficiency, and robustness of the construction and several optimizations. The results allow an assessment of the capacity of the scheme and an appropriate selection of parameters.

1 Introduction

Biometric authentication requires the storage of reference data for identity verification, either centrally (e.g., in a database) or locally (e.g., on the users token). However, the storage of biometric reference data poses considerable information security risks to the biometric application and concerns regarding data protection. As a potential solution to this dilemma, *biometric template protection* systems [BBGK08] use reference data which reveal only very limited information on the biometric trait. Another term frequently used for these schemes is *biometric encryption*. One of the most prominent approaches is the fuzzy vault scheme [JS02],

*This contribution represents a shortened version. Additional investigations, pseude codes and more detailed results are presented in the full paper [MIK+10b].

where the sensitive information (the biometric reference data) is hidden among random *chaff points*.

In [CKL03, UPJ05, UJ06, NJP07], the application of the fuzzy vault scheme to minutiae (*fuzzy fingerprint vault*) has been proposed. However, a subsequent analysis in [MMT09] has revealed that the parameters suggested do not provide security beyond 50 bit cryptographic keys. One of the suggestions in [MMT09] was to enhance security by using multiple fingers. This idea is supported by the observation in [Pla09] that a single fingerprint cannot provide enough entropy to implement a secure biometric template protection.

In this paper we present an implementation of a fuzzy vault based on the minutiae data of several fingerprints. We investigate the security and robustness of the scheme and of several optimizations applied, some of which have already been proposed in the previous constructions [CKL03, UPJ05, UJ06, NJP07]. In particular, we evaluate the impact of the basic parameters and optimizations to error rates, efficiency and security, and we derive suggestions for parameter selection.

This article is structured as follows: In Section 2, we specify the fuzzy multi-fingerprint vault and its optimizations and justify our design decisions. Section 3 assesses the security of the fuzzy fingerprint vault. In Section 4, we report the results obtained in evaluation with real fingerprints. Finally, in Section 5, we draw conclusions and identify open issues for future investigations.

2 Design of the scheme

2.1 Underlying biometric template protection scheme

In the fuzzy vault scheme [JS02], a polynomial is used to redundantly encode a set of (pairwise distinct) private attributes $m_1, \ldots, m_t$ (e.g., biometric feature data) using a variant of Reed-Solomon decoding. First, a random (secret) polynomial $P(z)$ over a finite field $\mathbf{F}_q$ with degree smaller than k is chosen. Then, each attribute m_i is encoded as element x_i of the finite field, i.e. $x_i = E(m_i)$, where E is an arbitrary injective map from the space of attributes to $\mathbf{F}_q$. Each of these elements x_i is evaluated over the polynomial, resulting in a list of (pairwise distinct) pairs $(x_i, y_i) \in \mathbf{F}_q^2$ with $y_i = P(x_i)$. In order to hide the private attributes, $r - t$ *chaff points* $x_{t+1}, \ldots, x_r \in \mathbf{F}_q$ are randomly selected so that $x_i \neq x_j$ for all $1 \leqslant i < j \leqslant r$. For each chaff point x_i, a random $y_i \in \mathbf{F}_q$ with $y_i \neq P(x_i)$ is chosen. The list of all pairs $(x_1, y_1), \ldots, (x_r, y_r)$, sorted in a predetermined order to conceal which points are genuine and which are the chaff points, is stored as the *vault*.

For authentication and recovery of the secret polynomial, another set of attributes (the query set) has to be presented. This set is compared with the stored fuzzy vault $(x_1, y_1), \ldots, (x_r, y_r)$, and those pairs (x_i, y_i) are selected for which x_i corresponds to an attribute in the query set. The selected points are then used to try to recover the secret polynomial using Reed-Solomon decoding.

If the number of genuine points among the identified correspondences (*correct matches*) is at least k, the secret polynomial can be recovered. However, if the set of correspondences also comprises chaff points (*false matches*), the number of correct matches must be greater than k, or the decoding must operate on subsets of the matches resulting in many trials. Details are given in Section 2.6.

In the original fuzzy vault scheme, correspondence between points in the query set and the fuzzy vault means equality (of the encodings in the finite field). However, for the application of the fuzzy vault to fingerprints the definition of this correspondence is usually adjusted to allow a compensation of noise in the measurement of the minutiae. Following the approach of [NJP07] and [UJ06], we define correspondence as mappings determined by a minutiae matching algorithm (see Section 2.3).

The fuzzy vault scheme is error tolerant with respect to the set difference metric, which covers exactly the errors introduced to (naively encoded) minutiae information by insertions, omissions, and permutation of minutiae. The deployment of a minutiae matching algorithm for identifying correspondences between the query set and the fuzzy vault adds robustness with respect to global rotations and translations or non-linear deformations of the fingerprint. Since the matching algorithms included in standard fingerprint software only output a match score and not the list of corresponding minutiae, we use our own matching algorithm (see Section 2.3).

2.2 Multi-biometric fusion

In order to obtain sufficient information for a secure scheme, we use the imprints of $f \geqslant 2$ fingers of each person. We implemented feature level fusion by encoding the minutiae of all fingers in one feature vector. In this vector each minutiae is encoded as a triplet (ℓ, a, b), where $\ell \in \{1, \ldots, f\}$ is an index of the finger on which the minutiae was detected, while a and b denote the Cartesian coordinates of the minutiae location in the fingerprint image. Chaff points are encoded analogously.

A justification for using feature level fusion is given in the full paper [MIK+10b].

2.3 Minutiae matching algorithm

We need to identify matching minutiae between fingerprints for enrollment and for verification: during enrollment minutiae matching is used to identify the most reliable minutiae from several measurements. During verification we have to identify a sufficiently large set of genuine minutiae within the vault to recover the secret polynomial.

The matching is performed for each finger separately by a simple matching algo-

rithm that identifies minutiae correspondences between two sets A and B of points (a, b) (minutiae or chaff points) in the fingerprint image. (We do not use minutiae orientation due to the intrinsic correspondences that render security assessment much more difficult.) The algorithm tries to maximize the number of correspondent points between the sets by finding a suitable global rotation and translation transformation T and tolerates (Euclidean) distances $||\cdot||_2$ between two points smaller than δ, where δ is a parameter of the algorithm.

A pseudo code description and more details are given in the full paper [MIK+10b].

The tolerance parameter δ varies: we use a greater value $\delta = \delta_e$ for enrollment than the value $\delta = \delta_V$ for verification to increase the number of reliable minutiae.

2.4 Optimizations

In this section we introduce several optimizations to the scheme. The impact of these optimizations are evaluated in Section 4.

2.4.1 Restriction of fingerprint area

If the sensor area is sufficiently large, minutiae rarely occur in the corners of the image. In order to ensure that the distribution of the randomly selected chaff points resembles that of genuine minutiae, we restrict both the chaff points and the minutiae considered for the vault to an area $\mathcal{M}$ with sufficiently high minutiae occurrence. A statistical evaluation of the minutiae positions of 82800 fingerprints having 500 DPI revealed that 7/8 of all minutiae occurred in an area defined by a centered ellipse that covers approximately 87000 pixels, which roughly corresponds to 2.25 cm^2. A figure visualizing the determined distribution of minutiae locations and the ellipse can be found in [MIK+10a]. Consequently, for the vault we choose minutiae and chaff points only from the set $\mathcal{M}$ composed of the union of these ellipses on the considered fingers.

2.4.2 Reliability filtering during enrollment

In order to minimize minutiae insertion and omission errors, we use only the most reliable minutiae for the feature vector. For this reason, we use multiple measurements during enrollment and consider only those minutiae in the feature vector that have been detected in all measurements. Details are given in Section 2.5.

2.4.3 Enforcing minimum distance

Due to the deviations in the measured minutiae locations, it can happen during authentication that a minutia in the query fingerprint is closer to a chaff point than to the corresponding minutiae in the vault. The frequency of such assignment errors

can drastically increase if the chaff points are selected too close to genuine minutiae. Therefore, we select the chaff points with a minimum distance d to the genuine minutiae (from the same finger) with respect to the Euclidean distance. Furthermore, in order to prevent that an adversary can exploit this minimum distance to distinguish chaff points from genuine minutiae, we also enforce the minimum distance among the minutiae and chaff points. In particular, if the Euclidean distance between two minutiae of the same finger is smaller than d, one of them is randomly disregarded, and chaff points are selected with minimum Euclidean distance d to all other chaff points and minutiae from this finger.

2.4.4 Quality filtering during authentication

On average, the number of *false matches* of minutiae with chaff points increases with the average number of surplus minutiae (i.e., minutiae not matching with real minutiae in the reference template) per query fingerprint. However, an increase of false matches requires stronger error correction by lowering the degree k of the secret polynomial, which decreases the security of the scheme.

In order to limit the average number s of surplus minutiae per query fingerprint, we filter the minutiae from the query fingerprint using the quality index value output by the minutiae extraction algorithm. Precisely, we define a minimum quality value Q and provide to the matching algorithm only those minutiae of the query fingerprint that have a quality value of at least Q. In our concrete implementation we used the MINDTCT algorithm of NIST [WGT+07] for minutiae extraction which outputs minutiae quality values in the range between 0 and 1.

2.4.5 Enforcement of minimum number of minutiae per finger

One of the main sources for failures during authentication is the difficulty to correctly align the query fingerprints with respect to the stored minutiae. This task is performed by the minutiae matching algorithm (described in Section 2.3) for each finger by identifying the isometry (rotation and translation) that maximizes the number of matches between the minutiae extracted from the query fingerprint and the points (representing minutiae or chaff point) stored in the reference data. However, this approach can only be successful if the reference template contains a sufficient number of minutiae of each finger; otherwise, i.e., if for one of the fingers the reference template contains only few minutiae, the number of wrong matches (with chaff points) resulting by chance from an incorrect isometry may be higher than the number of matches for the correct isometry. In practice, such cases can easily occur if one of the fingerprints captured during enrollment is of relatively poor quality.

For this reason, we require that the reference template computed during authentication contains at least χ minutiae from each finger, where χ is an additional parameter. Since this constraint reduces the number of possible reference templates its impact on security must be analyzed. We provide an estimation of this

reduction in Section 3.2.

2.5 Enrollment

Let $f \geqslant 2$ be the number of fingers used per person, q a prime power, $k < t < r \leqslant q$, and $\chi \leqslant t/f$.

For each user, a random polynomial P of degree less than k over the finite field $\mathbf{F}_q$ is selected. The coefficients of this polynomial represent the secret of the scheme. Then, for each finger u imprints are taken and the minutiae correspondences between these instances are identified using the matching algorithm with tolerance parameter $\delta = \delta_e$. Minutiae outside the considered set $\mathcal{M}$, i.e., with position outside the ellipse $\mathcal{E}$ on the respective finger, are neglected (see Section 2.4.1). Then, t of those minutiae that have been detected in all u imprints of the respective finger are selected at random so that at least χ minutiae are taken from each finger and each pair of chosen minutiae from the same finger has a minimum distance of d. This set T of t reliable minutiae can be considered the biometric template to be protected by the fuzzy vault scheme. The template T is amended with random chaff points, resulting in a set R of r points containing t genuine minutiae and $r-t$ chaff points, so that each point in R has a minimum distance of d to all other points on that finger. Furthermore, in order to ensure that minutiae and chaff points within the vault are not distinguishable by their order, they are lexicographically ordered.

In contrast to the original fuzzy vault scheme [JS02], the secret polynomial is redundantly encoded not by evaluating it on the biometric data itself but only on the minutiae's indexes in the ordered list. Precisely, for all $1 \leqslant i \leqslant r$ we (re-)define $x_i = E(i)$, where E is an injective embedding from the set $\{1, \ldots, r\} \subset \mathbf{Z}$ to $\mathbf{F}_q$. Further, we set $y_i = P(x_j)$, if $\mathbf{m}_i$ is a genuine minutia, and choose a random value $y_i \neq P(x_j)$, if $\mathbf{m}_i$ is a chaff point, where j is the index of $\mathbf{m}_i$ after applying the lexicographic order. This optimization allows a reduction of the field size to the range of r. The *vault* Y is given by the ordered list of minutiae and chaff points, paired with the corresponding y_j values. The vault and a hash value H of the polynomial's coefficients are stored in the database.

A pseudo code description of the enrollment is given in the full paper [MIK+10b].

2.6 Recovery of the polynomial

The unlocking of the vault (during authentication) requires the recovery of the secret polynomial P from a set of points (x_{j_i}, y_{j_i}), some of which (those resulting from *correct matches* with minutiae) lie on the polynomial, while others (resulting from *false matches* with chaff points) do not. For this task, a Reed-Solomon decoder RSDecode is used that receives as input a set of w points $(x_{j_1}, y_{j_1}), \ldots, (x_{j_w}, y_{j_w}) \in \mathbf{F}_q^2$ with $w \geqslant k$ and outputs $e_0, \ldots, e_{k-1} \in \{0, \ldots, q-1\}$,

so that $y_{j_i} = P(x_{j_i})$ holds for at least k of the (x_{j_i}, y_{j_i}) with $P(z) = \sum_{i=0}^{k-1} e_i z^i$, if such a polynomial exists. We assume that the Peterson-Berlekamp-Massey-decoder is used as suggested in [JS02]. This technique is successful, if at least $(w + k)/2$ of the w points handed over to the decoder are correct.

Evaluation reported in [MIK+10a] revealed that setting $w = 2m_c - k$ and $k \approx m_c - m_f$ can provide a good balance between efficient decoding and security, where m_c and m_f are the expected numbers of correct and false matches, respectively. However, if the match rate disperses considerably, it may by necessary to slightly deviate from this value, in order to reduce the False Rejection Rate. As we will see in Section 4.2, this is the case.

2.7 Authentication

We only implement an authentication in the verification scenario, where the identity of the (alleged) user is known a priori.

In order to verify the identity of a user, a query fingerprint is taken for each considered finger. The minutiae are extracted and matched with the minutiae and chaff points contained in the vault stored for the alleged user. (Thereby, the tolerance parameter δ_v used for the minutiae matching algorithm can differ from that used during enrollment.) The indices of those minutiae and chaff points in the vault matching with minutiae in the query fingerprint are identified; the encoded indices $x_i = E(i)$ along with the corresponding y_i values are given to RSDecode (see Section 2.6) to recover the secret polynomial P. If the number of genuine minutiae among these points is sufficiently high (see Section 2.6 for a discussion), the polynomial can be recovered. Finally, the correctness of the recovered polynomial is verified using the hash value stored in the database. Optionally, the coefficients of the recovered polynomial can be used for further cryptographic applications, e.g., as seed in a key derivation function.

A pseudo code description of the verification is given in the full paper [MIK+10b].

3 Security analysis

In this contribution we consider the security of the fuzzy vault for multiple fingerprints with respect to attacks that try to recover the minutiae or, equivalently, the secret polynomial from the vault. It is understood that there are other types of attacks against biometric template protection schemes to which the fuzzy vault is susceptible [SB07]. In particular, the cross matching of the vaults from several independent enrollments of a user represents a serious threat to the fuzzy vault. However, a comprehensive analysis of all potential attacks against the fuzzy vault would go beyond the scope of this paper.

3.1 Polynomial reconstruction attack

The most efficient method to recover the minutiae or the secret polynomial from the vault was published by Mihailescu [MMT09]. This brute force attack is designed to break the implementations of [CKL03] and [UJ06]; in the context of our scheme it is even slightly more efficient as the correctness of the recovered polynomial can be verified using the hash value of the secret coefficients and does not require additional evaluations of the polynomial. With this adaptation the attack systematically searches through all subsets $\{i_1, \ldots, i_k\}$ of $\{1, \ldots, r\}$, computes the unique polynomial P satisfying $P(E(i_j)) = y_{i_j}$ by Lagrange interpolation, and checks the correctness of this polynomial with the stored hash value. According to [MMT09], the number of trials needed is $1.1(r/t)^k$ and each trial requires $6.5k \log^2(k)$ arithmetic operations over $\mathbf{F}_q$. However, in the latter estimation an explicit constant of 18 for multiplication of the polynomials (see Corollary 8.19 in [GG03]) has been overlooked, and thus, we end up with a total number of approximately $129k \log^2(k)(r/t)^k$ arithmetic operations.

If the number of chaff points is close to the maximum possible, the attack described in [CST06] can be more efficient than brute force. The basic idea is that the free area around chaff points is smaller than around genuine minutia. Assuming a density 0.45 for random sphere packings [CKL03], the maximum number of chaff points per finger would be $0.45 \cdot 87000/V_d$, where V_d is the number of integer point in the sphere of radius d.

3.2 Entropy loss by the minimum number of minutiae per finger

Whereas the enforcement of a minimum number χ of minutiae per finger (see Section 2.4.5) aims at reducing the false rejection rate it also decreases the security of the scheme by narrowing the set of possible templates. This applies to the lower bound on attacks according to [MIK+10a] as well as to the practical attack of [MMT09]. The subsequent analysis quantifies this reduction of security.

We will assume that the minutiae chosen are independently and uniformly distributed among the F fingers. This assumption can be fulfilled by a suitable probabilistic selection method of the template T from the set of reliable minutiae during enrollment.

Using this assumption and the inclusion-exclusion-principle, we can estimate the probability $\zeta(t, \chi)$ that a template with t minutiae includes for each finger f at least χ minutiae by

$$\zeta(t,\chi) = 1 - f^{-t} \sum_{\ell=1}^{f} (-1)^{\ell} \binom{f}{\ell} \cdot \sum_{i_1,\ldots,i_\ell=0}^{\chi} \binom{t}{i_1, \ldots i_\ell, t - \sum_j i_j} (f-\ell)^{t-\sum_j i_j},$$

where $\binom{a}{b_1,\ldots,b_m}$ with $b_1 + \cdots + b_m = a$ denotes the multinomial coefficient.

On the other hand, the conditional probability p that a particular instance of a template T is chosen, if a minimum number of χ minutiae per finger is enforced, can be calculated from the probability p' that this instance is chosen, if no minimum number of minutiae per finger is enforced, by the equation $p = p'/\zeta(t, \chi)$. Therefore, the search space for an attack is narrowed by the factor $\zeta(t, \chi)$, and consequently, the best known attack could be adapted to require at most $129\zeta(t, \chi)k \log^2(k)(r/t)^k$ operations.

4 Results

In this section, we summarize the results of empirical parameter evaluations, the impact of our optimizations and the general performance of the scheme.

We used a test set of 864 fingerprints taken from 18 persons in the course of this research using an optical sensor, each person providing 6 imprints of 8 fingers (little fingers were excluded). In our experiments, we used 6 or all 8 fingers per person (without or with thumbs), but results referring to single fingers were averaged over all finger types.

For minutiae extraction, we used the MINDTCT algorithm of NIST [WGT+07]. We stress that other feature extraction algorithms may exhibit a different performance, and therefore, the resulting statistics may deviate from ours.

4.1 Size of feature vector

First, we determined how large the feature vector can be in dependence of the number u of measurements and the tolerance parameter δ_e used during enrollment. We did this by evaluating the number of minutiae per finger that are reliably (i.e., u times) detected in u measurements. Since this number varies considerably among individuals and measurements, acceptable Failure To Enroll (FTE) rates can only be achieved, if the required number of reliable minutiae is considerably lower than its average value. Therefore, we evaluated the maximum number M_r of reliable minutiae that is achieved in at least 80% of all measurements. The results of this evaluation are listed in Table 1.

4.2 Minutiae matching rates

In order to configure the error correction capabilities of our scheme appropriately, it is necessary to determine the rate at which the genuine minutiae in the vault are identified during authentication. For various tolerance parameters $\delta_v = \delta_e$, we computed the biometric template set T, containing t minutiae reliably detected in

Table 1: Number M_r of reliable minutiae per finger that is found in 80% of all measurements.

u	$\delta_e = 5$	$\delta_e = 7$	$\delta_e = 10$	$\delta_e = 15$
1	63	63	63	63
2	23	32	39	43
3	18	24	31	35
4	9	16	22	27
5	6	9	15	18

u measurements and matched them with the minutiae of an (independent) query fingerprint using our matching algorithm. We did not add chaff points to the template T. The average match rate, i.e. the average ratio between the number of matches found and t, are given in Table 2.

Similarly to the number of reliable minutiae, the match rate varies considerably between different measurements. Moreover, in the presence of chaff points, the match rates slightly decrease depending on the expected number of false matches (with chaff points), as the chaff points render the correct mapping of the minutiae more difficult for the matching algorithm. (This aspect is further discussed in Section 4.5.) Therefore, a reasonably small FRR can only be achieved if k is selected slightly smaller than the expected value of $m_c - m_f$ (see Section 2.6). Our empirical evaluation suggests to set k 10%-20% smaller than this value.

For $2 \leqslant u \leqslant 4$, we obtain good match rates at a reasonable number of minutiae. Therefore, we will subsequently focus on these cases.

4.3 Effect of quality filtering during verification

As argued in Section 2.4.4, quality filtering of the minutiae in the query fingerprints aims to reduce the number of surplus minutiae, i.e., minutiae in the query fingerprints that do not match with genuine minutiae in T. We evaluated the effectiveness and eligible configuration of the filtering based on the minutiae quality

Table 2: Average match rate (in percentage) in the absence of chaff points for $\delta_e = \delta_v$.

u	$\delta_v = 5$	$\delta_v = 7$	$\delta_v = 10$	$\delta_v = 15$
1	40	50	58	64
2	66	72	78	82
3	75	81	84	87
4	81	85	88	90
5	85	89	91	92

Table 3: Expected number of minutiae in a query fingerprint after filtering with minimum quality value Q.

Minimum quality Q	Av. no. τ of minutiae
0	70
0.1	67
0.2	52
0.3	48
0.4	41
0.5	33
0.6	32

values output by the MINDTCT algorithm of NIST [WGT+07].

The average number of minutiae detected in a single fingerprint depends on the sensor used, the feature extractor algorithms, the quality of the images, and even the finger type (e.g. thumbs contain more minutiae than other fingers). In our tests, we detected an average number of 84 minutiae per fingerprint (excluding thumbs) inside ellipse $\mathcal{E}$. Based on this number and the distribution of quality values, we can estimate the expected number τ of minutiae in a query fingerprint after filtering with minimum quality value Q. The results are listed in Table 3.

The reduced average number τ of minutiae per query finger given to the matching algorithm results in a decreased average number s of surplus minutiae per finger and in less false matches (i.e., matches with chaff points). On the other hand, it may also reduce the number of correct matches (and likewise the match rate) because the minutiae filtered out could have matched with genuine minutiae in the vault. For different sets of parameters we empirically determined the decrease of the number of correct and false matches resulting from the quality filtering. An example plot is presented in Figure 1.

For larger r and smaller δ_v, quality filtering with higher values of Q results in a more drastic reduction of the correct matches. Nevertheless, for various parameters we consistently found a value Q between 0.2 and 0.3 to be optimal, reducing the false matches by approximately 30% while decreasing the number of correct matches by less than 3%.

4.4 Effect of minimum number of minutiae per finger

If the tolerance parameter δ_v is set appropriately as described in Section 4.5, the number of correct matches typically exceeds the number of false matches. On the other hand, if the matching algorithm fails to identify the correct isometry, the number of correct matches is typically significantly lower than the number of false matches. As explained in Section 2.4.5, the enforcement of a minimum number χ of

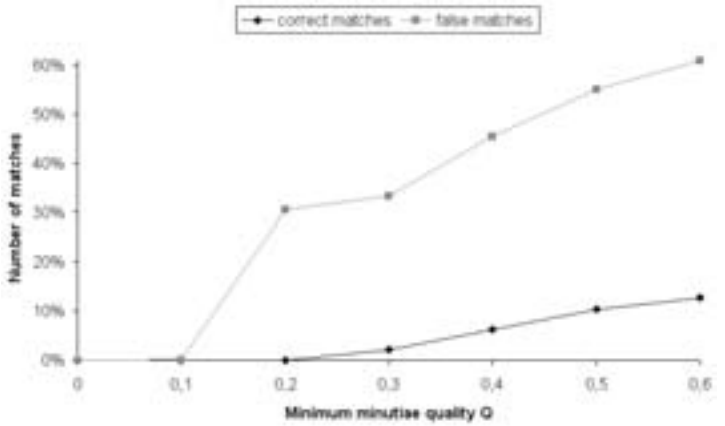

Figure 1: Reduction of the correct and false matches by quality filtering for $f = 6$, $t = 120$, $r = 400$ and $\delta_v = 7$.

minutiae per finger in the template T aims at reducing the frequency of such cases. We evaluated the effectiveness and reasonable configuration of this optimization by determining the ratio of fingers for which the number of false matches exceeded the number of correct matches for various values of the parameter χ. Furthermore, we analyzed the influence of this optimization to the FTE by determining the rate at which a finger contained at least χ minutiae and, hence, would succeeded to enroll. The results of this evaluation are displayed in Figure 2 by the curves of the match rate and the rate of successful enrollment. (Other failures of enrollment, particularly cases, where the fingers of a person contained less than t minutiae in total, were neglected.) Obviously, $\chi = 9$ already yields a considerable improvement with only moderately increased FTE rates.

The impact of the value of χ becomes particularly strong as the average number of false matches approaches the number of correct matches. As shown in Figure 3 for $f = 6$, $u = 4$, $t = 100$, $r = 600$, $\delta_e = 10$, $\delta_v = 7$ and $Q = 0.3$, where even for $\chi = 15$ the fraction between the average numbers of correct and false matches was 2.1 (as opposed to a fraction of 2.9 for the parameters of Figure 2), the average match rate steadily and considerably increases until $\chi = 15$. The decrease of the successful enrollment rate is similar to the case of Figure 2. This finding indicates that in these cases it may be worth to choose χ larger than 9 at the cost of higher FTE rates.

4.5 Balancing correct and false matches

In order to enable the minutiae matching algorithm to determine the correct isometry by which the query fingerprint is correctly aligned to the minutiae in the

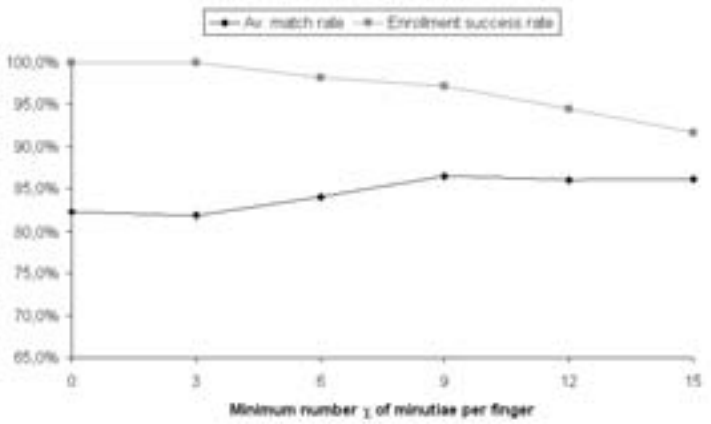

Figure 2: Impact of enforcing a minimum number χ of minutiae per finger in T to match rate and rate of successful enrollment for $f = 6$, $u = 4$, $t = 120$, $r = 400$, $\delta_e = 10$, $\delta_v = 7$ and $Q = 0$.

vault, we must ensure that, on average, the number of correct matches considerably exceeds the number of false matches. The results of Section 4.4 indicate that a fraction of 2 between the average numbers of correct and false matches already requires large values for χ which considerably increases the FTE.

In [MIK+10a], the expected number m_f of false matches is estimated by $(r - t)sV_{\delta_v}/|\mathcal{E}|$, where $V_\delta = 1 + 4\sum_{i=1}^{\lceil\delta-1\rceil}\left\lceil\sqrt{\delta^2 - i^2}\right\rceil$ is the number of integer points in the 2-dimensional plane with Euclidean norm smaller than δ and s is the average number of surplus minutiae (i.e., minutiae not matching with genuine minutiae) per query fingerprint. On the other hand, we can estimate $s \approx \tau - \mu t/f$, where τ is the number of minutiae of the query fingerprint after quality filtering.

Our experiments show that for typical parameters the average number of false matches is 20%-60% larger than these estimations imply, depending on the specific parameters. The deviation is presumably due to those outliers resulting from an incorrect determination of the isometry: if the matching algorithm is unable to detect the correct alignment, its optimization strategy with respect to the number of matches will yield extraordinary many false matches. Based on this observation, we adjust our above estimation to

$$m_f \approx 1.4(r - t)(\tau - \mu t/f)V_{\delta_v}/|\mathcal{E}|. \tag{1}$$

Yet, we expect the number of false matches to grow linearly with V_{δ_v}, which is a quadratic function in δ_v.

On the other hand, the average number m_c of correct matches is given by μt, where μ is the match rate, and therefore, grows slowly with increasing δ_v as shown in Table 2. Therefore, the selection of δ_v should carefully balance the expected numbers of correct and false matches. For $f = 3$, $u = 4$, $t = 66$, $r = 320$ and

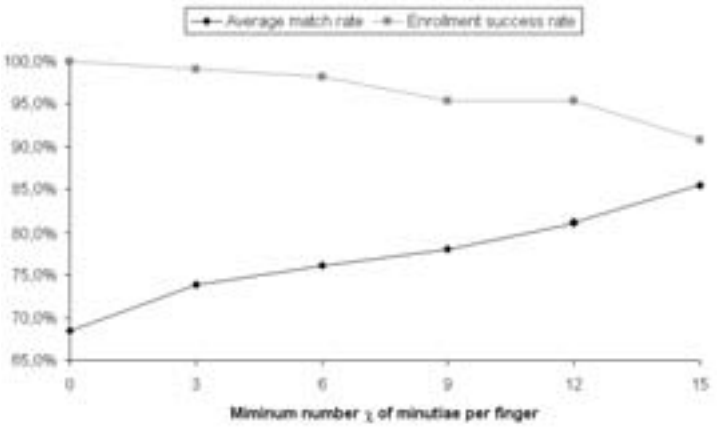

Figure 3: Impact of enforcing a minimum number χ of minutiae per finger in T to match rate and rate of successful enrollment for $f = 6$, $u = 4$, $t = 100$, $r = 600$, $\delta_e = 10$, $\delta_v = 7$ and $Q = 0.3$.

$Q = 0.3$, and for $5 \leqslant \delta_v \leqslant 15$ we estimated the number of false matches by (1) and the number of correct matches as μt using the match rates empirically determined. The results show that, for these parameters, $\delta_v \leqslant 8$ should be selected to ensure that the average number of correct matches is at least twice the number of correct matches.

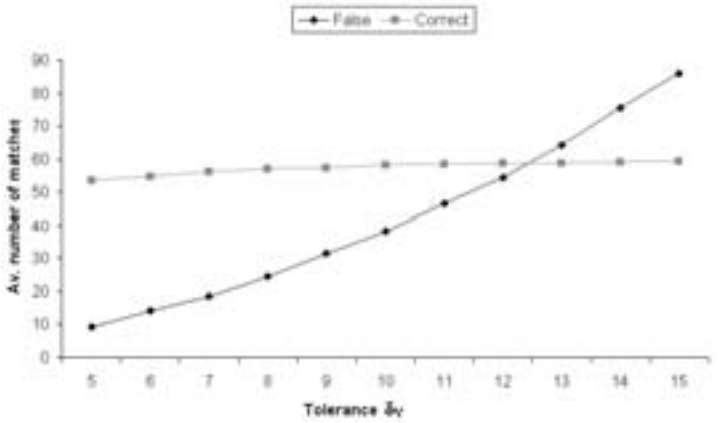

Figure 4: Impact of the tolerance parameter δ_v on the estimated average number of correct and false matches for $f = 3$, $u = 4$, $t = 66$, $r = 320$ and $Q = 0,3$.

For a smaller ratio r/t, the curves meet at higher values of δ_v, but still the accelerating growth of the number of false matches implies that $\delta_v \leqslant 8$ is a good choice.

Table 4: Parameters for a security level of 2^{Sec}.

f	u	δ_e	δ_v	t	r	k	Sec
2	2	7	5	26	240	27	70
3	2	7	5	90	351	41	97
3	3	7	5	70	360	34	97

4.6 Achievable Security

Based on our experiences gained, the example parameters listed in Table 4 have been determined to provide the indicated security level against existing attacks. We did not experimentally determine real error rates during enrollment and verification; therefore, these parameters are mere suggestions which require practical validation. We set $d = \lfloor 3/2 \cdot \delta_v \rfloor$, $Q = 0.3$, and $\chi = 9$. Furthermore, we choose $r < \lfloor 0.2 \cdot 87000/V_d \rfloor$ to avoid the attack described in [CST06] (see Section 3.1) that could significantly reduce security.

5 Conclusions

Our analysis shows that a fuzzy vault for multiple fingerprints can be very secure against template recovery from the helper data, if appropriate optimizations are applied. Filtering minutiae for reliability during enrollment and for quality during verification turn out to be particularly effective. Furthermore, enforcing a minimum number of minutiae per finger in the template significantly increases matching performance. Both optimizations are very sensitive to the respective thresholds, which must be carefully set on the basis of empirical data.

Finally, we would like to stress that our security analysis only covered template recovery attacks. Other types of attacks have been published [SB07] and need to be addressed before the scheme can be considered ready for use. We encourage research on methods to harden the fuzzy fingerprint vault against these attacks.

Acknowledgments

This work was conducted as part of the projects “BioKeyS-Multi” and “BioKeyS Pilot-DB” of the Bundesamt für Sicherheit in der Informationstechnik.

The matching algorithm was designed and implemented by Stefan Schürmans.

References

[BBGK08] Jeroen Breebaart, Christoph Busch, Justine Grave, and Els Kindt. A Reference Architecture for Biometric Template Protection based on Pseudo Identities. In *Proc. BIOSIG 2008*, volume 137 of *LNI*, pages 25–38. Gesellschaft für Informatik, 2008.

[CKL03] Charles Clancy, Negar Kiyavash, and Dennis Lin. Secure smartcardbased fingerprint authentication. In *WBMA '03: ACM SIGMM workshop on Biometrics methods and applications*, pages 45–52, 2003.

[CST06] Ee-Chien Chang, Ren Shen, and Francis Weijian Teo. Finding the original point set hidden among chaff. In *Proc. ACM Symp. on Information, Computer & Communication Security (ASIACCS)*, pages 182–188, 2006.

[GG03] Joachim von zur Gathen and J$^{'}$urgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2nd edition edition, 2003.

[JS02] Ari Juels and Madhu Sudan. A Fuzzy Vault Scheme. In *Proc. IEEE Int. Symp. Inf. Theory*, page 408, 2002.

[MIK+10a] J. Merkle, H. Ihmor, U. Korte, M. Niesing, and M. Schwaiger. Provable Security for the Fuzzy Fingerprint Vault. In *Proc. 5th Int. Conf. Internet Monitoring and Protection (ICIMP 2010)*, pages 65–73, 2010.

[MIK+10b] Johannes Merkle, Heinrich Ihmor, Ulrike Korte, Matthias Niesing, and Michael Schwaiger. Performance of the Fuzzy Vault for Multiple Fingerprints (Extended Version). eprint arXiv:1008.0807, 2010.

[MMT09] Preda Mihailescu, Axel Munk, and Benjamin Tams. The Fuzzy Vault for fingerprints is Vulnerable to Brute Force Attack. In *Proc. BIOSIG 2009*, volume 155 of *LNI*, pages 43–54. Gesellschaft für Informatik, 2009.

[NJP07] Karthik Nandakumar, Anil Jain, and Sharath Pankanti. Fingerprint-Based Fuzzy Vault: Implementation and Performance. *IEEE Trans.Inf. Forensics Security*, 2(4):744–757, 2007.

[Pla09] Rainer Plaga. Biometric keys: Suitable Uses and Achievable Information Content. *Int. J. Inf. Secur.*, 8(6):447–454, 2009.

[SB07] Walter J. Scheirer and Terrance E. Boult. CRACKING Fuzzy Vaults and BIOMETRIC ENCRYPTION. In *Proc. IEEE Biometrics Symposium*, pages 1–6, 2007.

[UJ06] Umut Uludag and Anil Jain. Securing Fingerprint Template: Fuzzy Vault with Helper Data. In *IEEE Workshop on Privacy Research In Vision*, pages 163–169, 2006.

[UPJ05] Umut Uludag, Sharath Pankanti, and Anil Jain. Fuzzy Vault for Fingerprints. In *Proc. Int. Conf. Audio- and Video-Based Biometric Person Authentication (AVBPA)*, volume 3546 of *LNCS*, pages 310–319. Springer, 2005.

[WGT+07] Craig Watson, Michael Garris, Elham Tabassi, Charles Wilson, Michael McCabe, Stanley Janet, and Kenneth Ko. *User's Guide to NIST Biometric Image Software (NBIS)*. National Institute of Standards and Technology, 2007.