# THE FUZZY VAULT FOR FINGERPRINTS IS VULNERABLE FOR BRUTE FORCE ATTACK AND COLLUSION ATTACK

Bastian Fischer

01.02.2010

# 1 Introduction

Since the Internet and its possibilities and services are growing each day, secure authentification becomes even more important for their users. Passwords are still the most used way to ensure security of authentification for the user. But in times of brute force attacks using libraries of many well known and often used passwords, more "exotic" passwords are needed which consist of hard to guess combinations of letters, numbers etc. But those are also hard to remember for the user. So the idea of an authentification method which ensures secure authentification using biometric data of the user tries to achieve the goal of security and easy handling. Fuzzy Vaults for fingerprints are used to lock a given secret with a fingerprint. They hide the secret information by adding false information, so that an attacker with the knowledge of a vault shall not be able to recover the secret. But are Fuzzy Vaults imune against brute force attacks or attacks in general?

In this report two attacks are discribed. The first shows the vulnerability of Fuzzy Vaults for brute force attacks by showing a technique to find out the encoded secret by guessing the so called "genuine set". The second attack shows how easy Fuzzy Vaults can be broken by just having knowledge of more than one of them.

## 2 The Fuzzy Vault

The Fuzzy Vault for fingerprints is an algorithm for hiding a secret string $S$ with a set of biometric data $T$. Having $T$, and of course the matching fingerprint, makes it easy for a user to unlock and recover $S$, while an attacker should have computational problems to find out $T$. But $T$ can be fuzzy, because biometric scanners do not produce 100% identical scans of the same fingerprint every time it is scanned, and also the fingerprint can be distored or rotated in some way. So it must be possible to recover $S$ by an biometric information $T'$ which is not identical to $T$ which requires error correction.

The string is prepared for the transmission in the vault as follows: Let $S$ 2 f $0, 1$g$^*$be a secret string of $l$ bits length. Now let Alice, Bob and Victor be the actors in our example. Alice wants to be identified by the string $S$ and has her finger scanned. A *locking set* L compromising the Cartesian coordinates of $t$ minutiae is selected from this finger template $T'$. Then the couples of coordinates are concatenated to single numbers $X_i = (x_i$k$y_i)$ 2 L . A finite field $\mathbb{F}_q$is chosen and attached to the vault. Then let $k' + 1 = \left\lceil \frac{l}{\log_2(q)} \right\rceil$be the number of elements in $\mathbb{F}_q$which is necessary to encode $S$ assuming $0 < \max_{X \in \mathcal{L}} X < q$. $X$ is mapped on $\mathbb{F}_q$. Then the *genuine set* G is built by choosing a polynomial $f(X)$ 2 $\mathbb{F}_q[X]$ of degree $k \quad 1 < k'$. Its coefficients encode $S$. So G looks like:

$$\mathsf{G} = \mathsf{G}(\mathbb{F}_q, S, t, k, \mathsf{L}) = \mathsf{f}\,(X_i, Y_i) : X_i \text{ 2 } \mathsf{L}\,; Y_i = f(X_i)\mathsf{g},$$

in other words the locking set L is evaluated to the genuine set G with $f$, which means G encodes $S$. To make an attack on this structure computationally hard for the attacker Victor, a large number of random uniformly distributed *chaff points* are added to the field $\mathbb{F}_q$ and mixed with the genuine set.

$$\mathsf{C} = \mathsf{f}\,(U_j, W_j) : j = 1, 2, ..., r \quad t\mathsf{g},$$

with $U_j$ 62 Land $W_j$ 6= $f(U_j)$. So finally, the common vault V is generated by shuffling G an C:

$$\mathsf{V} = \mathsf{V}(k, t, r, \mathbb{F}_q) = \mathsf{G}\,[\,\mathsf{C}.$$

If Bob now wants to recover $f(X)$ and $S$, he needs an orignal template $T$ of Alice's fingerprint. Then he will generate an *unlocking set* U which contains $X_i$ coordinates of vault points. These coordinates approximate coordinates of minutiae in $T$. To ensure an adequat approximation with different templates

of the same finger, the templates must have negligeable non linear distortions and must be aligned modulo affine transforms. Since biometric scanners do not produce 100% identical scans for the same finger every time it is scanned, the unlocking set can also be erronated. Because of that problem, some chaff points might be closer to the so scanned second template $T'$ than locking points. Also this limits the choice of very large unlocking sets. To face and solve this problem Juels and Sudan [JS] suggest Reed Solomon code for error correction.

Now Victor tries to break the vault and has the following problem: the chaff points generate a large number of subsets of $t$ elements. Their coordinates are interpolated by a set of polynomials $g(X)$ 2 $\mathbb{F}_q[X]$of degree k. So the correct vault points for the polynomial $f(X)$ is hidden between those polynomials $g(X)$. This case can be explained by the following Lemma [PM], which lays the ground for the next chapter:

**Lemma 1:** *For every* $\mu$, $0 < \mu < 1$ *and every vault* $\mathsf{V} = \mathsf{V}(k, t, r, \mathbb{F}_q)$, *there are at least* $\frac{\mu}{3} \quad q^{k-t} \quad (r/t)^t$ *random polynomials* $g(X)$ 2 $\mathbb{F}_q[X]$ *such that* $\mathsf{V}$ *contains* $t$ *couples* $(U_j, g(U_j))$. (the proof can be found in [JS])

# 3 Brute Force Attack on Fuzzy Vault for fingerprints

In the last chapter, Victor tried to break the vault and had the problem, that there was a computationaly hard number of polynomials $g(X)$ because of the chaff points. But if Victor was able to intercept a vault $\mathsf{V} = \mathsf{V}(k, t, r, \mathbb{F}_q)$ not knowing any details of the correct genuine set or anything else about the original data, he might still use the brute force strategy to recover the encoded secret $S$. For this, he needs to find as many points as the degree of the polynomial $f(X)$, which is $k$. In every try the chance, that the chosen $k$ points of the vault are also in the genuine list is

$$1/P = \frac{\binom{r}{k}}{\binom{t}{k}} \quad (r/t)^k < 1.1 \quad (r/t)^k,$$

for $r > t > 5$. The probability, that a random pair $(X, Y)$ 2 $\mathbb{F}_q^2$ lays on the graph of the searched polynomial $f(X)$ 2 $\mathbb{F}_q[X]$ is equal to $P[Y = f(X)] = 1/q$. To interpolate a polynomial of degree $k$ with Lagrange interpolation it takes $O(k \log^2(k))$ operations. Checking an additional vault point $(U, W)$

with $W = f(U)$ if it lays on the graph of the polynomial $f(X)$ can be done in $O(k)$ operations. So $K = O(\log^2(k))$ such tests can be done at the cost of one interpolation. An attack can be described by the follwing Lemma [PM]:

**Lemma 2:** Let $\mathsf{V}(k, t, r, \mathbb{F}_q)$ be a fuzzy fingerprint vault and $k < t$ be chosen by Victor. Then an intruder having intercepted $\mathsf{V}$ can recover the secret $S$ in $R = C \quad (r/t)^k$ operations, where $C < 8rk$ is the complexity of the brute force attack.

**Proof:** We know that a set of $k$ points from the locking set $\mathsf{L}$ can be found by Victor in $< 1.1 \quad (r/t)^k$ trials. To find such a set and then the secret $S$, for each $k \quad$ tuple $\mathsf{T} = (X_i, Y_i)_{i=1}^k \quad \mathsf{V} \quad$ Victor has to do

1. Compute the interpolating polynomial $g_{\mathcal{T}}(X)$. Let $K = 6.5 \quad \log^2(k)$ (Lagrange interpolation with implicit constant 6.5). So computing all the interpolation polynomials requires $< 7.2 \;\; k \log^2(k) \;\; (r/t)^k$operations.

2. Search a point $(U, W) \; 2 \; \mathsf{V} \; \mathsf{n} \; \mathsf{T}$ with $g(U) = W$. This requires as many operations as $r/K$ Lagrange interpolations. If no point is found, then discard $\mathsf{T}$.

3. If $\mathsf{T}$ was not discarded, search for a further point $(U', W') \; 2 \; \mathsf{V} \; \mathsf{n} \; \mathsf{T}$ with $g(U') = W'$. This step has the probability $1/q$. If a point is found, add it to $\mathsf{T}$ , else discard $\mathsf{T}$.

4. Proceed until either there are no points left on the graph of $g(X)$, or D points have been found in $\mathsf{T}$, where we assume, that there is a degree $k \quad 1 < D < t$ which is minimal with the property that among all polynomials $g(X) \; 2 \; \mathbb{F}_q[X]$ of degree $k \quad 1$ which interpolate vault points, $f(X)$ is the only one which interpolates with probability close to 1 at leats D points. So $g(X) = f(X)$ with high probability.

Combining all required operations by these steps and giving weights by the probabilities of occurence, one can finally say:

$$R < 7.2 \;\; (r/t)^k \;\; k \;\; K \;\; r \;\; \sum_{j=0}^{\infty} (1/q)^j = 7.2 \;\; (r/t)^k \;\; k \;\; K \frac{rq}{K(q \quad 1)} < 8.0 \;\; (rk) \;\; (r/t)^k$$

□

### 3.1 factors that influence the complexity of the brute force attack

1. If the minutiae are assumed to be uniformly distributet over the template, $r$ and $t$ are scaled by the same factor, so that $r/t$ does not change, and also the complexity. So restricting the region of interest from which Victor chooses points for his unlocking set does not make any sense in this case.

2. A higher degree $k$ of the polynomial $f(X)$ increases the complexity, but also requires large unlocking sets. Since the quality of a fingerprint scanner can not be guaranteed to be at a comparable level for all scanners, this could lead to problems.

3. More chaff points means more points overall and more tries in brute force, so that the complexity grows. But there has to be a distance between chaff points, which ensures that the minutiae locations are still findable. And, of course, the image size limits the number of chaff points, too.

4. Reducing the size $t$ of the genuine list makes the genuine points harder to find, which increases complexity, but it must not be reduced below the required minimum, so that one can still interpolate.

## 4 The Collusion Attack

A second attack on the Fuzzy Vault for fingerprints scheme is the collusion attack. We assume that Bob carries multiple smartcards issued by different organizations. We also assume, that Victor has access to these multiple vaults. Now there are three possible scenarios:

1. All vaults are locked with Bob's fingerprint, so they have the same locking set $\mathsf{L}_i = \mathsf{L}$.

2. All vaults are locked with the same key, and also the secret message is the same $S_i = S$.

3. Only the secret message is the same, but locked with different keys.

## 4.1 Collusion Attack with $\mathsf{L}_i = \mathsf{L}$

Victor has access to $n$ vaults $\mathsf{f}V_{\mathcal{L},1}, V_{\mathcal{L},2}, ..., V_{\mathcal{L},n}\mathsf{g}$ all locked with the same locking set $\mathsf{L}$. Each vault contains $r$ points, $V_{\mathcal{L},j} = \mathsf{f}(x_{i,j}, y_{i,j})\mathsf{g}_{i=1}^{r}$. We assume that the vault size is not maximal (i.e. $r < q$). Then Victor can reduce the number of candidate polynomials by exploiting three properties:

1. the locking set is the same for all vaults and "visible" to the attacker
2. chaff points are generated randomly and independently of the key
3. non-maximal vault size means chaff points vary from vault to vault

The goal of the Collusion Attack is to reduce the number of spurious polynomials by identifiyng and removing chaff points. So, where $X_j = \mathsf{f}x_{i,j}\mathsf{g}_{i=1}^{r}$ do the following algorithm[TPM]:

1. Create an empty set $X'$
2. FOR i := 1 to r DO
3. IF $x_{i,1}$ 2 $X_j$ for all j THEN
4. $X'$:= $X'$ [f $x_{i,1}$g
5. END IF
6. END FOR
7. $V_{\mathcal{L},j}^{eff}$ := f$(x_{i,j}, y_{i,j})$ 2 $V_{\mathcal{L},j}$j$X_{i,j}$ 2 $X'$g, denoted the $j^{th}$effective vault
8. $r_{eff}$ := $\left|V_{\mathcal{L},j}^{eff}\right|$ = j$X'$j, denoted effective vault size

The algorithm searches for x-coordinates, which are the same in all testet vaults. Since they were locked with the same key, there is a high possibility, that some of them are identical and belong to the locking set $\mathsf{L}$. Because of randomly generated chaff points in each vault, it is easy to assume, that their x-coordinates are different. Indeed the algorithm identifies every point which does not appear in all $n$ vaults as chaff point and removes them from the effective vault. At the end the effective vault only consists of points which were in all $n$ vaults. The more vaults are known, the more chaff points can be removed, so that $X'$ approaches $\mathsf{L}$.

## 4.2 Collusion Attack with $\mathsf{L}_i = \mathsf{L}$ and $S_i = S$

If Victor knows two or more vaults locked with the same key and containing the same secret, the key can be revealed even easer.[TPM]

1. Create an empty set $V_{eff}$
2. FOR $i := 1$ to $r$ DO
3. IF $(x_{i,1}, y_{i,1})$ 2 $V_{\mathcal{L},j}$ for all $j$ THEN
4. $V_{eff} := V_{eff}$ [f $(x_{i,1}, y_{i,1})$g
5. END IF
6. END FOR
7. $S' := Decode(V_{eff})$
8. $V_{eff}$ is the effective vault
9. $r_{eff} := \mathsf{j}V_{eff}\mathsf{j}$ is the effective vault size

Since $x_{i,j}$ and $y_{i,j}$ are related in the same common polynomial $f(X) = s_1 + s_2x + s_3x^2 + ... + s_kx^{k-1}$these points have to be identical in all vaults. The randomly generated chaff points however do not have a high probability to be identical across the vaults. Thus this attack might even work if Victor only knows $n = 2$ vaults. In this case, the attack would also work with a maximal vault size, because the true points have fixed $x_i$ and $y_i$ while the chaff points still are matched to random $y_i$, though their $x_i$ values are the same across all vaults. Victor could use this fact to find out all chaff points and remove them easily.

## 4.3 Collusion Attack with $S_i = S$

If Victor knows $n$ vaults locking the same message, but with different keys, it is not as easy as in the previous two cases. Assuming that the probability distribution of the locking sets is well defined, so that the probability of a particular point being a true point is fixed, the probability of a point $x$ being in the locking set is defined as $p_t$. With this probability the point $(x, f(X))$ appears in the vault. Otherwise, when $x$ is not part of the locking set, its

probability being a chaff point is $p_{chaff} = c/v$ , with $c$ as the number of chaff points to be added and $v$ the number of possible values for them.With the probability $(1 \quad p_t)p_{chaff}$ we would have a point $(x, y')$ in the vault, with $y'$uniformly distributed over $\mathbb{F}_q$ n f $f(X)$g. We assume that the vault is maximal and $p_{chaff} = 1$ for the following. $P(x, y' = y) = (1 \quad p_t)/(q \quad 1)$, where $y$ is different from f(X). If $p_t \quad 1/q$, then $p_t \quad (1 \quad p_t)/(q \quad 1)$.

Let $N \quad k$ be points where $p_t \quad 1/q$, given $n$ vaults f $V_1$,$V_2$, ..., $V_n$gcontaining the same secret $S =$ f $s_i$g$_{i=1}^k$, where each vault contains $r$ points, $V_j =$ f $(x_{i,j}, y_{i,j})$g$_{i=1}^r$. Then perform algorithm [TPM]:

1. Create a $q$ by $q$ table $Count(x, y)$ and initialize all values to 0
2. FOR $j := 1$ to $n$ DO
3. FOR $i := 1$ to $r$ DO
4. $Count(x_{i,j}, y_{i,j}) := Count(x_{i,j}, y_{i,j}) + 1$
5. END FOR
6. END FOR
7. $Count(x, y)$ contains the distribution data over the $n$ vaults.
8. Create an emtpy set $V_t$
9. FOR ALL $a$ 2 $\mathbb{F}_q$DO
10. Find the mean $u_a$ of $Count(a, y)$
11. the estimated true point is $(a, b)$ where $b := y'$where j$Count(a, y') \quad u_a$j is maximal
12. $V_t := V_t$ [ f $(a, b)$g
13. $Q(a) :=$ j$Count(a, b) \quad u_a$j
14. END FOR
15. $V_t$ is the estimated true vault, which contains the points most likely to lie on the message polynomial.
16. $Q(a)$ is a measure of the quality of the point retained

17. $V_{eff} := s$ pairs of $(a, b)$ with the highest $Q(a)$

18. $S' := Decode(V_{eff})$

The found points that way have a high probability to be true points. It will consistently paired with a particular $y$ value. When $n$ is large and Victor knows many vaults, also the never paired points, where $p_t \quad 1/q$ can be usefull. To find out which estimated true points are most likely real true points, Victor has only to watch the frequency of changing in the $y$ values. More changes means lesser reliable and lesser changes means more reliable.

# 5 Ideas to improve the security of Fuzzy Vault

1. **Using more Fingers.** As we have seen, the parameters $r, k, t$ , which have the most influence on the security factor, depend on image size, variance of minutiae location and average number of reliable minutiae. Just using two fingers to create the vault would square the security factor by increasing the number of trials.

2. **Non-random Chaff points**. Randomly generated chaff points can, if their number grows, get to close to true points or, because of the randomness, one cannot garantuee a particular distance between them. Choosing a non-random chaff point generation would make it easier to maximize their number in the field. Also a Collusion Attack could be prohibided because of identical x-Values of the chaff points.

3. **Using Quizzes as additional minutiae information.** Minutiae could be upgraded with some extra information, such as orientation, neighboring data, incoming lines, etc. One could attach a quiz to each minutiae, which can be easily solved by Bob, but not by Victor, because Victor has to handle $b$ bits of new uncertain data per minutiae. With a polynomial degree of $k$, the security would be increased by the factor $2^{kb}$. An example is the addition of the orientation data to a minutiae. Let $X$ be the concatenated coordinates of a fixed minutiae and $\alpha$ its orientation in a granularity of $\pi/n$ for a small integer $n$. Then the value $\beta$ is introduced to the vault, so that a minutiae is represented by $(X, Y, \beta)$. When Bob wants to unlock the vault, he computes the integer $0 \quad j < n$ such that $j\pi = n\alpha \quad n\beta mod\pi$. $j$ encodes a certain

transformation $Y' = T(Y)$ of the received value $Y$ and interpolating the value will be set to be $Y' = f(X)$. The creator of the vault controls $\beta$ and can choose it such that $j$ can easily and safely be recovered by the genuine user. Chaff points are also transformed, they get a randomly chosen $\beta$. However changing the $Y$ values will not prohibit a Collusion Attack, since it checks for identical $X$ values.

4. **Using maximal vault size.** In this case a vault will have all passible $x_i$ values, if locking set or not. So the chaff points do not vary, which protects the vault from a Collusion Attack, as long as the same secret is not encoded twice with the same locking set.

# 6 Conclusion

The Fuzzy Vault for fingerprints is vulnerable to a simple bruteforce attack in general and also even easier to the Collusion Attack, as long as the attacker has the knowledge about two vaults locked with the same key. This lack of security demands solutions for the future, in which the complexity of a brute force attack is increased, and/or extra information is added to the minutiae. Also all this can still be limited by the biometrical hardware.

# References

[PM] Preda Mihailescu: The Fuzzy Vault for Fingerprints is vulnerable to Brute Force Attack, 22. August 2007

[TPM] Hoi Ting Poon, Ali Miri: A Collusion Attack on the Fuzzy Vault Scheme, January 2009, IseCure Magazine Volume 1, Number 1 (pp. 27-34)

[JS] A. Juels and M. Sudan: A fuzzy vault scheme, In Proc. of the IEEE International Symposium on Information Theory (2002), p. 408, Ed. A. Lapidoth and E. Teletar.