Bipartite Biotokens: Definition, Implementation, and Analysis

W.J. Scheirer[2,1,†] and T.E. Boult[1,2,†,⋆]

[1] Univ. of Colorado at Colorado Springs, Colorado Springs, CO - 80918
[2] Securics Inc, Colorado Springs, CO - 80918

Abstract. Cryptographic transactions form the basis of many common security systems found throughout computer networks. Supporting these transactions with biometrics is very desirable, as stronger non-repudiation is introduced, along with enhanced ease-of-use. In order to support such transactions, some sort of secure template construct is required that, when re-encoded, can release session specific data. The construct we propose for this task is the *bipartite biotoken*. In this paper, we define the bipartite biotoken, describe its implementation for fingerprints, and present an analysis of its security. No other technology exists with the critical reissue and secure embedding properties of the bipartite biotoken. Experimental results for matching accuracy are presented for the FVC 2002 data set and imposter testing on 550 Million matches.

1 Introduction

Template protection schemes solve an important problem inherent in biometrics: the threat of permanent feature compromise. Biometrics, unlike passwords or PINs, cannot be changed during the course of an individual's life. Many different schemes have been proposed in the literature [1] for template protection. Certain classes of these schemes support key release upon successful matching. *Key-binding* biometric cryptosystems bind key data with the biometric data. *Key-generating* biometric cryptosystems derive the key data from the biometric data. Both classes support a key release that may be used for cryptographic applications, including standard symmetric key cryptography, where key storage is problematic. Biometrics coupled with traditional cryptography presents several advantages, including ease-of-use and stronger non-repudiation properties. Unfortunately, the work to date has not been able to support cryptographic transactions as described in [13]. Further, the actual security and matching accuracy of even the most popular schemes is questionable.

The *fuzzy vault* scheme [2] is a key-binding biometric cryptosystem that hides a secret κ within a large amount of chaff data. Briefly explained, Alice places κ in a fuzzy vault and locks it using a set A of elements from some public universe U. To unlock the vault, and retrieve κ, Bob must present a set B that substantially overlaps with A. To protect κ, it is encoded as coefficients of a polynomial p. A set of points R is constructed from A and $p(A)$. In addition to these points, chaff points C are randomly generated and inserted into R. The

⋆ Work supported in part by NSF STTR† 0750485 and NSF PFI* 0650251

Pre-print of paper to appear in LNCS 5558 *ICB 2009* (http://www.springerlink.com/content/y6854128xj40w511/). The original publication is available at http://www.springerlink.com.

subset matching problem is solved with an error correction code. To decode κ, if Bob's B approximately matches A, he can isolate enough points in R that lie on p so that applying the error correcting code he can reconstruct p, and hence κ. Several implementations of biometric fuzzy vaults have been produced, including a fingerprint implementation [3], a password hardened implementation [4], and a multi-modal fingerprint & iris implementation [5].

Multiple serious attacks have questioned the security of fuzzy vaults. The work of [11] introduces three attacks against a variety of secure template technologies. For fuzzy vaults, the attack via record multiplicity (ARM), surreptitious key inversion (SKI) attack, and substitution attacks all apply. The authors of [5] concede that the fuzzy vault "is not a perfect template protection scheme" because of the attacks of [11], yet the security analysis presented in [5] does not consider their impact. Password hardened fuzzy vaults [4] were introduced in response to the ARM attack, but still fall prey to the SKI attack, facilitating recovery of the original biometric data, and the substitution attack, allowing the placement of a backdoor into the template. Other, brute-force oriented, attacks against fuzzy vaults have included CRC checks [6], and chaff point identification [7]. On the issue of performance, the published results have been promising, albeit achieved with very limited testing. How the matching accuracy of fuzzy vaults scales to realistic amounts of data has yet to be shown.

The *fuzzy extractor* scheme [8] is a key-generating cryptosystem that binds some random data with the biometric data to produce a unique key. A fuzzy extractor incorporates a *secure sketch* construct to allow the precise reconstruction of a noisy input w given an instance of the sketch s and a sample w'. A secure sketch SS bound with a random number i forms the basis of the fuzzy extractor instance P, which returns a key R, when approximate input matching is successful. Given (questionable) assumptions, [8] shows that in an information theoretical sense, that fuzzy extractors could achieve entropic security, with P and R leaking no information that helps to predict w. The security analysis of [8] is largely constrained to modeling the probability of an attacker guessing R, and the effects of key generation on this probability.

While theoretical security analyses may be important, in biometrics, the operational security is tied to the GAR and FAR. For effective security a system needs the FAR to be less than 1 in *millions* or *billions*. Despite the formal models of security in [8], an impostor may be able to achieve a false match releasing the key. This security is a constraint of the matching algorithm, not just the template protection scheme. To date, there is no published work on the GAR/FAR performance of fuzzy extractors. Moreover, fuzzy extractors may suffer from practical constraints during error-prone data collection [10], making it difficult to generate a key that is both stable and highly random.

Revocable biotokens [12] have emerged as a different solution to the template protection problem, and have been described as being able to support key release [13]. For any biometric data that can be split into stable and unstable components, the stable portion can be encrypted in a reliable fashion, while the unstable portion is left in the clear. This provides for the definition of a biotoken

transform that scales/translates the data, and then separates it into a quotient q and modulus or remainder, r. Since q is stable, it can be encrypted or hashed for both probe and gallery data, and require an exact match. This transform induces a distance measure in encoded space: first test if the encoded q values are identical; if they are, then the residuals r are then used to compute distance. In this paper, we analyze secure key release from revocable biotokens.

This paper introduces the implementation details of the bipartite biotoken construct. In Sec. 2, we review the definition of bipartite biotokens, as introduced as a general theoretical construct in [13]. With this definition, we go on to summarize an implementation of fingerprint bipartite biotokens in Sec. 3, and present a security analysis of this implementation in Sec. 4. Finally, in Sec. 5, we experimentally show that bipartite biotokens outperform existing secure template data release mechanisms, and have *useful genuine accept rates when set for zero false accepts in over 550 Million imposter trials.*

2 The Definition of Bipartite Biotokens

The notion of data splitting to support revocable fingerprint biotokens was introduced in [12]. Using this knowledge, and the concept of public key cryptography, we can develop the re-encoding methodology for revocable biotokens. The re-encoding property, introduced in [13], is essential for supporting a viable transactional framework - tokens with unique data must be generated quickly and automatically to support the transaction. Bipartite biotoken generation from a stored biotoken allows the required data release when matching against tokens generated from original biometric features during the course of the transaction.

Assuming the biometric produces a value v that is *transformed* via scaling and translation to $v' = (v-t)*s$, the resulting v' is split into the overall stable component q, and the the residual component r. In the base scheme, for a user j, their residual $r_j(v')$ is left in the clear. The amount of stable & unstable data is a function of the modality being considered. For the initial transformation $w_{j,1}(v', P)$ of q, a public key P is required. For nested re-encodings, w_j is re-encoded using some transformation function T (which may be a hash function, or another application of public key cryptography) creating a unique new transformation for each key that is applied: $w_{j,1}(v', P),\ w_{j,2}(w_{j,1}, T_2), \ldots, w_{j,n}(w_{j,n-1}, T_n)$

Using public key cryptography, the nesting process can be securely invertible if the private key associated with the first stage of encoding is available. With this nesting in mind, we can define three properties for the bipartite biotoken:

1. Let B be a secure biotoken, as described in [12]. A bipartite biotoken B_p is a transformation $bb_{j,k}$ of user j's kth instance of B. This transformation supports matching in encoded space of any bipartite biotoken instance $B_{p,k}$ with any secure biotoken instance B_k for the biometric features of a user j and a common series of transforms P, T_2, ..., T_k.
2. The transformation $bb_{j,k}$ must allow the embedding of some data D into B_p, represented as: $bb_{j,k}(w_{j,k}, T_k, D)$.
3. The matching of B_k and Bp_k must release D if successful, or a random string r if not successful.

3 The Implementation of Bipartite Biotokens

The implementation of the bipartite token, Fig. 1, is an extension of the concepts of revocable biotokens [12] and fuzzy vaults [2], which are prerequisite for a solid understanding as in the limited space we focus on the key advances. There are four major advances in the bipartite biotoken implementation:

1. The bipartite representation implements Reed-Solomon for error correction
2. The bipartite representation uses biotoken encoded "pair rows", which are rotation and translation independent
3. The bipartite representation does *not* store the points at which the embedded polynomial is evaluated
4. The bipartite representation allows for multiple simultaneously embedded polynomials, supporting larger keys with lower numbers of matching pairs.

While the original fuzzy vault work suggested the use of Reed-Solomon (RS) codes, we are unaware of any fingerprint fuzzy vaults that have actually implemented them, probably because of the inherent difficulty of alignment, ordering issues, and the high potential error rate. Our implementation uses an RS code with varying levels of error correction selectable at encoding time. For efficiency, we choose to work over $GF(2^8)$, where the coefficients and evaluation points are all 8 bit quantities. We represent the data D to be stored as a K-byte block, with E bytes of error correction, yielding a total payload block $B = K + E$. The polynomial encodes the B bytes of data. The RS polynomial representing the B byte payload body is then evaluated at a set of points, with the value of the resulting polynomial being stored. This allows for a very fast implementation, with the average matching and key extraction attempt requiring less than 1 millisecond on a 3Ghz processor, where we use pre-computed gallery files and start

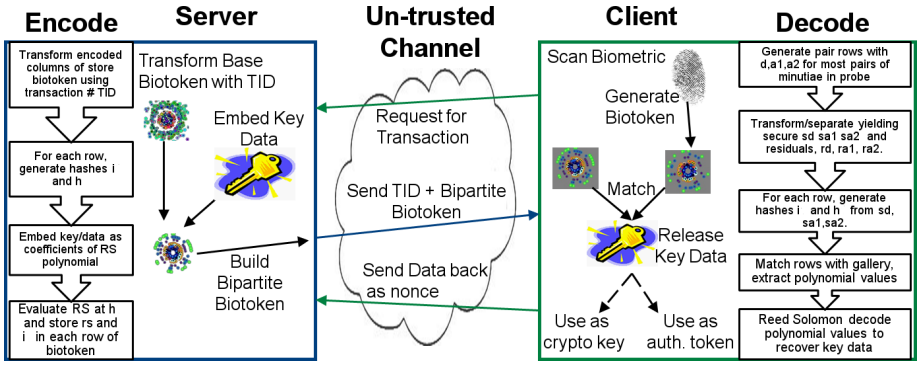

Fig. 1. Sequence diagram for the bipartite biotoken. Since the embedded data can be unique on a transactional basis, a variety of cryptographic protocols can be supported [13]. The embedded data can be a nonce that is sent back to the server for validation. It can also be a one-time token that is used for authentication. Or, in a more traditional application of key-binding schemes, it can be a symmetric or public cryptographic key. All are advantageous when the communications channel is un-trusted; only a legitimate party can unlock the embedded secret.

from minutiae for the probe. With this, we can easily vary both the key size, up to 1024 bits, and the level of error correction, with little impact on speed.

Using the "pair row" representation of the Bozorth-like matcher of [12], we have a representation that is inherently rotation and translation invariant. With the biotoken encoding of a row pair we have the raw distance and angles separated and the stable parts of those numbers are protected. Let d, a_1 and a_2 be the distance and angle fields of the row, and let sd, sa_1 and sa_2 be the stable components of these with rd, ra_1 and ra_2 the reflected modulus [12] residuals.

For polynomial evaluation, we hash the 24 bits of sd, sa_1 and sa_2 into i, an 8 bit quantity that is stored in the gallery. The value i is then hashed, per transaction, a second time to define the point at which the polynomial is evaluated. To support multiple key columns, we evaluate this hash h for different polynomials yielding values $rs_1 \ldots rs_4$. Note the evaluation point/hash value h is not stored.

The result is an "encoded bipartite row" that contains the unprotected fields and 6 protected fields (the encoded stable field w used for matching, index i and 4 columns of evaluated polynomials). We require at least 14 rows, padding the key if it does not require 4 columns to represent it. The location of the w is randomized per row. The evaluated RS polynomials for the 4 key columns, $rs_1 \ldots rs_4$, follow w using a circular mapping of the 6 slots. For example, if the random index was 3, then the sequence would be: $[rs_3, rs_4, w, rs_1, rs_2, i]$

When matching a probe, the system creates all the fields for each of its rows, including the "un-stored" hash value (h) for polynomial evaluation. A probe row potentially matches a gallery row if it finds a matching w among the encoded fields and the residuals (rd, ra_1, ra_2) are within threshold. This test is necessary, but not sufficient, for a correct match. With w identified, the algorithm can then extract the evaluated polynomial values, $rs_1 \ldots rs_4$. If w is incorrectly identified, if the row is an accidental match, or if the underlying hash value (h) is incorrect (because of a random collision in generating/matching w), some values labeled $rs_1 \ldots rs_4$ will be extracted, but will be incorrect. Prints will produce many potentially matching rows, usually (determined empirically) 200-800 if a true match and 50-600 if a non-matching print. The second stage of our Bozorth-like matching is generation of a consistent subgraph from the potentially matching rows. This results in selection of a set (20-70) of mostly correct matched rows. Without an effective way to select probable rows from the set of potentially matching rows, the level of error correction or search needed would be impractical (e.g., $\binom{200}{20}$ is 10^{27}). We extract the k values for each of the j key columns and obtain a set of hash evaluation points h_j and their Reed-Solomon polynomial evaluations $rs_{j,k}$ at the associated points.

Now comes one of the important implementation details, addressing both security and efficiency. One could effectively improve robustness by increasing the level of ECC, but doing so increase the ease with which an attacker can crack the key. Instead we use a two level hashing to improve robustness. Our two level mapping will, in general, map multiple $sd, sa1, sa2$ sets to the same index. We implemented a procedure to collect the multi-values during the mapping, check for consistency and use that redundancy to help resolve any conflicts that arise

when noisy data is mapped. The result of the mapping and consistency check is a vector of length B polynomial values (some of which may be missing) that holds the values of the evaluated RS polynomial for each location. The vector B, with gaps marked, is fed into the RS decode function, which allows us to recover D with up to g gaps and e errors, as long as $2g + e < E$, where E is the number of ECC bytes used. Each key column is recovered separately, with larger keys being the concatenation of multiple columns. For added security, a checksum is compute over the 6 unprotected columns the gallery biotope. The data D are XORed with a checksum before embedding, and again after decoding, and this prevents any tampering with the biotoken.

4 Security Analysis of Bipartite Biotokens

A security analysis of the underlying revocable biotokens was presented in [12]. The security analysis of bipartite extension is twofold - the analysis of the impact of the attacks of [11] and the brute force attack necessary to recover D.

The bipartite approach prevents the ARM, SKI, and the blended substitution attacks of [11]. For the ARM attack, we refer back to the implementation details of Sec. 3, where we introduced the hashing methodology to protect sd, sa_1 and sa_2 through data reduction ambiguity and transaction specific hasing, and the further RS polynomial encoding of the resulting hash. Thus, an attacker with access to the RS polynomial encodings cannot correlate between different biotokens - the evaluation points are never found within the encoding. For the SKI attack, we again note that the encoding binding D is tied to a hashed form of the secure data, which does not change the security analysis of [12]. The blended substitution attack is solved through the use of the specially crafted embedded data wherein D is XORed with a checksum of the stored biotoken. Thus, the system automatically checks for tampering by XORing the computed checksum of the biotoken with the released data before returning the key. If the checksum is correct, the valid D will result. Bipartite biotokens are still susceptible to a straight substitution attack, whereby an attacker replaces the columns of the biotoken with their own. However, this is effectively a denial of service for the legitimate user and hence detectable.

To recover D, an attacker would have to estimate B with a sufficiently small number of errors. At a minimum this requires K correct rows, where for each row the attacker would have to "label" the fields in each row, they would have to guess the hashes/indices which serve as the evaluation point for the polynomial. For each row there are 5 choices for ordering, and 2^8 possible hash values per key for a total ambiguity of 1280 per-row per key. The attacker would have to simultaneously recover and order K of these. It is likely there may be a reduction from the full hash space to the evaluation index of the RS polynomial, which may reduce the per-row ambiguity down to K^{5*B}, so if $K = 16$ and $E = 6$, it may reduce the attack effort to 2^{480} attempts. Thus, as we shall see, this is not the limiting aspect of the bipartite biotokens security.

To date, security analyses of protected templates presumed the attacker cannot find added constraints to relate the data or exploit non-uniformity in the dis-

tributions. An analogy would be discussing string entropy for password strength, knowing that the distribution of actual passwords is far from uniform; dictionary attacks and rainbow tables are often very successful. Prior works [8] [9] have used "entropy" models to address this and suggest security bounds. The problem with this is that measuring entropy depends on how the well the data is "coded" and the models of interdependence. It is difficult to accurately estimate or bound entropy from below, which is what is needed for a security estimate. In practice a likely more intelligent attack exists against any biometric protection scheme than simple brute force, which is to use a large amount of biometric data to see if any of the imposters can release the encoded data. This *doppleganger attack* is the biometric equivalent of a dictionary attack. The hundreds of thousands of prints publicly available provide at least a basic doppleganger dictionary.

If the False Accept Rate (FAR) of a system is 1 in X attempts, then a doppleganger attack consists of trying sufficiently more than X different attempts. When papers such as [8] [9] prove they are "secure" or others [3] [4] [5] claim they have N bits of security but have measurable FARs greater than 1 in 2^N, then one can only conclude the assumptions underlying the proofs of the security models are fundamentally flawed. A FAR of zero, tested on $< 2^N$ items (for example, [3] [4] tested on $< 2^{14}$) only documents N-bit security. Since this depends on experimental analysis we return to our analysis of the doppleganger attack against bidirectional biotokens after presenting the experimental results.

5 Experimental Results

To test the matching accuracy and security of the implemented bipartite biotokens, we ran a series of large-scale tests for the FVC 2002 [14] data set, varying a series of parameters. These parameters include bytes of error correction, released data size, size of the probe biotoken, and size of the gallery biotoken. Most importantly, many different embedded data sizes were tried, from 112 bits to 1024 bits. Support for "large key" sizes has not been available for any published template protection scheme to this point. Large key sizes here include 512 bits and 1024 bits, both of which are suitable for public key cryptography. All results we report are at single points of the ROC curve, where the FAR is 0, in order to mitigate doppleganger attacks described in Sec. 4. Our focus in this paper is high security; other points on the ROC curve represent opportunities for attack, as the FAR grows larger. In order to facilitate this requirement, all experiments reported use 6 bytes of ECC or less, except for the 1024 bit experiments, which allow larger ECC and still maintain a 0 FAR on large sets. Many different probe and gallery size combinations produce the same GAR, for the same ECC level.

First, we compare revocable biotokens to three fuzzy vault approaches presented in [3] and [4] for three different released data sizes. As in those papers, part of FVC2002 DB2 was used, providing 100 distinct probe/gallery pairs, yielding 100 genuine matches and 9,900 impostors trials. Table 1 contains the results, with bipartite biotokens showing significant improvement. Moreover, the best performing fuzzy vault scheme, the mosaic with 2 queries, incorporated information from 4 different print impressions (2 to build the mosaic, and 2 for

the queries), while bipartite biotokens outperforms it with 1 probe and 1 gallery. GAR numbers reported for the fuzzy vaults have been adjusted to count reported "failure to capture rates" as mis-detections (as they operationally are).

Support for standard cryptographic key sizes is of primary interest. Thus, in Table 2 we show results for two common symmetric key sizes (192 bits and 256 bits) and two common public key sizes (512 bits and 1024 bits), for the same protocol as in Table 1 using FVC2002 for DB1 and DB2. A drop in performance is noted for the 1024 bit data size, but we note the GAR for DB2 is better than the results reported by [4] for a much smaller embedded data size. While the small test presented in Tables 1 & 2 indicates promising performance, it is inadequate to gauge real operational performance. For instance, any "0% FAR" reported is really just FAR $< 2^{-14}$.

For a meaningful security analysis we need orders of magnitude larger imposter testing, and multiple attempts at entry. Thus, we created a much larger test out of DB1 and DB2, including all available images per finger. For true matches, we have 200 unique fingers. For the impostor test data DB1 and DB2 provide a total of 158,400 impostor attempts from the same collection. To show the operational security of bipartite biotokens and protection from doppleganger attacks, we extended our tests with mixed data from FVC 2002 and 2004, the rolled prints in NIST DB29, and NIST DB14. This provides a doppleganger dictionary of over 63,000 images used to attack each gallery entry, yielding potentially billions of non-match attempts of which we have completed 550 Million so far. Scanned/rolled prints are justified for use in the attack, even if not operationally relevant, as they provide more minutiae. Imposter/FAR testing also included rolled against rolled. Data was processed with *mindct* to extract minutiae, keeping the highest 150 quality minutiae found.

	112 Bits		128 Bits		160 Bits	
	GAR %	FAR %	GAR %	FAR %	GAR %	FAR %
F.P. Fuzzy Vault	89	0.13	89	0.01	84	0
F.P. FV, Mosaic with 2 Queries	96	0.24	95	0.04	89	0
Password Vault	88	?	86	?	79	?
Bipartite Biotokens	97	0	97	0	97	0

Table 1. A comparison of different template protection schemes on the FVC2002 DB2 data set for three small released data/key sizes. The password vault experiments of [4] presume the attacker does not know the password; the FAR rates when the attacker does know this information (as we presume in our tests) are not given but are known not to be zero. Bipartite Biotokens provide significant improvements over the three fuzzy vault schemes of [3] and [4]. Bipartite biotoken error correction parameters were 5 bytes for 112 bits, 2 bytes for 128 bits, and 6 bytes for 160 bits.

FVC02 DB#	192 Bits		256 Bits		512 Bits		1024 Bits	
	GAR %	ECC	GAR %	ECC	GAR %	ECC	GAR %	ECC
1	97	5	94	2	95	5	77	10
2	97	2	97	2	92	6	82	9

Table 2. Results for larger key sizes appropriate for public key cryptography. Best GAR and ECC sizes yielding FAR=0 for FVC02 DB1 & DB2.

Recovery Attempts	128 Bits GAR %	256 Bits GAR %	512 Bits GAR %	1024 Bits GAR %	1024 Bits 9B ECC GAR %
1	93.0	93.0	91.0	63.0	78.0
2	97.5	97.5	97.0	74.0	87.0
3	98.0	98.0	97.5	77.0	89.0
4	98.5	98.5	98.5	77.5	89.0
5	99.0	99.0	99.0	79.0	89.5
6	99.0	99.0	99.0	81.0	89.5

Table 3. Multiple recovery attempts for a test gallery of 200 unique fingers from FVC 2002 DB1 and DB2. 6 bytes of error correction was used with both the probe and gallery biotoken size of 8000 bytes, except for right column, which used 9 bytes ECC, 8000 byte gallery and 20,000 byte probe sizes. These are the same parameters as our large test below, with 0 FAR over more than 550 Million imposter trials.

For a more realistic model of usage, we are interested in assessing the effect of multiple attempts to match on the GAR - a realistic scenario, as the user is given multiple attempts to match in an operational system. For multiple matching attempts using the 7 other prints for each gallery, the prints were attempted in order of "quality," as defined in [14], from best to worst. [3] uses a similar methodology for allowing 2 attempts to match, but also fuses 2 images for probe and gallery. Results in table 3 show good initial performance but major operational improvement with multiple attempts (the 7th did not produce different results from the 6th). Since 1024 bit keys require more matching, we can improve the GAR by increasing ECC and token size, while maintaining its zero FAR rate. Thus we report 1024 bits using 9 bytes of error correction and larger probes, yielding an acceptable GAR rates on 2 or more attempts.

When using appropriately chosen keys this experiment shows an acceptable GAR with *Zero False Accepts from processing 550 Million impostor trials to date*, with 6 Bytes of ECC, 128 bit, 256 bit, or 512 bit keys and 8000 bytes for the probe and gallery. This unprecedented "single finger" performance can be attributed to the biotoken transform dimensional lifting [12] combined with the novel key embedding/recovery process with error correction. Multi-finger implementations would further enhance the security. As any biometric security analysis should, our testing presumes the attacker has all passwords and all non-embedded keys. Operationally these are generally unknown. Including the security gains of the multiple non-embedded keys and passwords would increase the effective operational security well beyond the tested FAR.

This experiment also highlighted the need to analyze weak pseudo-random number generators and *weak keys* - well known issues in crytposystems.[1] Any biometric template protection with keys or random data, especially if using error correction, must address these issues as weak keys/seeds can drive up the FAR and reduce security. For example, the FAR drops to about 1 in 10 million with random keys/seeds. Our results reported are for appropriately chosen keys. As keys and random number usage impacts the biotope transforms, it is very complex to analyze; weak keys will be the subject of a future paper.

[1] See Sec. 14.10 of B. Schneier, *Applied Cryptography* J.Weily&Sons, 1996.

6 Conclusion

The key-binding biometric cryptosystem problem is a challenging, yet essential aspect of the template protection domain. In this paper, we introduced the implementation details that are necessary to build a bipartite biotoken, which supports many different cryptographic protocols, as well as a thorough security analysis covering a range of common attacks. Our experiments show a significant improvement in accuracy and embedding capacity over the most recent published results for several implementations of the fingerprint fuzzy vault. With an 8KByte tokens securely storing a 256bit embedded key, 93% GAR and 97.5% "two try" GAR was shown. At those setting testing with a doppleganger dictionary of over 60,000 attacks per print, totalling over 550 Million impostor tests to date (and still counting), there have been no false accepts when using appropriate keys. Bipartite biotokens provide an extremely attractive secure template technology, ready for large scale use.

References

[1] A. Jain, K. Nandakumar and A. Nagar "Biometric Template Security," In *EURASIP Journal on Advances in Signal Processing*, vol. 2008, Article ID 579416.

[2] A. Juels and M. Sudan, "A Fuzzy Vault Scheme," In *Proc. of the IEEE Intl. Symposium on Information Theory*, 2002.

[3] K. Nandakumar, A. K. Jain and S. Pankanti, "Fingerprint-based Fuzzy Vault: Implementation and Performance," In *IEEE Trans. on Info. Forensics and Security*, vol. 2, no. 4, pp. 744-757, Dec. 2007.

[4] K. Nandakumar, A. Nagar and A. K. Jain", "Hardening Fingerprint Fuzzy Vault Using Password," In *2007 Int. Conf. on Biometrics*, LNCS 4642, 2007.

[5] K. Nandakumar and A. K. Jain, "Multibiometric Template Security Using Fuzzy Vault," In *IEEE Conf Biometric Theory, Application and Systems*, 2008.

[6] P. Mihailescu, "The Fuzzy Vault for Fingerprints is Vulnerable to Brute Force Attack, 2007. Online at http://arxiv.org/abs/0708.2974v1

[7] W. Chang, R. Shen and F. W. Teo, "Finding the Original Point Set Hidden Among Chaff, In *Proc. of the ACM Sym. on Info., Computer And Comm. Security*, 2006.

[8] Y. Dodis, L. Reyzin and A. Smith, "Fuzzy Extractors," In *Security with Noisy Data: Private Biometrics, Secure Key Storage and Anti-Counterfeiting*, P. Tuyls, B. Skoric and T. Kevenaar, Eds., chpt. 5, pp. 79-99, Springer-Verlag, 2007.

[9] X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky and A. Smith, "Secure Remote Authentication Using Biometrics," In *Proc. of EUROCRYPT*, Aarhus, Denmark, 2005.

[10] L. Ballard, S. Kamara and M. Reiter, "The Practical Subtleties of Biometric Key Generation," In *USENIX Security Symposium*, pp. 61-74, Aug. 2008.

[11] W. Scheirer and T. Boult, "Cracking Fuzzy Vaults and Biometric Encryption," In *Proc. of the 2007 IEEE Biometrics Symposium, held in conjunction with the Biometrics Consortium Conference (BCC 2007)*, Baltimore, MD.

[12] T. Boult, W. Scheirer and R. Woodworth, "Secure Revocable Finger Biotokens," In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2007

[13] W. Scheirer and T. Boult, "Bio-cryptographic Protocols With Bipartite Biotokens," In *Proc. of the IEEE 2008 Biometrics Symposium, held in conjunction with the Biometrics Consortium Conference (BCC 2008)*, Tampa, FL.

[14] D. Maio, D. Maltoni, J. Wayman, and A.K. Jain, "FVC2002: Second Fingerprint Verification Competition," In *Proc. of the 2002 Int. Conf. Pattern Recognition*