One-Time Biometric Token based Authentication

Rohan Kulkarni Anoop Namboodiri

International Institute of Information Technology
Hyderabad, India
{rohan.kulkarni@research, anoop}.iiit.ac.in

ABSTRACT

Widely used online commerce systems require an user to submit his sole banking credentials or credit card details for availing desired services, thus involving high risks with untrusted service providers. Often used one-time password based systems provide additional transaction security, but are still incapable of differentiating between a genuine user trying to authenticate or an adversary with stolen credentials. This brings out a strong need for biometrics based one-time password systems. In this paper we propose a one-time biometric token based authentication protocol which works within the framework of current online transaction schemes allowing an user to carry out a financial transaction with a service provider which completes with an authorization from the bank. The proposed protocol is based on key-binding biometric cryptosystems and upholds the requirements of secure authentication, template protection and revocability while providing privacy to individual's biometrics and anonymity from the service provider. We demonstrate our system's security and performance using iris biometrics to authenticate individuals.

Keywords

One-Time Passwords, Security, Biometric Authentication, Error Correcting Codes

1. INTRODUCTION

The idea of one-time passwords(OTPs) emerged to improve traditional password based authentication where an individual's password leakage directly compromises his system's security. The person can still change his password, but till then it could be too late. If financial accounts are involved then the loss of password for even a small period of time can cause huge losses to the individual. Often used techniques of smart-cards, pins and OTPs sent to the cellphone can be hijacked and individuals can be impersonated with a simple theft of device.

ICVGIP'14 December 14 - 18 2014, Bangalore, India

On the other hand, biometrics based systems provide a reliable solution for recognizing individuals. With the property of uniqueness that biometrics provide, the risk of permanently losing one's biometric trait exists. An adversary secretly capturing an instance of some biometric trait may permanently compromise that individual's identity based on that particular trait. To overcome such attacks, large number of multi-factor systems have been developed which use passwords, tokens or PINs along with biometric traits. In this work, we propose a protocol to develop one-time tokens with biometric traits for online bank transactions.

Biometric authentication systems are built on the premise that they must provide revocability, diversity and non-invertibility of underlying biometric templates. These properties ensure that if biometric templates are leaked, they can be updated again, other applications using similar systems are not affected by them and there is no loss of an individual's biometric privacy. In order to address these problems, cancelable biometric systems were introduced by Ratha *et al.*[17]. They proposed user specific distortion functions operating on cartesian, polar or functional basis. Also proposed in [19, 23, 15], such systems have to balance the trade-off of discriminability v/s non-invertibility [7].

Biometric cryptosystems [22] try and achieve security equivalent to cryptographic protocols. There are key-generation schemes which attempt to generate strong and stable keys from biometric traits as discussed in [4]. However, the intra-class variations in biometrics bring down the performance of such systems. The key-binding schemes [9] perform rather better and provide sufficient security and privacy. Fuzzy vault based schemes [8] create a secure vault by evaluating biometric features on a polynomial and storing it with chaff data. The authentication is based on successful recovery of the polynomial. Fuzzy commitment schemes use error correcting codes generated from random keys to mask the biometric templates. Biometric samples leading onto successful error correction are authenticated.

One-time passwords based on time stamps, discussed in [5] have been extended to biometrics to develop the concept of one-time biometric templates. To the best of our knowledge, they were introduced in [21] where they require a one-time-transform generating server. It communicates a common transaction based one-time-transform function to both the authentication server and the client and the protocol happens over several rounds of communication. They suggest invertible transforms to be applied so as to preserve the accuracy of the biometric match. Also, their one-time-transforms require an earlier biometric feature value. Any

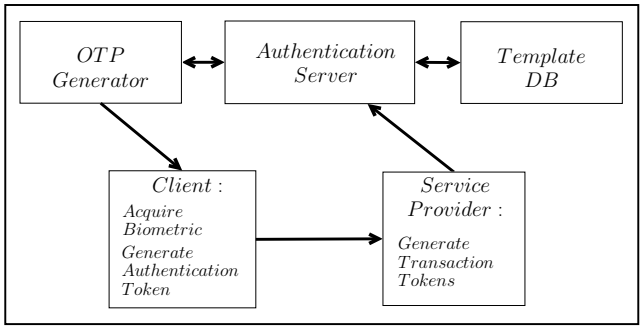

Figure 1: General One-Time Password Authentication

specific implementations have not been provided in their work.

One-time templates for face features have been provided by Lee *et al.*[14]. They propose repeatable transforms based on orthogonalizing and randomizing biometric features. Each authentication generates a new transform based on an user specific secret. The subsequent authentication attempts must update current transformation vectors present at the client and the server for a successful authentication. The matching accuracy is upheld even while authenticating in the transformed domain. Bringer *et al.*[2] propose anonymous time based authentication with cancelable biometrics. Their work does not rely on user specific keys for transforms, rather they assume the scanning hardware to be capable of generating one-time credentials. On every authentication attempt the scanner must communicate with the authentication server to compute time specific cancelable transformations. They propose certain properties of the distortions which prevent using one sensor's transforms onto another sensor, however implementation details of such transform functions is not provided. Secure, transaction based authentication using biotokens is also proposed in Scheirer *et al.*[20] but only for minutia based fingerprint matching.

Our proposed protocol is for fixed-length binary feature feature vectors matched using hamming distance. Our implementation is based on error correction codes, also used by Nandakumar *et al.*[16] for fingerprints, Kumar *et al.*[13] for palmprints, Kanade *et al.*[10] and Rathgeb *et al.*[18] for iris biometrics. However current protocols cannot be directly used for OTP based transaction authentication. In our proposed authentication scenario, the server holding the biometric credentials, on receiving a request, provides a random one-time key to the client. The client then computes an authentication token based on the received one-time key and its biometric information. The one-time biometric tokens are then transferred to the service provider to execute the transaction.

2. ONE-TIME PASSWORDS

We describe a general one-time password based transaction scheme in *Figure 1.* It comprises the following entities - *(i)* OTP Generator, *(ii)* Authentication Server, *(iii)* Template Database, *(iv)* Service Provider and *(v)* Client interface. The storage module and the one-time password generator can be together with the authentication module or exist as separate entities depending on the protocol implementation. When biometric authentication is included in the process, involved biometric data must be secured against adversarial attacks of eavesdropping, substitution and impersonation. One or multiple communication links can be attacked, hence the one-time passwords, biometric tokens and transaction data must be implicitly secure, without additional requirement of transport layer security(TLS) to ensure that no biometric information is leaked. Public certificates are generally issued only to the servers due to difficulties in their validation and clients' authentication is based only on their biometrics and tokens. During the authentication process, the server side, on receiving a request, provides a one-time key to the client. It is possible to serve this one-time password request without any biometric verification as the crucial part is the validation of final transaction. Generating completely random OTPs, independent of biometric templates also reduces computation and communication overhead and avoids biometric information leakage when they are being transferred to the client. Next, the client's biometric information and bank credentials need to be secured before providing them to the service provider. To uphold the privacy of client's bank credentials, transaction anonymity needs to be provided with respect to the service provider. Even during the final authentication at the server, details of the biometric data should not be revealed. We develop our protocol considering these aspects of one-time password based authentication.

3. ERROR CORRECTION

The feature vectors extracted from different biometric instances of the same individual certainly have some differences. We attempt to correct the differences of the query instance to exactly match to the stored template using a two layered error correction scheme. The first or the top layer handles random errors throughout the code and the second or the inner layer handles burst errors. This scheme is similar to the one proposed by Hao *et al.*[6] where they use Hadamard codes to correct random errors and Reed-Solomon codes to handle error bursts (localized errors). A pseudo random key of length $(K * b)$ *bits* is generated and encoded using Reed Solomon encoding to output a $(N * b)$ *bit* Reed-Solomon code. The obtained values are further encoded using the Hadamard linear error correction code, which is repeatedly applied over block length b generating (2^{b-1}) *bit* codes for each block. The final obtained code of $(N * 2^{b-1})$ *bits* is then XORed with the biometric vector to obtain the secure code. The Reed-Solomon encoding is based on the parameters b, K and N and satisfies the following condition:

$$K = N - 2T \tag{1}$$

Where, the input message consists of K blocks, the output code has N blocks, the error handling capacity of the code is T blocks and the block size is b *bits*.

The Hadamard encoding outputs a (2^{b-1}) *bit* code for a b *bit* message with the correction capacity of $(2^{b-2} - 1)$ *bits*.

While decoding, a modified code is received by the decoder. The obtained code is split into the blocks of (2^{b-1}) *bits*, decoded by finding the closest linear code using the Hadamard matrix which maps each block to a b *bit* code. The linear decoder performs most of the error correction. There exist several burst errors, where the complete linear coded block

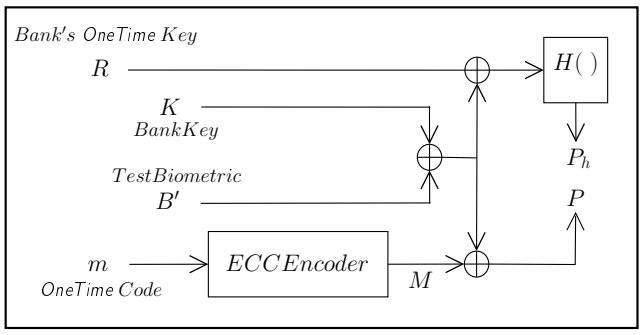

Figure 2: One-Time Biometric Token

may be erroneous. They cannot be corrected using the linear correction, the RS decoding in the second part handles them. If the localized errors are within the capacity of the RS code, the decoding will output the perfect original key. Large number of errors will output a different unrelated key.

4. AUTHENTICATION PROTOCOL

The protocol describes the steps to compute a one-time biometric token and use it to carry out a bank transaction with a service provider. The user is authenticated by the bank based on his biometric trait and an authentication key present in a smart-card issued to him, which also has his bank identity. The service provider is authenticated using its identity and an issued key. We expect the bank and the service provider to have public certificates signed by a trusted authority, as this has become a standard for all internet transactions today. As anywhere else, these certificates are used by our protocol to verify the authenticity of the party one is communicating with. They are not used to encrypt the one-time keys or biometric data of the user, however these are used by the protocol to bind the user's one-time transaction token with the specific service provider for enhanced security.

In our workflow, the user first verifies the service provider and his bank through their certificates. He then requests the bank to issue him a transaction ID and a one-time key. He then scans his biometric information, uses the smart-card key and a randomly generated code to create an authentication token. The bank's received one-time key is binded using the biometric, the smart-card key, the service providers certificate and the transaction information using a cryptographic hash. This generated pair of values along with the transaction ID is the one-time authentication token (*Figure 2*) of the client which is sent to the service provider. The service provider sends it to the bank along with his authentication credentials and the transaction information. The bank then verifies the validity of the transaction request to complete the authentication process. The protocol is described below in detail.

4.1 Registration Phase

The registration process takes place in a secure environment. An user $\mathcal{U}$ registers with the bank $\mathcal{BK}$ by providing its biometric B. The bank creates the user's identity U_{id} and token K and stores its biometric template as $B \oplus K$. The user's identity and token are communicated to the user through a smart-card. The service provider $\mathcal{SP}$ registers at the bank to acquire its identity SP_{id} and password SP_{key} . The bank's public certificate C contains its public key C_p and secret key C_s. Similarly, the service provider's certificate C' has its public key C'_p and secret key C'_s.

4.2 Authentication Phase

The steps during the authentication are described below.

INPUT:
$\mathcal{U}$: U_{id}, Biometric B', Token K.
$\mathcal{BK}$: Biometric template $B \oplus K$, Certificate C - Public Key C_p, Secret Key C_s.
$\mathcal{SP}$: SP_{id}, SP_{key}, Certificate C' - Public Key C'_p, Secret Key C'_s.
OUTPUT:
$\mathcal{BK}$ returning a *Success* or *Failure* after validating the transaction between $\mathcal{U}$ and $\mathcal{SP}$.
PROTOCOL:

1. $\mathcal{U}$ first verifies the SP and BK certificates C and C' to be signed by a trusted signing authority. It then requests $\mathcal{BK}$ for a one time key by sending his U_{id} encrypted with C_p.

2. $\mathcal{BK}$ identifies the U_{id} decrypting the request. It generates a transaction ID T_{id} and a random binary R of length equal to the stored template and maps them with the received U_{id}. The unique T_{id} and R are sent to $\mathcal{U}$ as the transaction key.

3. $\mathcal{U}$ receives T_{id} and R. It generates a random key m and computes an error correcting code M as per the scheme mentioned above. It extracts a feature vector B' from his biometric instance and XORs it with the token K along with M to get $P = B' \oplus K \oplus M$. P is the user's password for authentication. To generate the per transaction token, it computes a vector $B' \oplus K \oplus R$. It also concatenates C and T_{info} to it to bind this transaction to that particular $\mathcal{SP}$. It then computes $P_h = H(B' \oplus K \oplus R || C || T_{info})$ which acts as a verification hash for the transaction. It sends T_{id}, P and P_h to $\mathcal{SP}$.

4. $\mathcal{SP}$ receives T_{id}, P and P_h and sends the signed hash of the received value as a receipt. It verifies $\mathcal{BK}'s$ certificate and creates a request for a transaction, $Req = SP_{id} || H(SP_{key}) || T_{info}$. Encrypted request $E_{C_p}(Req)$ is sent to $\mathcal{BK}$ along with the T_{id}, P, P_h received from the user.

5. $\mathcal{BK}$ receives the request, identifies $\mathcal{SP}$. Authenticates it by matching stored SP_{id} and SP_{key} with the received ones. Next, it verifies the user data. Using the received T_{id}, BK obtains the corresponding U_{id}, the one-time key R and the biometric template $B \oplus K$ corresponding to the U_{id}. BK performs a XOR, $P \oplus (B \oplus K)$ i.e. $B' \oplus K \oplus M \oplus B \oplus K$, to obtain a modified error correcting code M'. It decodes M' using the decoding method mentioned earlier to get the key m'. If this decoding is unsuccessful BK aborts the transaction. A successful decoding does not mean a successful authentication as the decoding may have

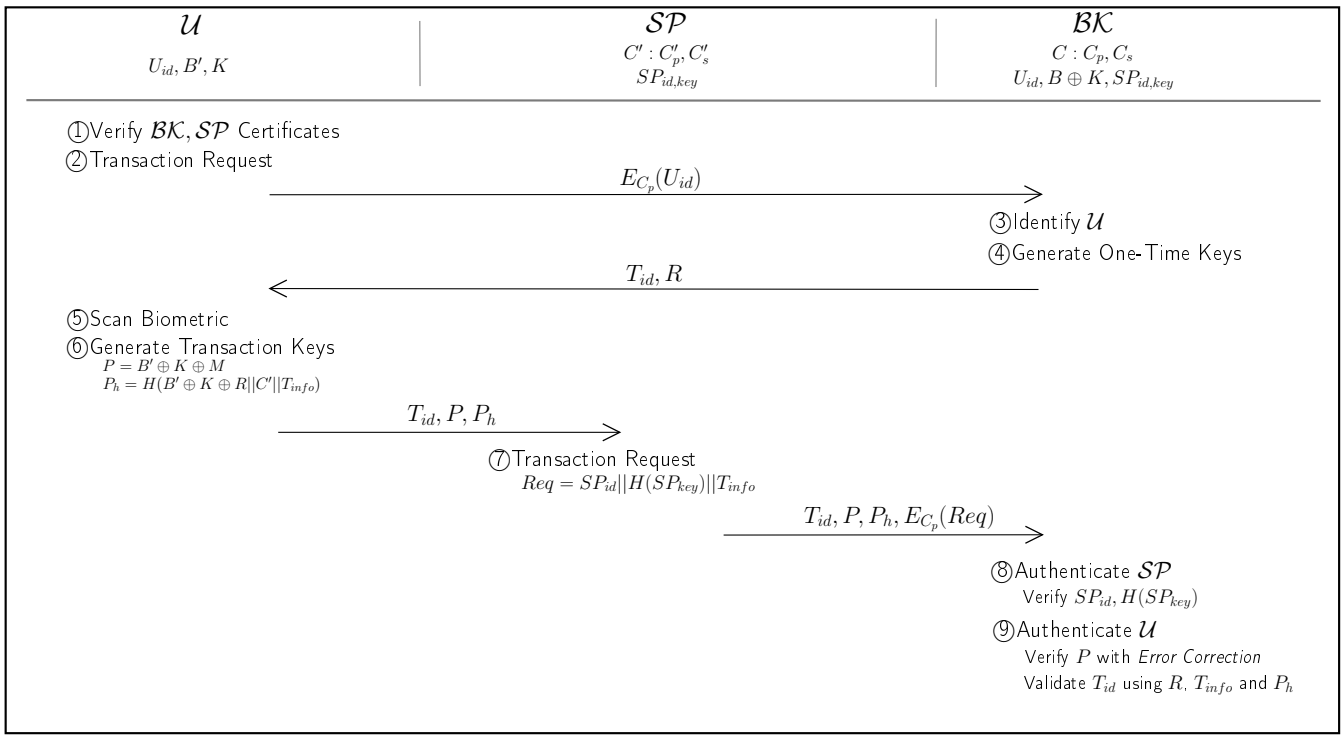

Figure 3: Proposed one-time biometric token based authentication protocol.

lead to extraction of a m' different from the users m due to the variations in B'. BK encodes the received m' again to generate a M''. It performs $M' \oplus M''$ to obtain the hamming distance vector D. This distance is the difference between the two biometric samples. It then verifies the hamming distance to be less than the matching threshold. If this fails BK aborts the transaction. For a successful authentication, the template, one-time R and P_h are verified as follows. The template $B \oplus K$ is XORed with R and the distance vector D to obtain $B_v = B \oplus K \oplus R \oplus D \sim B' \oplus K \oplus R$. The $SP's$ certificate and received T_{info}, along with B_v are matched with P_h.

$P_h \stackrel{?}{=} H(B_v||C||T_{info})$, if true the authentication is a *success* else a *failure*.

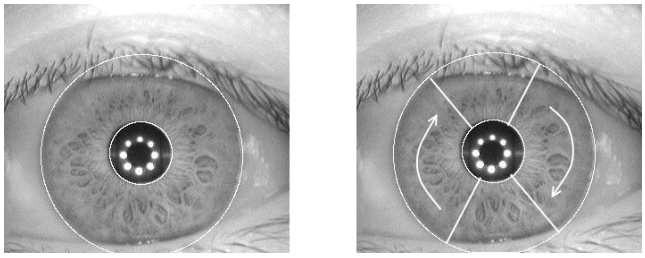

Figure 4: Iris segmentation and region of interest.

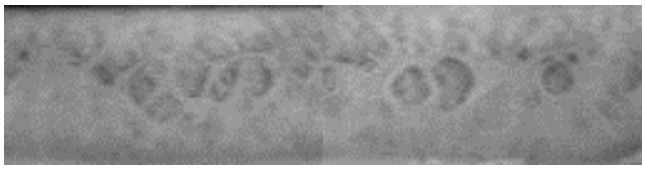

Figure 5: Polar transformation of iris region of interest.

5. EXPERIMENTS

The proposed protocol can be executed on top of any biometric trait based on binary feature vectors using hamming distance as the distance measure and compatible error correcting codes. We evaluate its matching performance on iris trait by extracting binary feature vectors from a subset of CASIA V3 iris dataset [1] consisting of 100 individuals with 8 samples each. Iriscodes are extracted based on the scheme presented by Ko *et al.*[12] and then binarized using a similar approach as Rathgeb *et al.*'s [18] to output a 2000 *bit* feature vector. The iris region is first segmented as described by Daugman *et al.*[3]. The segmented iris image is transformed to polar coordinates to obtain a 64×600 image. To remove the occlusion caused due to eye-lids and eye-lashes, only the part in the range [135° *to* 225°] and [315° *to* 45°] is selected as the region of interest as shown in *Figures 4* and *5*. Of the cropped 64×300 image the central 60×250 pixels are considered for feature extraction. The 60×250 image is divided into rectangular cell regions, each containing 3×10 pixels. The average value of the pixels in each cell are used to represent the cell. A cumulative sum based change analysis as discussed in [12] is performed on the obtained 20×30

Key Length (*bits*)	*b* (*bits*)	*FAR* (%)	*FRR* (%)
112	8	0.00	4.78
96	8	0.00	0.87
140	7	0.00	6.09
126	7	0.00	3.04
112	7	0.00	2.61
98	7	0.00	1.30

Table 1: Matching performance of the proposed protocol on CASIA V3 Iris evaluation subset at different key lengths. (*b* - RS code block size)

representative matrix to obtain a feature representation of length 1000. The values of the feature vector $\in \{0, 1, 2\}$. We then binarize each value into two bits, $0 \rightarrow 00$, $1 \rightarrow 01$ and $2 \rightarrow 10$. The binary vector is then rearranged in the form of two halves, first half consisting of the first bits of the *two bit* binary value and the second half consisting of the second bit. This places together the '1' bits in the binary feature. Such a grouping helps in error correction in our two-layer error correcting code. While computing a XOR operation, the burst errors occurred will result in lesser faulty blocks when rearranged, as compared to the initial binary representation. This will help the block level decoding function to maintain lower correction constraints in turn allowing for longer keys. Rathgeb *et al.*[18] have used a single RS-block encoding for error correction but we were unable to achieve high matching accuracy with that.

In the enrollment stage three iris images are pre-processed to extract corresponding feature vectors and a majority voted bit values are considered to create the template vector for each individual. The 2000 *bit* binary vector is extended to a 2048 *bit* vector by inserting 0 *bits* as we use a 2048 *bit* error correcting code. The templates are then XORed with individual specific keys of equal length. During authentication, only one iris sample is captured, pre-processed and binarized as discussed above to obtain the query vector. It is XORed with the individual's key and the one-time error correcting code. The decoding is verified as discussed in our protocol in *Section 4*. The matching accuracy of the system is evaluated at different key lengths of the error correcting code. The genuine comparisons are used to recover the encoding keys to confirm the match. The key length of the code is varied using both the RS code parameters and the Hadamard matrix size. The results obtained are mentioned in *Table 1* and *2* when evaluated on a systems with CPU speed of 2.1 GHz and 2 GB RAM. As the key size is increased, the error correction capacity of the code reduces, increasing the FRR of the system. Increased template entropy due to user specific keys prevents any False Accepts. This system can also be directly used with face biometrics adopting the face matching technique of Kanade *et al.*[11].

6. PROTOCOL ANALYSIS

The computation costs of the protocol are described in *Table 2*. The protocol does not involve any homomorphic operations or secure two-party computations, thus keeping the computation costs low. We analyze in detail the privacy and security offered by the protocol along with its correctness below.

	Time (in *ms*)
Client	8
Service Provider	4
Server	70

Table 2: Computation costs of the proposed protocol.

6.1 Correctness

In the execution of the protocol, tasks of validating user's biometric one-time token and service provider's credentials are performed at the bank. The verification of the service provider is a straight forward matching of keys. The credentials of the client are the key encoded biometric feature P, which validates biometric sample, and the hash value P_h , which validates the one-time-ness of the authentication. The decoding capacity of the code is comparable to the system threshold. If a query sample provided by some user consists of errors greater than the correcting capacity, it results into a decoding failure or a generation of an incorrect key. Re-encoding the generated incorrect key with the same error correction scheme will certainly not output a code which matches with the earlier obtained code within the limits of the threshold. So, if a valid decoding takes place, the hash P_h will be exactly matched in the verification. Thus a query which would have been rejected by the biometric system without template protection will surely be rejected by our system.

6.2 Privacy Analysis

The protocol ensures user anonymity from the service provider. The user does not provide any of his bank credentials to the service provider directly, only binded with the one-time keys. Consolidating the data over several users' transactions, the service provider can XOR the tokens to obtain differences between them. Identifying tokens which are from same individual from the rest is not feasible as the keys used are strong(*Table 1*).

6.3 Attacks and Security Analysis

A dictionary attack is among the easiest of the attacks attempted, where an adversary can try out some database of iris images against the stored templates to obtain illegitimate authentication. In our protocol the templates are XORed with user specific keys that defend such an attack. XORing with a key raises the difficulty of an dictionary attack to a brute force attack of the order of template size. The key used, in a way, randomizes the underlying biometric template and increases the inter class entropy while maintaining intra-class variations.

6.3.1 Client Security

The protocol ensures biometric security from an illegitimate user by ensuring that the client does not receive any biometric information during the authentication process. One-time transaction ID and random one-time key are communicated to it, only providing additional security from replay attacks.

6.3.2 Server Security

The biometric templates stored at the server are XORed with user specific keys hiding the underlying biometric feature vector. A breach of the server's database will only reveal the secured biometric templates to the adversary. For different banks using the same authentication system, the user keys will be different. So even a breach of multiple databases will not enhance the chances of leaking of original biometric information. If an adversary is present at the server during the authentication process, he has access to both the template and the query vectors. The intermediate information revealed consists of the original error correcting code used by the client, the distance vector and the matching score. The distance vector just reveals the indices where the query and probe samples matched or differed. Thus the adversary cannot obtain the original bits in the biometric feature vector using any of the intermediate values.

6.3.3 Network Security

An adversary may try to break in the network and sniff the data transferred or try and impersonate the bank or the service provider to get hold of user credentials. Impersonation attacks are defended at the step of certificate validation. Certificates issued only by a standard authority are trusted. Transaction information that can be picked up from the network are the random key sent by the bank(R) and the user's authentication token(P, P_h). The service provider's credentials and transaction data is sent to the bank through a public key encryption. Brute force attacks on P are not feasible due to strong encoding keys involved. If multiple authentication instances can be collected by an adversary, they would be encoded using different encoding keys, resulting into a different error correction code. If we consider the underlying biometric template to be exactly the same, on XORing the two authentication instances will reveal the locations at which the two codes differ. This can be extended to predict the locations at which the encoding keys differ, however no bit of the underlying key is revealed. Intra-class variations in the biometric samples provide sufficient entropy to mask direct leakage of such information.

In the worst case, the adversary steals an individual's smart-card key, the transaction tokens and one-time keys sent by the bank, the scenario shifts to breaking a key-binding scheme involving the iris feature vector. The security of the key-binded iriscode depends directly on the length of the encoding key, but the iriscode involved is not equivalent to a completely random vector. As discussed in the implementation section, it consists of 200 groups with 5 values per group and the values $\in \{0, 1, 2\}$. The 1's and 2's are continuous inside the group and the rest of the values are 0. These consecutive 1's or 2's can start and end at any index from 1 to 5. Thus the allowed combination of values per group is 20. In general, a completely random 5 *bit* key provides security equivalent to 5 *bits* as all permutations(32) of 0's and 1's are permitted. Then, in our case, the effective security of the 5 values of the iriscode provide security equivalent to 3.12 *bits*. The 5 values, when binarized, contribute 10 *bits* to the iriscode. Thus, 10 *bits* of an iriscode provide effective security equivalent to atleast a 3 *bit* random key. Considering that the iriscode is XORed with an error correcting code, we will have to consider its error correction capacity to effectively compute the security provided by the iriscode. Let the feature vector be N *bits* and the error correction capacity of the code be θ *bits*. Then, for an adversary, to successfully decode the key-binding it has to exactly predict the $(N - \theta)$ *bits* of the iriscode. From the values discussed above, a $(N - \theta)$ *bit* iriscode will effectively provide security equivalent to a $3 * (N - \theta)/10$ *bits*. The error correction capacity of the code in turn depends directly on the encoding key length. Shorter the key involved, larger number of corrections are possible for a fixed code length, thus lower the security. Considering the key-lengths provided in *Table 1*, a 96 *bit* key will output a code with a correction capacity of 690 *bits*. Then, the effective security provided by the 2000 bit iriscode is 493 *bits* which is equivalent of saying that the adversary has to exactly predict a 493 bit string to break the key-binding. Considering the computation required for that task a brute force attack is unfeasible.

7. CONCLUSION

In this work, we have proposed a one-time biometric token based authentication scheme. This protocol has been detailed in a scenario of an online bank transaction, where the user generates a one-time biometric token based on a random one-time key received from the bank. It provides a strong alternative to the currently used methods where user's sole bank credentials or card details need to be provided to service providers. Proposed way of transaction upholds user's biometric privacy and provides anonymity while dealing with any service provider. The bank does not transfer any biometric related information to the user during the execution of the protocol, it follows currently used methods of just sending a random key and mapping it with the user and the transaction ID. Unlike previous protocols it does not require updating biometric templates or keys after each transaction and works within the communication framework of current online banking systems. Performance and security of the system has been analyzed and it meets desired standards.

8. REFERENCES

[1] Casia iris dataset. http://biometrics.idealtest.org/dbDetailForUser.do?id=3.

[2] J. Bringer, H. Chabanne, and B. Kindarji. Anonymous identification with cancelable biometrics. In *Image and Signal Processing and Analysis, Proceedings of 6th International Symposium on*, pages 494–499. IEEE, 2009.

[3] J. Daugman. How iris recognition works. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):21–30, 2004.

[4] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Advances in cryptology-Eurocrypt*, pages 523–540. Springer, 2004.

[5] N. Haller, C. Metz, P. Nesser, and M. Straw. A one-time password system. Technical report, RFC 1938, May, 1996.

[6] F. Hao, R. Anderson, and J. Daugman. Combining crypto with biometrics effectively. *Computers, IEEE Transactions on*, 55(9):1081–1088, 2006.

[7] A. K. Jain, K. Nandakumar, and A. Nagar. Biometric template security. *EURASIP Journal on Advances in Signal Processing*, page 113, 2008.

[8] A. Juels and M. Sudan. A fuzzy vault scheme. *Designs, Codes and Cryptography*, 38(2):237–257, 2006.

[9] A. Juels and M. Wattenberg. A fuzzy commitment scheme. In *Proceedings of the 6th ACM conference on Computer and communications security*, pages 28–36. ACM, 1999.

[10] S. Kanade, D. Petrovska-Delacrétaz, and B. Dorizzi. Generating and sharing biometrics based session keys for secure cryptographic applications. In *Biometrics: Theory Applications and Systems (BTAS), Fourth IEEE International Conference on*, pages 1–7. IEEE, 2010.

[11] S. G. Kanade, D. Petrovska-Delacretaz, and B. Dorizzi. A novel crypto-biometric scheme for establishing secure communication sessions between two clients. In *Biometrics Special Interest Group (BIOSIG), Proceedings of the International Conference of the*, pages 1–6. IEEE, 2012.

[12] J.-G. Ko, Y.-H. Gil, and J.-H. Yoo. Iris recognition using cumulative sum based change analysis. In *Intelligent Signal Processing and Communications, ISPACS'06. International Symposium on*, pages 275–278. IEEE, 2006.

[13] A. Kumar, A. Kumar, and S. Schuckers. Development of a new cryptographic construct using palmprint-based fuzzy vault. *EURASIP Journal on Advances in Signal Processing*, 2009(13), 2009.

[14] Y. Lee, Y. Lee, Y. Chung, and K. Moon. One-time templates for face authentication. In *Convergence Information Technology, International Conference on*, pages 1818–1823. IEEE, 2007.

[15] A. Lumini and L. Nanni. An improved biohashing for human authentication. *Pattern Recognition*, 40(3):1057–1065, 2007.

[16] K. Nandakumar. Biosake: Biometrics-based secure authentication and key exchange. In *Biometrics (ICB), International Conference on*, pages 1–8. IEEE, 2013.

[17] N. K. Ratha, S. Chikkerur, J. H. Connell, and R. M. Bolle. Generating cancelable fingerprint templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(4):561–572, 2007.

[18] C. Rathgeb and A. Uhl. Systematic construction of iris-based fuzzy commitment schemes. In *Advances in Biometrics*, pages 940–949. Springer, 2009.

[19] M. Savvides, B. Vijaya Kumar, and P. K. Khosla. Cancelable biometric filters for face recognition. In *Pattern Recognition, Proceedings of the 17th International Conference on*, volume 3, pages 922–925. IEEE, 2004.

[20] W. J. Scheirer and T. E. Boult. Bipartite biotokens: Definition, implementation, and analysis. In *Advances in Biometrics*, pages 775–785. Springer, 2009.

[21] Y. Ueshige and K. Sakurai. A proposal of one-time biometric authentication. In *Security and Management*, pages 78–83, 2006.

[22] U. Uludag, S. Pankanti, S. Prabhakar, and A. K. Jain. Biometric cryptosystems: issues and challenges. *Proceedings of the IEEE*, 92(6):948–960, 2004.

[23] J. Zuo, N. K. Ratha, and J. H. Connell. Cancelable iris biometric. In *Pattern Recognition, 19th International Conference on*, pages 1–4. IEEE, 2008.