Secure Biometric Crypto System

Mrs.S.Karthigaiveni† ,R.Swathiramya†† ,U.R.V.Nandhiny†††, S.Sivaranjani Nachiyar††††,S.Jenila†††††

†Department of information technology,Periyar Maniammai University,Thanjavur,Tamil Nadu,India
PG Student ,Periyar Maniammai University,Thanjavur,Tamilnadu,India

Summary
Reliable user authentication is becoming an increasingly important task in the Web-enabled world . Biometrics-based authentication systems offer obvious usability advantages over traditional password and token-based authentication schemes. However, biometrics also raises some issues in lack of privacy,template security,revocability.The use of cryptographic primitives to bolster the biometric authentication system can solve the issues in biometric system..The combination of biometrics over cryptography may lead to a problem of lack of accuracy in biometric verification. In this paper, We propose a cryptographic protocol for biometrics authentication without revealing personal biometrical data against malicious verifier the protocol is termed as blind biometric authentication protocol, which addresses the concerns of user's privacy, template protection,trust issue.The accuracy problem can be solved by designing a classifier. The protocol is blind in the sense that it reveals only the identity, and no additional information about the user or the biometric to the authenticating server or vice-versa. The proposed protocol is secure to different attacks.

Key words:
biometrics, cryptosystems, privacy, public key cryptography, security, authentication

1. Introduction

Today's human authentication factors have been placed in three categories, namely What you know, e.g password, secret, personal identification number (PIN); What you have, such as token, smart card etc. and What you are, biometrics for example. However, the first two factors can be easily fooled. For instance, password and PINs can be shared among users of a system or resource. Moreover, password and PINs can be illicitly acquired by direct observation. The main advantage of biometrics is that it bases recognition on an intrinsic aspect of a human being and the usage of biometrics requires the person to be authenticated to be physically present at the point of the authentication. These characteristics overcome the problems whereas password and token are unable to differentiate between the legitimate user and an attacker. In addition biometric authentication information cannot be transferred or shared; it is a powerful weapon against repudiation. However, it also suffers from some inherent biometrics-specific threats [1]. A hacker who gains physical or remote access to an authentication server can steal the stored templates, which are non replaceable in case of plain templates. Concerns are also on the privacy as many biometrics reveal personal information beyond just identity. Widespread use of biometric authentication also provides the ability to track a person through every activity in his life, which introduces another significant privacy concern. The primary concerns in widespread use of biometrics for remote and onsite authentication are in i) template protection, ii) privacy of the user, iii) trust between user and server, and iv) network security. The ideal solution to overcoming all the privacy and security concerns would be to apply a strong encryption on the biometric samples as well as the classifier parameters, and carry out all the computations in the encrypted domain. However, the primary goal of a strong encryption algorithm is to destroy any pattern that would be present in the data. We now need to carry out a pattern classification task (identity verification) in the encrypted domain. These two goals are contradictory. In other words, security/privacy and accuracy seems to be opposing objectives. Different secure authentication solutions achieve their goal through a compromise between privacy and accuracy or by making restrictive assumptions on the biometric data. The primary difference in our approach is that we are able to design the classifier in the plain feature space, which allows us to maintain the performance of the biometric itself, while carrying out the authentication on data with strong encryption, which provides high security/privacy. However, such a solution would require an *algebraic homomorphic encryption* scheme [2]. The only known doubly homomorphic scheme has recently been proposed by Gentry [3] and would mostly lead to a computationally intensive theoretical solution. We show that it is possible to achieve a practical solution using distribution of work between the client (sensor) and the server (authenticator), using our proposed randomization scheme .

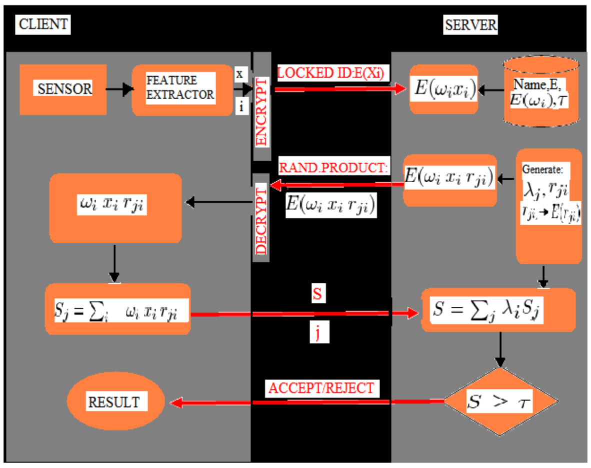
2. Blind Authentication

We define *Blind Authentication* as "a biometric authentication protocol that does not reveal any information about the biometric samples to the authenticating server. It also does not reveal any

Manuscript received January 5, 2012
Manuscript revised January 20, 2012

information regarding the classifier, employed by the server, to the user or client." Blind authentication, proposed in our paper, is able to achieve both strong encryption-based security as well as accuracy of a powerful classifier. While the proposed approach has similarities to the blind vision [4] scheme for image retrieval, it is far more efficient for the verification task.
Blind Authentication addresses all the concerns mentioned before
1) The ability to use strong encryption addresses template protection issues as well as privacy concerns.
2) Non-repudiable authentication can be carried out even between nontrusting client and server using a trusted third party solution.
3) It provides provable protection against replay and clientside attacks even if the keys of the user are compromised.
4) As the enrolled templates are encrypted using a key, one can replace any compromised template, providing revocability, while allaying concerns of being tracked.
In addition, the framework is generic in the sense that it can classify any feature vector, making it applicable to multiple biometrics. Moreover, as the authentication process requires someone to send an encrypted version of the biometric, the nonrepudiable nature of the authentication is fully preserved, assuming that spoof attacks are prevented.

We assume that authentication is done through a generic *linear classifier*. One could use any biometric in this framework as long as each test sample is represented using a feature vector of length n . Note that even for biometrics such as fingerprints, one can define fixed length feature representations [5]. Let ω be the parameters of the linear classifier (perceptron). The server accepts the claimed identity of a user, if , ω.x < τ where τ is a threshold. As we do not want to reveal the template feature vector (ω) or the test sample (x) to the server, we need to carry out the perceptron function computation directly in the encrypted domain. Computing ω.x involves both multiplication and addition operations, thus computing it in the encrypted domain requires the usage of a doubly homomorphic encryption scheme [6]. In the absence of a practical doubly homomorphic encryption scheme (both additive and multiplicative homomorphic), our protocol uses a class of encryption that are multiplicative homomorphic, and we simulate addition using a clever randomization scheme over one-round of interaction between the server and the client. An encryption scheme E(x) is said to be multiplicative homomorphic, if E(x).E(y)=E(xy) for any two numbers x and y . We use the popular MD5 encryption scheme , which satisfies this property.


CLIENT
SERVER
SENSOR
FEATURE EXTRACTOR
x
i
ENCRYPT
LOCKED ID:E(Xi)
$E(\omega_i x_i)$
Name,E, $E(\omega_i)$,τ
RAND.PRODUCT:
$E(\omega_i\, x_i\, r_{ji})$
Generate: λ_j, r_{ji} $r_{ji} \rightarrow E(r_{ji})$
$\omega_i\, x_i\, r_{ji}$
DECRYPT
$E(\omega_i\, x_i\, r_{ji})$
$S_j = \sum_i \omega_i\, x_i\, r_{ji}$
S
j
$S = \sum_j \lambda_i S_j$
RESULT
ACCEPT/REJECT
$S > \tau$
Fig 1. Blind authentication process.

An overview of the authentication process is presented in Fig. 1. We assume that the server has the parameter vector ω in the encrypted form, i.e E(ω) , which it receives during the enrollment phase. The authentication happens over two rounds of communication between the client and the server. To perform authentication, the client locks the biometric test sample using her public key and sends the *locked ID* to the server. The server computes the products of the locked ID with the locked classifier parameters and randomizes the results. These *randomized products* are sent back to the client. During the second round, the client unlocks the randomized results and computes the sum of the products. The resulting *randomized sum* is sent to the server. The server derandomizes the sum to obtain the final result, which is compared with a threshold for authentication. As we described before, both the user (or client) and the server do not trust each other with the biometric and the claimed identity. While the enrollment is done by a trusted third party, the authentications can be done between the client and the server directly. The client has a biometric sensor and some amount of computing power. The client also possesses an MD5 private– public key pair, and . We will now describe the authentication and enrollment protocols in detail.

2.1. Authentication

We note that the computation of requires a set of scalar multiplications, followed by a set of additions. As the encryption used is homomorphic to multiplication, we can compute, $E(\omega_i x_i) = E(\omega_i)E(x_i)$, at the server side. However, we cannot add the results to compute the authentication function. Unfortunately, sending the products to the client for addition will reveal the classifier parameters to the user, which is not desirable. We use a clever randomization mechanism that achieves this computation without revealing any information to the user. The randomization makes sure that the client can do the summation, while not being able to decipher any information from the products. The randomization is done in such a way that the server can compute the final sum to be compared with the threshold. The overall algorithm of the authentication process is given in Algorithm 1. Note that all the arithmetic operations that we mention in the encrypted domain will be -operations, i.e., all the computations such as *(a op b)* will be done as *(a op b) mod q* , where q is defined by the encryption scheme employed.

```
Algorithm 1: Authentication
1: Client computes feature vector, $x_{1,\dots,n}$ , from test data
2: Each feature $x_i$ is encrypted ($E(x_i)$) and sent to server
3: Server computes kn+k random numbers, $r_{ji}$ and $\lambda_j$ , such that, $\forall_i$ , $\sum_{j=1}^{k} \lambda_j r_{ji} = 1$
4: Server computes $E(\omega_{ji} x_i r_{ji}) = E(\omega_{ji}) E(x_i) E(r_{ji})$
5: The kn products thus generated are sent to the client
6: The client decrypts the products to obtain: $\omega_{ji} x_i r_{ji}$
7: Client returns $S_j = \sum_{i=1}^{n} \omega_{ji} x_i r_{ji}$ to the server
8: Server computes  S= $\sum_{j=1}^{k} \lambda_j S_j$
9: if  S > τ then
10:        return Accepted to the client
11:    else
12:        return Rejected to the client
13:  end if
```

In the algorithm, the server carries out all its computation in the encrypted domain, and hence does not get any information about the biometric data(x) or the classifier parameters(ω) . A malicious client also cannot guess the classifier parameters from the products returned as they are randomized by multiplication with r_{ji}. The reason why the server is able to compute the final sum S in Step 8 of Algorithm 1 is because we impose the following condition on r_{ji}s and λ_j s during its generation:

$$\forall_i, \sum_{j=1}^{k} \lambda_j r_{ji} = 1 \qquad (1)$$

The authentication process thus maintains a clear separation of information between the client and the server and hence provides complete privacy to the user, and security to the biometric. Moreover, the clear biometric or parameters are never stored at any place, thus avoiding serious losses if the server or the client computer is compromised. We now look at the enrollment phase of the protocol.As in this heading, they should be Times 12-point boldface, initially capitalized, flush left, with one blank line before, and one after. Use Heading 2 style in the template.

2.2Enrollment

During the enrollment, the client sends samples of her biometric to the enrollment server, who trains a classifier for the user. The trained parameters are encrypted and sent to the authentication server, and a notification is sent back to the client. Fig. 2 gives an overview of the enrollment process. The biometric samples sent by the client to the enrollment server could be digitally signed by the client and encrypted using the servers public key to protect it. The use of a third party for enrollment also allows for longterm learning by the enrollment server over a large number of enrollments, thus improving the quality of the trained classifier. Algorithm 2 gives a step-by-step description of the enrollment process. Note that the only information that is passed from the enrollment server to the authentication server is the users identity, her public key, the encrypted versions of the parameters, and a threshold value.

```
Algorithm 2: Enrollment
1: Client collects multiple sample of her biometric, $B_{1,\dots,k}$
2: Feature vectors, $x_i$, are computed from each sample
3: Client sends $x_i$, along with her identity and public keyE, to the enrollment server
4: Enrollment server uses $x_i$ and the information from other users to compute an authenticating classifier $(\omega, \tau)$ for the user
5: The classifier parameters are encrypted using the users public key: $E(\omega_i)$
6: $E(\omega_i)$s along with the user's identity, the encryption key (E) , and the threshold (τ) , are sent to the authentication server for registration
7: The client is then notified about success.
```

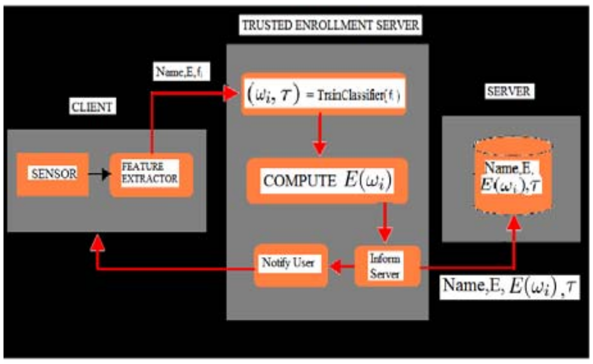

TRUSTED ENROLLMENT SERVER
Name,E,f
CLIENT
(ω_i, τ) = TrainClassifier(f)
SERVER
SENSOR
FEATURE EXTRACTOR
COMPUTE $E(\omega_i)$
Name,E, $E(\omega_i)$,τ
Notify User
Inform Server
Name,E, $E(\omega_i)$,τ

Fig 2. Enrollment based on a trusted third party (TTP)

3. Security,Privacy and Trust in Blind Authentication

Security of the system refers to the ability of the system to withstand attacks from outside to gain illegal access or deny access to legitimate users. Since we are dealing with insecure networks, we are primarily concerned with the former . In terms of information revealed, security is related to the amount of information that is revealed to an attacker that would enable him to gain illegal access. Privacy on the other hand is related to the amount of user information that is revealed to the server. Ideally, one would like to reveal only the identity and no additional information. Most of the current systems provide very little privacy, and hence demands trust between the user and the server. An ideal biometric system would ensure privacy and hence need not demand any trust, thus making it applicable in a large set of applications.We now take a closer look at the security and privacy aspects of the proposed system.

3.1System Security

Biometric systems are known to be more secure as compared to passwords or tokens, as they are difficult to reproduce. As the authentication process in the proposed system is directly based on biometrics we gain all the advantages of a generic biometric system. The security is further enhanced by the fact that an attacker needs to get access to both the user's biometric as well as her private key to be able to pose as an enrolled user.

1)Server Security: We analyze the security at the server end using two possible attacks on the server.

Case 1: Hacker gains access to the template database. In this case, all the templates (or classifier parameters) in the server are encrypted using the public key of the respective clients. Hence gaining access to each template is as hard as cracking the public key encryption algorithm. Moreover, if by any chance a template is suspected to be broken, one could create another one from a new public–private key pair. As the encryption's are different, the templates would also be different. Brute-force cracking is practically impossible if one uses a probabilistic encryption scheme, even for limited-range data.

Case 2: Hacker is in the database server during *the authentication.* In such a situation, the hacker can try to extract information from his entire "view" of the protocol. Specifically, the view consists of the following five components:

1) encrypted values of all ω_is, that is $(\omega_i), i \in [1,n]$;

2) encrypted values of all x_is, that is $E(x_i), i \in [1,n]$;

3) all the random values used in the protocol, that is all the η_{ji} s, $i \in [1,n]$ and $j \in [1,k]$;

4) all the $\lambda_j s, j \in [1,k]$; and

5)all intermediate sum $S_j = (\sum_{i=1}^{n} \omega_i\ x_i\ \eta_{ji})\%N$ for all $j \in [1,k]$.

We ask, what can the hacker learn about the critical data, viz $\omega_i s$ and $x_i\ s$. Note that the hacker only obtains *k* linear congruences over the *n* variables , $y_1, y_2, \dots\dots, y_n$, namely $S_j = (\sum_{i=1}^{n} \eta_{ji}\ y_i)\%N$ for all $j \in [1,k]$, where $y_i = \omega_i x_i$. Even though this may reveal some information about $y_i s$, it is impossible to recover the original biometric, as it requires $|Y|^{n-k}$ authentication trials ($|Y|$ is domain of y_is),each involving the help of the client and his private key. The effort required to guessing the original biometric is high and it is difficult.

Case 2.1: If the hacker is in the server over multiple authentication trials of the same user. Then he will have multiple sets of k linear congruences to infer the values of y_i. However note that the value of x_i will change slightly slightly over multiple authentications, which gets reflected in the values of y_i . Now the hacker's problem is to compute an approximate estimate of y_i from his view of congruences over noisy y_i s, which we cal l y_i. Let $\varepsilon_i \in$ **E** be the noise between the two instances of x_i. From linear algebra, we know that every additional set of *k* linear congruences will reduce the brute-force attack complexity by $O|Y|^k$. Thus, it seems like after a certain number of authentication trials, a hacker will have sufficient congruences to uniquely solve for *n* the variables. However, we now show that even this is not possible, as during each authentication trial, the hacker not just obtains *k* additional equations but also ends up adding new *n* variables. To ensure complete privacy, one has to make sure that the information gained by the additional *k* equations is less than the uncertainty introduced by the new *n* variables. Our scheme assumes that the server runs the delegated code faithfully. If the server is malicious, it can try to learn additional information about the client's biometric by using a selected vector (say unit vector in a direction) instead of the template for the product. However, the client can detect this using an input, whose result is known. For example, the client can randomly send a vector, which is known to be authentic (not authentic), and check if the the server accepts (rejects) it. Another option would be to use a probabilistic encryption scheme for the template, and keep the randomness in the encryption, a secret, as the server never needs to decrypt any data. In this case, the server will not be able to use any data other than the temple provided for computations.

*Case 3: Impostor trying blind attacks from a remote machine.*It is clear that a brute force attack will have a complexity of the product of that of the plain biometric and the private key. However, note that in the final step, the computed confidence score S is a linear combination, and is compared with a threshold. Hence, if the impostor replaces the partial sums S_j with random numbers, he might be able to pass the confidence test without knowing anything about the biometric or the private key. Also note that the probability of success in this case could be very high. However, a simple modification of the protocol at the server side could thwart this attack. The server could multiply all the sums with a random scale factor *sf* and check if the returned sum is a multiple *sf* of or not. From his view, the impostor cannot learn *sf as* GCD is not defined for congruences. In short, we see that the server is secure against any active or passive attack, and will not reveal any information about the classifier or the user's biometric.

2)Client Security: At the client side,we will consider the fallowing attack scenarios .

Case 4: Hacker gains access to the user's biometric or private key. Our protocol captures the advantages of both the biometric authentication as well as the security of the PKC. If the attacker gets hold of the user's biometric from external sources, he would also need the private key of the user to be able to use it. If only the private key of a user is revealed, the security for the effected individual falls back to that of using the plain biometric. Note that in practice, the private key is secured by storing it in a smart card, or in the computer using a fuzzy vault. In short, an impostor needs to gain access to both the private key and the biometric to pose as a user. Even in this case, only a single user will be affected, and replacing the lost key would prevent any further damages. In practice, periodic replacement of the private key is advisable as in any PKC-based system.

Case 5: Passive attack at the user's computer. In this case, the hacker is present in the user's computer during the login process. As the private key can be secured in a hardware which performs the encryption, the hacker will not have direct access to the private key. In other words, he will only learn the intermediate values of the computations. The hackers view will consist *kn* of quadratic congruences: $y_j r_{ji}, i \in [1,n], j \in [1,k]$. He further knows that there exists $k\lambda_i$ s that satisfy *n* congruences: $\sum_j \lambda_j r_{ji} \% N = 1$. Thus he has *kn+n* quadratic congruences in *kn+n+k* variables. This, as in case 2, results in an effort equivalent to a brute force attack. However if the hacker can stay in the user's computer over multiple authentications, then at some point of time, he will have a sufficient number of congruences to solve for y_is (see case 2). Note that y_i s does not reveal any useful information about the classifier. Moreover, any partial information gained is of no use as an authentication cannot be performed without access to the private key. Note that an active attack in this case is identical to that of case 3, and the hacker does not know the private key.

3)Network Security:An insecure network is susceptible to snooping attacks .Let us consider the fallowing attack scenarios .

Case 6: Attacker gains access to the network. An attacker who may have control over the insecure network can watch the traffic on the network, as well as modify it. The confidentiality of the data flow over the network can be ensured using the standard cryptographic methods like symmetric ciphers and digital signatures. Furthermore, all the traffic on the network are encrypted either using the clients public key or using the random numbers generated by the server. Hence, even if successfully snooped upon, the attacker will not be able to decipher any information. A replay attack is also not possible as the data communicated during the second round of communication is dependent on the random numbers generated by the server.

3.2 Privacy

Privacy, as noted before, deals with the amount of user information that is revealed to the server during the process of enrollment and authentication. We noted that there are two aspects of privacy to be dealt with:

1. *Concern of revealing personal information:* As the template or test biometric sample is never revealed to the server, the user need not worry that the use of biometrics might divulge any personal information other than her identity.
2. *Concern of being tracked:* One can use different keys for different applications (servers) and hence avoid being tracked across uses. In fact, even the choice biometric or real identity of the user itself is known only to the enrolling server. The authenticating server knows only the user ID communicated by the enrollment server and the biometric is obtained in the form of an encrypted feature vector.As the user and server need not trust each other, the framework is applicable to a variety of remote and on-site identity verification tasks. Moreover, we note that there is no delegation of trust by the server to a program or hardware at the user's end, thus making it applicable to a variety of usage scenarios.

3.3 Biometric Verification

We use skeleton method to verification which is much suitable for our protocol. The primary limitation of the protocol in its current form is its restriction to fixed length feature vector representation of the data . However, we note that there are techniques that employ fixed length representations of various biometrics with performances that are comparable to those using variable length representations. Some of the well-known matching techniques using variable length features are graph-based local structure representations of minutiae by Kisel *et al.* [7], Timeseries representations of hand geometry by Vit *et al.* [8], and Face Bunch Graph representations of face by Wiskott *et al.* [9]. Comparable accuracies have been reported using fixed length representation such as the invariant moment features for fingerprints by Yang *et al.* [10], the hand geometry features by David *et al.* [11], the 3-D morphable face model by Blanz *et al.* [12], and the DCT coefficient representation of the Iris by Monro *et al.* [13], all achieve performances close to the state of the art in the respective biometrics.

3.4Computation and Communication Overheads

The additional computation that needs to be carried out can be divided into two parts: 1) Modulo multiplications to be done for encryption/decryption and inner product, and 2) the additional time spent in the computation of random numbers, products, and sums. As the modulo multiplications and encryption decryption operations can be done efficiently using dedicated hardware available [14], we analyze the time required for both, separately. Consider a biometric with feature vector of length . In the protocol, the client needs to do encryptions for the test vector . For the linear classifier, the server needs to do *kn* encryptions of the random numbers and 2 *kn* multiplications, so as to compute $E(\omega_i x_i r_{ji})$, where $k \leq n$. The client needs to do *kn* decryptions. Additional computations at the server includes *n+kn* modulo multiplications of encrypted numbers at the server end *kn*, and nonencrypted additions at the client end. In addition, the server generates *kn* random numbers. For most practical biometrics, the total run time required for all these (nonencrypted) computations together on current desktop machines is less than10 ms. The communication overhead, in addition to regular authentication, includes sending *kn* numbers from the server to the client and sending *k* numbers from the client back to the server for evaluation of the final result.

4. Conclusions

The primary advantage of the proposed approach is that we are able to achieve classification of a strongly encrypted feature vector using generic classifiers. In fact, the authentication server need not know the specific biometric trait that is used by a particular user, which can even vary across users. Once a trusted enrollment server encrypts the classifier parameters for a specific biometric of a person, the authentication server is verifying the identity of a user with respect to that encryption. The real identity of the person is hence not revealed to the server, making the protocol, completely blind. This allows one to revoke enrolled templates by changing the encryption key, as well as use multiple keys across different servers to avoid being tracked, thus leading to better privacy. The proposed blind authentication is extremely secure under a variety of attacks and can be used with a wide variety of biometric traits. Protocols are designed to keep the interaction between the user and the server to a minimum with no resort to computationally expensive protocols such as secure multiparty computation (SMC) [15]. As the verification can be done in real-time with the help of available hardware, the approach is practical in many applications. The use of smart cards to hold encryption keys enables applications such as biometric ATMs and access of services from public terminals.

References

[1] R.M. Bolle, J.H. Connel, N.K. Ratha, Biometric perils and patches, Pattern Recognition 35 (2002) 2727–2738.

[2] C. Fontaine and F. Galand, “A survey of homomorphic encryption for nonspecialists,” *EURASIP*, vol. 1, pp. 1–15, 2007.

[3] C. Gentry, “Fully homomorphic encryption using ideal lattices,” *STOC*, pp. 169–178, 2009.

[4] S. Avidan and M. Butman, “Blind vision,” in *Proc. Eur. Conf. Computer Vision (ECCV)*, 2006, pp. 1–13

[5] F. Farooq, R. M. Bolle, T.-Y. Jea, and N. Ratha, “Anonymous and revocable fingerprint recognition,” in *CVPR Biometrics Worshop*, Jun. 2007, pp. 1–7.

[6] A. Menezes, P. C. van Oorschot, and S. A. Vanstone*, Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press, 1996.

[7] A. KiselL, A. Kocochetkov, and J. Kranauskas, “Fingerprint minutiae matching without global alignment using local structures,” *INFORMATICA*, vol. 19, no. 1, pp. 31–44, 2008.

[8] V. N. W. A. Ratanamahatana, Hand geometry verification using time series representation Sep. 2007.

[9] L. Wiskott, J.-M Fellous, N. Krueuger, and C. M. von der Malsburg*, Face Recognition by Elastic Bunch Graph Matching, ser. Intelligent Biometric Techniques in Fingerprint and Face Recognition*. Boca Raton, FL: CRC Press, 1999.

[10] J. Yang, J. Shin, B. Min, J. Park, and D. Park, “Fingerprint matching using invariant moment fingercode and learning vector quantization neural network,” *ICCIS*, vol. 1, pp. 735–738, Nov. 2006.

[11] M. Faundez-Zanuy, D. A. Elizondo, M. Ángel Ferrer-Ballester, and C. M. Travieso-González, “Authentication of individuals using hand geometry biometrics: A neural network approach,” *Neural Process. Lett.*, vol. 26, no. 3, pp. 201–216, 2007.

[12] V. Blanz and T. Vetter, “Face recognition based on fitting a 3D morphable model,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25-9, no. 9, pp. 1063–1074, Sep. 2003.

[13] D. M. Monro, S. Rakshit, and D. Zhang, “DCT-based iris recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 4, pp. 586–595, Apr. 2007.

[14] T. Blum and C. Paar, "High-radix montgomery modular exponentiation on reconfigurable hardware," *IEEE Trans. Comput.*, vol. 50, no. 7, pp. 759–764, Jul. 2001.

[15] A. C.-C. Yao, "How to generate and exchange secrets," *Foundations of Computer Science*, pp. 162–167, 198

MS.KARTHIGAIVENI received M. E. degree in computer and communication from anna university, Chennai in 2005. She is working in Periyar Maniammai University, Thanjavur, Tamilnad for the past 7 years as Assistant Professor in the Deparment of InfomationTechnology. She guided more than 20 projects for UG and PG students. Her current research focus is on Image processing and wireless networks.

R.SWATHIRAMYA received the bachelor of Engineering degree in 2010 and pursuing the master of engineering(SoftwareEngineering). Her research interests are in the area of Networks, Network security, biometrics, Imageprocessing, cryptography R.Swathiramya B.E, Final year M.E(software engineering) PG Student,Periyar Maniammai University, Thanjavur, TamilNadu, India

U.R.V.NANDHINY received the bachelor of Engineering degree in 2006 and pursuing the master of engineering (Software Engineering). Her research interests are in the area of Networks, Network security, Datamining U.R.V.Nandhiny B.E, Final year M.E(software engineering) PG Student, Periyar Maniammai University, Thanjavur, TamilNadu, India

S. SIVARANJANINACHIYAR received the bachelor of Engineering degree in 2010 and pursuing the master of engineering (Software Engineering).Her research interests are in the area of Networks, Network security, Datamining, cryptography S.Sivaranjani Nachiyar B.E Final year M.E(software engineering) PG Student, Periyar Maniammai University, Thanjavur, TamilNadu, India

S.JENILA received the bachelor of Engineering degree in 2010 and pursuing the master of engineering (Software Engineering).Her research interests are in the area of Network security, Datamining, Data Structure,image processing S.Jenila B.Tech, Final year M.E(software engineering) PG Student,Periyar Maniammai University, Thanjavur,TamilNadu, India