International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 5 Issue: 6 1261 – 1265

A Treatise on the use of Finger Print for a Biometric Authentication System

Jayati Ghosh Dastidar
Department of Computer Science
St. Xavier's College, Kolkata (Autonomous)
Kolkata, India

***Abstract*—**In this paper an attempt has been made to present a scheme that combines Biometrics, in the form of finger prints with passwords to answer some key authentication questions such as who one is, and what does one know. In addition nonces and time-stamps have been used to prevent replay attacks. The method proposed here is a two-factor authentication algorithm that can be used between a client and a server or in general a service requester and a service provider. The method is resistant to man-in-the-meet attacks and also does not depend upon third-party authentication agents.

Keywords*-biometric; authentication;finger-print;nonce;symmetric-key; cryptography*

*****

I. INTRODUCTION

*Who are you?*is a question that we everyday ask and get asked [1]. It has a lot of importance and significance in the world of cryptography. The whole concept of authentication is based on determining who an individual user is, before allowing the user to go ahead and perform actual business transactions using the system.Authentication is the first step in any cryptographic solution. This is so because unless we know who is communicating, there is no point in encrypting what is being communicated. As we know, the whole purpose of encryption is to secure communication between two or more parties. Unless we are absolutely sure that the parties really are what they claim to be, there is no point in encrypting the information flowing between them. Otherwise there is a chance that an unauthorized user can access the information. In cryptographic terms, we can put this in other words: there is no use of encryption without authentication.

Taken from the Greek words – '*Bio*' meaning life, and '*Metric*' meaning the measure (study) of life;*biometric* technologies are defined as automated methods of identifying or authenticating the identity of a living person based on a physical or behavioral characteristic. A biometric device is perhaps the ultimate attempt in trying to prove *who you are*. This science exploits the uniqueness of parts of the human body like fingerprints, the iris, face and geometry of the hands. Recent advances in technology coupled with a significant price drop make biometric authentication systems a viable alternative.

Positive user identification is the key, but it must be easy to use and cost-effective; therefore, finger imaging is the answer. The following points are in support of this claim

- Fingerprints do not change over time.
- Fingerprints stop unauthorized access.
- All fingers are unique, which allows each person to have ten easy use of identifiers.
- Base of all world-wide identification.
- Fast and easy to use.
- We do not forget our fingers.
- Users respect them, fraudsters are afraid of them.
- Low-cost solution.
- Protects privacy.

Fig. 1 depicts how a Biometric system works and the possible points of attack in a biometric authentication system.Of the possible attack points, protection against all but the second one is supposed to be in-built within the biometric reader itself. Hence, it is felt that the second point of attack, viz. *"Resubmitting previously stored digitized biometric signal during sample transmission"* needs special attention during the implementation of a Biometric Authentication System.

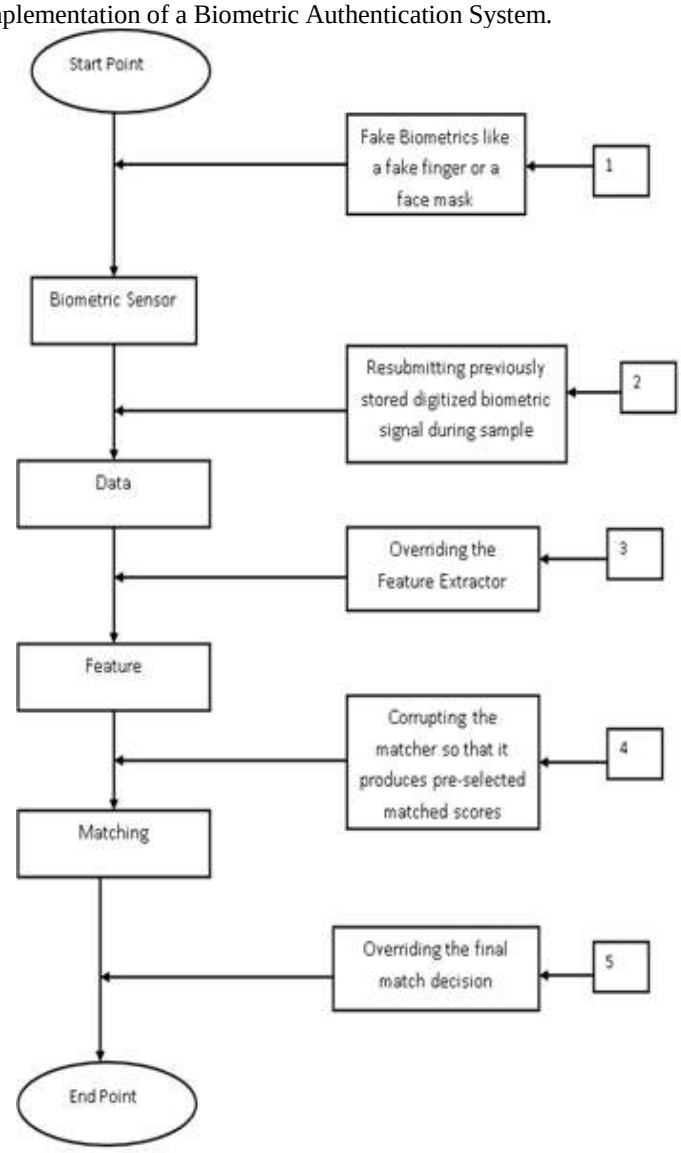

Figure 1. Working of a biometric authentication system

II. DELIVERABLES

The key considerations for the design of a biometric authentication system are:

- A previously collected sample should not be resubmitted for gaining access into the system. Thus the system should be protected against Replay Attacks.
- A proper authentication mechanism should be in place by which the sample giver should be uniquely able to identify itself to the sample matcher. This means mechanism for protection against Non-Repudiation Attacks should be in place.
- Biometric authentication should be further bolstered by the use of passwords/PIN to answer questions to "Who you are?" and "What you know?", simultaneously.

Designing a correct authentication protocol is harder than it looks. Three general rules that often help are as follows [2]:

- Have the initiator prove who he is before the responder has to.
- Have the initiator and responder use different keys for proof.
- Have the initiator and responder draw their challenges from different sets.

The image data flow between the client PC and the server requires security. This can be achieved by the use of strong cryptographic techniques such as *3-DES* or some *Public Key Infrastructure Systems (PKI)*[3], or an Authentication mechanism such as *Kerberos*[2] that is capable of defying Replay Attacks also.

Public-key infrastructure (PKI) is the combination of software, encryption technologies, and services that enables enterprises to protect the security of their communications and business transactions on the Internet [1]. This feature is important when considering the use of PKI in the trusted device model. This model allows trusted devices to accept digital certificates from outside sources and encrypt and sign the data with their own certificates, this is essential for authentication and online transactions.

3-DES is publicly known algorithm that converts binary data to cipher data and cipher data to binary data using the same key. 3DES is an approved symmetric encryption algorithm, which uses 64 bits (56 bits as key) to encrypt 64-bit block data. Also 3-DES uses three keys to encrypt and decrypt data, so effective key length is 168 bits.

Kerberos is an authentication protocol based on the Needham-Schroeder protocol that allows work stations to access network resources in a secure manner.

The Authentication method proposed here has been adapted from the very famous Kerberos Authentication Protocol [4]. To be precise the proposed method has exploited the concept of 'nonce' to prevent replay attacks.

Before proceeding with the description of the algorithm we need to clarify the following points (assumptions):

- The Biometric samples are stored in a central database under the custody of the server.
- The server is the entity that will perform the authentication and is hence the authenticator.
- The client is the one who will submit his/her biometric sample for authentication to the server.
- Both the server and the client are equipped with intelligent terminals and hence will be able to perform cryptographic tasks.
- The server is a trusted authority.
- The client will henceforth be known as the 'user'.

Every user has to first go through an enrollment process, wherein he/she will have to go through the following steps:

Step 1: A user account is created. He/she is given a user id which will have to be used in every session, and by which he/she is uniquely known.

Step 2: The user also chooses a password. A derivative of this password will serve as a secret key that will be used for cryptographic operations required in a session. The password or a derivative of the password is supposed to serve as a symmetric key for encryptions and decryptions.

Step 3: The user gives a few live samples (typically three) of his/her finger print which will be used for creating a template at the server end. The live sample will be used for matching with live samples supplied later on for authentication.

The objective of this project is to develop this authentication algorithm. This algorithm is expected to answer the two of the most crucial questions *"Who are you?"* and *"What do you know?"*

III. DESIGN METHODOLOGY

The Authentication algorithm that has been talked about in the 'Deliverables' section quite effectively combines the use of password and finger prints to bolster the authentication mechanism. The 'finger print' of a user gives the answer the question "Who are you?". Whereas the 'password' effectively answers the question "What do you know?".The proposed authentication algorithm proceeds as follows:

Step 1: This step is initiated by the user. The user sends his/her login id/user id (assigned during enrollment) to the server. The server understands that the purported 'user' is requesting for logging in.

Step 2: This step is initiated by the server. The actions performed in this step are as follows:

a) The server generates a random challenge (akin to a nonce). Let this be R.

b) The server creates a hash of the user's password H(P) (P is the password of the user, taken from the user database maintained by the server) using the hash function H. Let H(P) = h.

c) The server now encrypts R with the help of a symmetric key algorithm and the hash h as the secret key. That is we calculate $E_h(R)$, where E is a symmetric key algorithm function.

d) The server sends this to the user.

Step 3: The initiator of the third step is the User and is initiated when he/she receives the encrypted challenge from the user. The steps are as follows:

a) The user receives C supposedly equal to $E_h(R)$.

b) The user is now asked to enter to enter the password, P.

c) The user-end computer independently computes the hash of P using the same hash function H(P) = h.

d) This is now used in decrypting C. Dh(C) = R' (should be equal to R).

e) The user is now required to provide a live sample of his/her finger print.

f) This image is now encrypted using R' using a symmetric key algorithm.

Step 4: This step is initiated by the server after receiving the encrypted image file. The actions performed are:

a) The server decrypts the file using the previously generated R.

b) The server then matches the decrypted image using the sample template, stored in the user database.

Access is allowed if the two match else it is disallowed. The algorithm has been illustrated in Fig. 2.

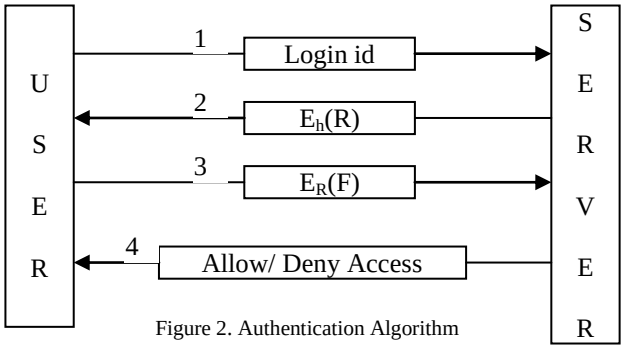

Figure 2. Authentication Algorithm

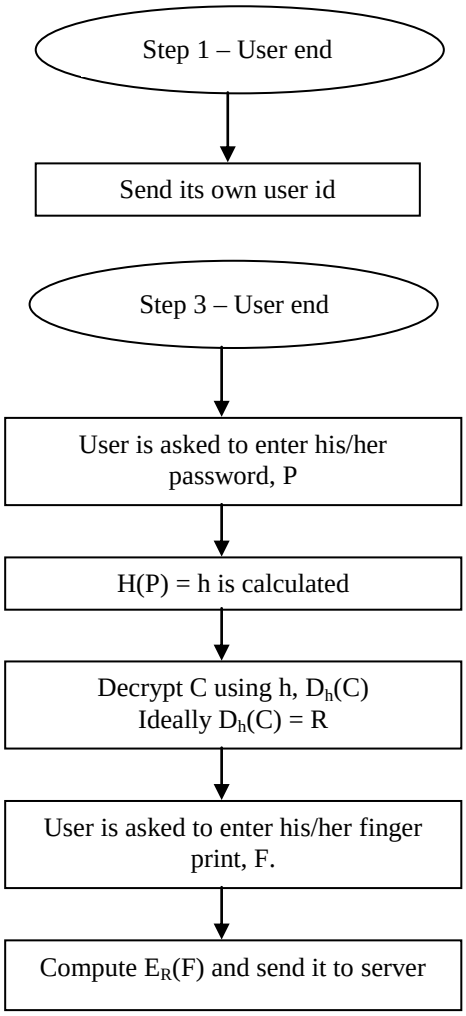

Figure 3. User-end activities

The whole process is initiated by the user. The user sends his/her login/user id to the server in message 1.

On receiving message 1, the server computes the hash, h of the password that has been securely stored in the server database. The server then generates a number, R. The number R, has a random component, a time stamp and if possible some secret server information embedded within it. Together R will serve as a one time session key. The server encrypts R with h and sends it to the user as message 2.

At this point the user is asked to enter his/her password. He user end computer independently computes h. The password is cleared from the user's end computer. The h is used in decrypting message 2 and obtaining R from it. An intruder will not be able to do this because he/she does not know the user's password and h can be obtained only from it. The user is now asked to give his/her live finger print sample. This sample, F is now encrypted using R and sent to the server in the form of message 3.

The server on receiving ER(F) recovers F and matches with an existing template in the database. If the match is within acceptable limits,access permission is given, else it is denied in the form of message 4.

Fig. 3 shows the detailed activities at the user's end and Fig. 4 shows the detailed algorithm at the server's end.

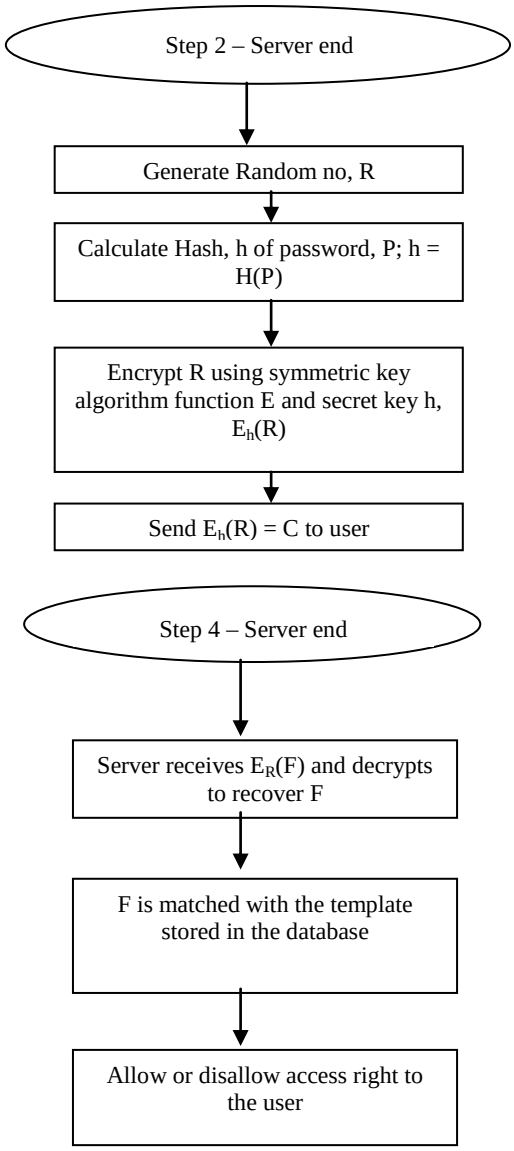

Figure 4. Server-end activities

The said concept has been developed as an attempt to simplify the more sophisticated **Needham-Schroeder**, **Kerberos** and **Otway–Rees** authentication Protocols. The protocols have been modified in an attempt to adapt for combination with Biometric Samples. Further the protocols are used for both way authentication between a sender and a receiver. Here the protocol has been used for authentication of a user by a server (one way). It has been assumed that the

server is a trusted entity, free from being hacked and that no imposter will serve as a server.

IV. RESULTS AND DISCUSSIONS

As explained in Fig. 2, Fig. 3 and Fig. 4 the proposed algorithm requires the following:

a) A symmetric key encryption/decryption algorithm
b) A hash function
c) A good random number generator

After studying several algorithms the following has been the choice of the algorithms:

- A symmetric key algorithm - DES
- A hash function – SHA-1

The symmetric key encryption algorithm will be required in the following circumstances:

- For sending the one time session key, R from the server to the user in an encrypted form.
- For sending the finger print image file in an encrypted form, from the user to the server.

The decryption algorithms will be required at the opposite ends. The choice of DES can be further bolstered by using 2-DES or 3-DES or any other strong symmetric key encryption algorithm.

The hash function will be required in the following circumstances:

- For finding a derivative of the user's password in the server. This derivative has been used for encrypting R at the server.
- For recalculating h at the user's end. This h will be required to recover R, which has been sent by the server in an encrypted form.

The choice of the hash function is also flexible. SHA-1 can be replaced by any other suitable algorithm.

As for the random number generator, there are several well-known methods to generate random numbers any one of them van be chosen. In this case I have preferred to use the rand() function provided in the C library, just to simplify matters and keep my work field limited to cryptography and not deviate into statistics.

The proposed algorithm has the following points in its support:

a) The random challenge is a random number, preferably generated just once (nonce). Hence it cannot be predicted by anyone and reused later on. Re-usage is totally overruled because the random number, R also has a time stamp component embedded within it. Generation of R by anyone else other than the server is also overruled because it has some secret information derived exclusively from the server. This concept is also used in Cookies in the Oakley Key Determination Protocol used in ISAKMP (Internet Security Association and Key Management protocol) used in the IP-Sec Protocol. One of the problems that symmetric key cryptography faces is that the strength of the algorithm lies in the secrecy of the key used. If by any chance the key is exposed, the whole algorithm falls apart. In this algorithm a new secret key is used for every session. This means even if, the key is revealed during one session, it will be of no use in the next, or in any other session.

b) Another problem that arises in many authentication protocols is that, the authentication is based on the password assigned to a user. Very often the password has to travel through the network for being authenticated. In this algorithm however the password never has to travel. A derivative of the password (hash) is calculated. This hash is also not transmitted as it is. Instead it is used to encrypt the one time session key to be transmitted from the server to the user. The user then decrypts the transmitted message with the aid of the hash (independently calculated). At no time does the user have to transmit the password or the hash to the server. This characteristic makes it doubly secure. One may argue that since the hash is being used, it may be possible to recover the password from it. However the hash function that will be used to calculate the hash from the password will be a one-way trap-door kind of a function, h=H(P). Such functions have the property that it is computationally easy and feasible to calculate h, given P. However it is computationally infeasible to compute P, given h. Formally speaking the hash function has the following properties:

- Given a message it, should be very easy to find its corresponding message digest (hash). Also for a given message the message digest must always be the same.
- Given a message digest, it should be very difficult to find the original message for which the digest was created.
- Given any two messages, if we calculate their message digests must be different. This property is often stated as two different properties [3]:

a) Weak Collision Resistance: For any given block x, it is computationally infeasible to find y ($y \neq x$) such that H(y)=H(x).

b) Strong Collision resistance: It is computationally infeasible to find any pair (x,y) such that H(x)=H(y).

The first two properties state the one-way-ness of the algorithm. The last is a way by which an intruder will not be able to construct a password of its own.

The discussions so far indicate that the security of the algorithm is dependent on the password. However, it is not so. Even if the password is cracked, the second and perhaps more secure, barrier is still left to be crossed, viz. the finger print. All of us know that the finger print of a person is unique to oneself. Thus no one else will be able to produce a finger print of a genuine user.

However what can be done is that, an artificial mould of one's finger can be created. This mould can then be used to provide the finger print at the user's end. Thus a desperate unauthorized user may gain entry into the system by making a mould and forcing the password out of him or break the hash. This problem can also be overcome by having a sensor installed into the biometric reader that will be able to sense the temperature, or even better the pulse of the finger (mould) that is being provided. Such advanced biometric readers are not very much far-fetched. Sensitive places, such as defense installations may well be equipped with such readers.

We will now proceed to see how the Replay Attack is prevented by using this algorithm. Before doing so we have to first understand what an Replay Attack is all about and how it becomes relevant in this scenario.

Replay Attack is a form of attack on a system wherein an attacker gets hold of a message, and attempts to re-send it, hoping that the user does not detect this.

In this case an attacker may intercept an encrypted finger print image file during its transmission from the user to the server. The attacker may store this intercepted message. Later on he/she may re-send it to the server thinking that the server

will not be able to detect this, and thus try an gain entry into the system. However this is not to be. This is because the user sends the finger print image file in an encrypted form. The encryption has been performed using a one-time key that has randomness as well as a time stamp component. Thus a previously stored encrypted image cannot be re-transmitted. This will be detected by the server. The time stamp component instills a lifetime into the image file. Thus, an image file which has been collected a very long time ago cannot be transmitted after a given period of time.

This is how Replay Attacks are prevented.

LIMITATIONS

The server has been assumed to be a trusted authority – This means that the User database present in the server has will not be subject to any attacks and that it will remain secure. Therefore no steps have been taken to secure the server. In reality however this risk could not be taken.

No imposter will mimic the server – It has been assumed that the user will always be communicating with the authentic server and not to an imposter. The Algorithm has nothing in it to authenticate the server. Hence this Algorithm is susceptible to Non-Repudiation Attacks.

The Password is known solely to the user – The transmission of the one-time key has been done using the hash of the password. Thus if the password is somehow, leaked to a third party, then the basis of the major authentication question "What do you know?" falls apart. To prevent this from happening passwords should be frequently changed, and the user should take measures to keep it a secret.

Server and user are not located at remote locations – It has been assumed that the user and the server are located at geographically near location. This means the user can physically go to the server for the enrollment process and for a frequent change of password process. However this may not be the case. In case the server and the user would be present at geographically remote locations, then a suitable Key Agreement Algorithm would be required.

The password is unique – For the purposes of this algorithm it has been assumed that the password is unique to a user. Violation of this assumption would mean that two different users would use the same hash. Thus the answer to the question "What do you know?" would not be uniquely obtained. This means two users would know the same thing.

The server is centralized – The server that will be doing the authentication is centralized. Hence it may be overloaded with encryption work, if too many authentication requests come in simultaneously. This may result in a Denial of Service to others. An attacker may take advantage of this, and send a barrage of requests, so as to exhaust the server's resources and thus launch a Denial of Service attack (DOS). Such attacks can however be prevented by the use of cookies. A user with a valid cookie only should be entertained, as is done in the Oakley Key Determination Protocol.

Server is the single point of compromise – As with all centralized authentication systems, if the server-database is somehow compromised, then the finger print images and passwords of all users will be at the disposal of the attacker. No system will be secure enough, if the attacker knows that there is just one point in the system that needs to be brought down.

Communication Problem – Since all the users will try to communicate to the central server, this may result in a congestion problem in the network routes leading up to the server.

What if Compromised – If a biometric sample is once compromised, it is compromised forever. This is because a user will not be able to produce an alternate thumb for providing a live sample. Then off course a sample from some other finger has to be taken.

V. CONCLUSION

Reliable user authentication is becoming an increasingly important task in the Web-enabled world. The consequences of an insecure authentication system in a corporate or enterprise environment can be catastrophic, and may include loss of confidential information, denial of service, and compromised data integrity. The value of reliable user authentication is not limited to just computer or network access. Many other applications in everyday life also require user authentication, such as banking, e-commerce, and physical access control to computer resources, and could benefit from enhanced security.

Fortunately, automated biometrics in general, and fingerprint technology in particular, can provide a much more accurate and reliable user authentication method. Biometrics is a rapidly advancing field that is concerned with identifying a person based on his or her physiological or behavioral characteristics. Examples of automated biometrics include fingerprint, face, iris, and speech recognition.

Yet, any system, including a biometric system, is vulnerable when attacked by determined hackers. We have highlighted eight points of vulnerability in a generic biometric system and have discussed possible attacks. We suggested several ways to alleviate some of these security threats. Replay attacks have been addressed using data-hiding techniques to secretly embed a telltale mark directly in the compressed fingerprint image. A challenge/response method has been proposed to check the liveliness of the signal acquired from an intelligent sensor.

REFERENCES

[1] A. Kahate, Cryptography and Network Security, Tata McGraw-Hill Education, 2003.

[2] A. S. Tanenbaum and D. Wetherall, Computer Networks, Pearson Education, 2013.

[3] W. Stallings, Cryptography and Network Security – Principles and Practice, Prentice Hall, 2003.

[4] B. Clifford Neuman and T. Ts'o , "Kerberos: An Authentication Service for Computer Networks". IEEE Communications. 32 (9): 33–8. doi:10.1109/35.312841, 1994.