Encryption using finger-code generated from fingerprints

Cynthia Sthembile Ntshangase
The Council for Scientific and Industrial Research
Pretoria, South Africa
smlambo@csir.co.za

Meshack Bafana Shabalala
The Council for Scientific and Industrial Research
Pretoria, South Africa
mshabalala@csir.co.za

***Abstract*—In this paper, the literature survey of different algorithms for generating encryption keys using fingerprints is presented. The focus is on fingerprint features called minutiae points where fingerprint ridges end or bifurcate. Minutiae points require less memory and are processed faster than other fingerprint features. In addition, presented is the proposed efficient method for cryptographic key generation using finger-codes. The results show that the length of the key, computing time and the memory it requires is efficient for use as a biometric key or even as a password during verification and authentication.**

Keywords—fingerprint; minutiae; cryptography; finger-code

I. INTRODUCTION

The identity of an individual is the abilities, beliefs, personality, appearances, and/or expressions that differentiate an individual amongst others [1]. There are three main methods that are used to prove identity, namely: Something we have, (e.g. ID Card/Document or smart card), something we know (e.g. password or PIN number), and something we are (e.g. Biometrics such as fingerprints or iris). During old days the only methods that were used are something we have and something we know. Nowadays many applications are replacing these old methods or combining them with something we have (e.g. biometrics). Biometrics is a biological or behavioral characteristic of a human being that can distinguish an individual from another, such as; fingerprints, iris, ear, face, gait, footprints and more. The literature has shown that most biometrics if analyzed in details can be used for identification or verification because of their uniqueness [2]. The focus in this paper in more on fingerprints because they are widely used, easy to capture and has been successfully used by law enforcement for more than a century [2].

In the literature, it has been shown that combining fingerprints and smart cards higher security can be achieved [1]. A smart card is a security token that has an embedded chip. A smart chip is a small microprocessor or memory chip that is embedded with smart cards so that is called 'smart cards' [1]. The chip stores electronic data and programs that are protected by advanced security features. It gains the processing power when it is connected to an appropriate reader so that it can serve different applications. Smart chips are increasingly used nowadays because provides data portability, high security and convenience [2]. Although the concept of smart cards is not particularly new, however, it is the practical use of smart cards and smart chips in different applications that have never been more popular. Smart chips are used in smart cards, tokens, bio-cryptography, ID cards, Bank cards, cell phones, computers, health departments, transportation, and other devices with applications that require more security.

Bio-cryptography is one of the applications nowadays where smart cards and fingerprints (or biometrics) are combined for the purpose of achieving high security. In the field of authentication, conventional cryptography depends on either something we know or something we have. The problem with these methods is that they cannot authenticate genuine individuals. Hence, including something we are as fingerprints can add more security. However, fingerprints alone are vulnerable against types of attacks, such as false fingerprints, matcher attack [4],[5]. That is why fingerprints are combined with cryptographic key methods for high security.

In general, a characteristic of cryptographic keys is large unpredictable random numbers. The methods of using cryptographic keys use the concept of asymmetric and symmetric key cryptography [6]. In symmetric concept, called secret key, use the same key for both encryption and decryption; asymmetric concept, called public key, use different keys for encryption and decryption. The symmetric concept is stronger than asymmetric concept because symmetric uses the private key. Symmetric algorithms include AES, and asymmetric algorithms include RSA [7].

Although different works have been proposed to generate cryptographic keys, there is still a gap in achieving strong bio-cryptographic key at a short computation time and memory size. In this paper, we propose a new method of generating a strong cryptographic key at a short time and memory. Firstly, existing methods have been presented for generating bio-cryptographic keys and the comparison with the proposed approach in terms of, key length, and computing time.

The rest of this paper is structured as follows: Section 2 is the review of related work. Section 3 is the explanation of the proposed method for generating the cryptographic key. In section 4 is the comparison of methods with the proposed method. Section 5 is the discussion of the results, and the last section is the conclusion and future works.

II. RELATED WORK

In this section presented are recent different algorithms used to generate fingerprint keys and applications where they have been used.

In 2015 [7] proposed fingerprint encryption algorithms using the fingerprint keys and a look-up table for the

encryption and decryption process. This method used the fingerprint key with a simple equation in order to generate the encryption key. The encryption key used to encrypt and decrypt data. The results showed three different ciphertexts in a hexadecimal form which are encrypted by using this method. The differences among these texts are very large. The large differences due to use a larger lookup table with (256×256) dimensions in encryption. The simulation results presented that this encryption method gives high security with a good performance. However, this algorithm uses minutiae type which is easily affected by the noise, distortion, and transformation of the fingerprint image, which can result in a different key. Performance measures used are:

1. the length of the fingerprint key,
2. the symmetric concept, and
3. the size of columns and rows of the look-up table (256×256) dimensions

In 2015, [9] authors improved work proposed in [3] from the algorithm and described the drawbacks of the previous method, namely, effects affecting the length and angle between average point and minutiae. Since one fingerprint can only contain a limited number of minutia points, approximately between 20 - 40. Therefore there are high possibilities that if attackers acquire minutiae information successfully, they can narrow down the search range of Key. In other words, the cryptographic keyspace created by 20 - 40 minutiae points is small and it will be easy to hack. They have proposed a solution to solve this problem by increasing the total number of minutiae. This is achieved by adding noise through image enhancement and this noise will increase the number of minutiae and also will increase the key strength. However, the results of this algorithm can be affected by adding noise because it changes the originality of fingerprint features. Performance measures:

1. Comparison between fingerprint encryption and AES algorithm
2. Brute force attack to measure number of times required to break the key, in seconds

In 2015 [11] authors derived a secure cryptographic key by incorporating dual fingerprints of a human being, so as to provide better security. Feature level fusion of fingerprints is suggested for cryptographic key generation. Fingerprints are acquired, processed and finally extracted features are combined together to generate the 256-bit cryptographic key. The experimental results have demonstrated the efficiency of this approach to produce user-specific strong cryptographic keys. Performance measures:

1. Confidentiality and integrity
2. Man in the Middle Attack
3. Exhaustive Search Attack
4. Brute Force Attack

In [12] 2015 authors presented a technique to generate an asymmetric key pair for RSA by combining fingerprint features. Generate keys from extracted minutiae points including ridge ending, ridge bifurcation, crossover and isolated points. After extracting minutiae points and store their locations in arrays, shuffling of the individual array is performed. At the end, all duplicate numbers are removed. The result of this process is a large random number of 1024 bits array converted into hex. Results showed that that the presented random number can be used in any encryption algorithm to generate the key. Authors then generated two keys that can be used for the RSA encryption and decryption which are 512 bytes long. As randomness is involved at many levels starting from fingerprint selection to key generation, the complexity of key is expected to be high. The generated RSA key pair can be used as any conventional RSA key pair to send and receive encrypted data. Performance measures used are:

1. Randomness of generated number
2. Fingerprint key length
3. Complexity of the key

In 2012 [13] presented an algorithm for deriving the key from fingerprints for the applications based on the Elliptic Curve Cryptography (ECC). This method can provide high security with good performance in terms of computational and memory requirements because it requires small keys. ECC method uses the concept of algebraic structure for elliptic curves over finite fields to generate cryptosystems. This approach was implemented in MATLAB and was used to generate a cryptographic key of different sizes, which is suitable for any real time cryptography. The result of this method is a key length of 128 bytes. The average computation time required to generate the key is 0.032ms. Performance measures used:

1. Fingerprint length
2. Computation time

In 2014 [14] authors presented a method that generates the bio-cryptographic key from fingerprint minutiae. The first step is to generate two vectors from extracted minutiae set. These vectors are used on three functions namely, concatenation, salting process, and shuffling process for the purpose of generating random prime numbers. This generated number is then used in RSA cipher algorithm to generate 2048 key. The results presented that the method can be implemented in different processors like a microcontroller, ARM, and FPGA. Performance measures used are:

1. Fingerprint key length
2. Complexity of the key generator algorithm

Recently in 2016, [10] a biometric cryptosystem is proposed which provides a better secures approach to encryption and decryption methodologies by using a key-pair generated from fingerprint impressions. This method follows the encryption scheme that is similar to the RSA encryption algorithm because of the similarities in the key-pair application towards encryption and decryption. Presented results showed that computation time required to generate a cryptographic key, public or private is 0.002 seconds. Performance measures used:

1. Computing time required to compute the fingerprint key for encryption
2. Security

III. Proposed Method

The proposed method is a novel approach of a finger-code based type minutiae key generation where minutiae points are transformed into a binarized form. In order to illustrate how this approach works, let's first assume a random set of minutiae points in a 2D Cartesian plane. In each set, reference minutiae points are chosen at random. Each minutia will be a reference. For each reference minutia, the aim is to represent the neighborhood of the reference minutia in binary form. In order to achieve this for each reference minutia point, a square grid/tessellation of N by N dimensions is constructed where the reference minutia is at the center. The square grid is further subdivided into ΔS sub-squares. The grid takes the orientation of the reference minutia point as seen. Each sub-square in the tessellation represents a single bit. Initially, all sub-squares are set to zero bit. For each sub-square that contains a minutia point, the square is set to bit 1 as represented by the white color, shown in Fig.1. Note that the sub-square has to contain only one minutia point. In a case where two or more minutiae are in one sub-square then the tessellation would be regarded as invalid.

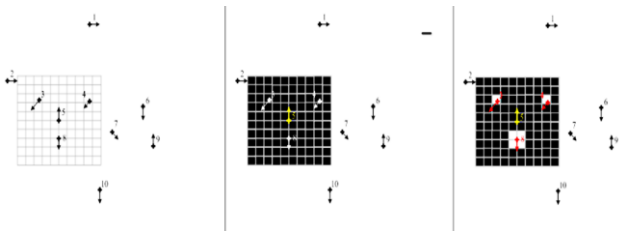

Fig. 1.Representation of minutiae square tessellation

The explanation of the procedure of the proposed method is as follows: The input is a set of minutiae extracted from a fingerprint image. The output is the text file containing the generated finger code, as shown in Fig.3.

A. Algorithm Description

Inputs:

Extracted set of minutiae points I, the size N of an N by N square tessellation, and the increment size for grids in a tessellation.

Outputs:

Finger codes of binary string with 0's and 1's

Procedure:

For each set of points

- For each minutia, set each as a reference minutia and create a square tessellation that is in a direction of the current minutiae orientation, as shown in Fig.2.

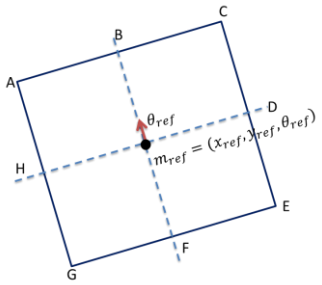

Fig. 2. A representation of a square tesselation with reference minutiae at the center.

- Calculate four coordinates A, C, E, and G that represent points on the corners of the tessellation. This is performed by first calculating B and F coordinates, given a reference point and a distance from *m_ref* to B or F.
  - $x_B = x_{box_{size}} * cos(\theta_{ref}) + x_{ref}$
  - $y_B = y_{box_{size}} * sin(\theta_{ref}) + y_{ref}$
- To compute F, as it is in the opposite direction of θ_{ref}, add 180 degrees in to θ_{ref} and compute F coordinates as in B.
  - $x_F = x_{box_{size}} * cos(\theta_{ref} + 180) + x_{ref}$
  - $y_F = y_{box_{size}} * sin(\theta_{ref} + 180) + y_{ref}$
- To calculate A, C, E, and g coordinates, the gradient of a line perpendicular to *m_ref* line need to be known. This is performed by adding 90degrees angle in θ_{ref}. Given
  - $\theta_{perp} = \theta_{ref} + 90$, A and C can be computed by setting B as a point in the middle as performed when computing B or F.
- Now that coordinates of the square tessellation are computed, store them in an array with the reference minutiae, then find minutiae points inside the tessellation excluding the reference point.
- For each point in the set I find points within the tessellation and store them in the same array where the coordinates of the tessellation are stored.
- Rotate all points so that the reference minutiae orientation is 90 degrees.
- Translate neighbors of the reference minutiae so that they are moved to the origin of an array, (excluding the coordinates that represent the tessellation). This process is called coordinate normalization.
- Represent the location in an array where there are minutiae points in binary.
- Return the representation and store it in a text file, this process is illustrated in Fig.3.

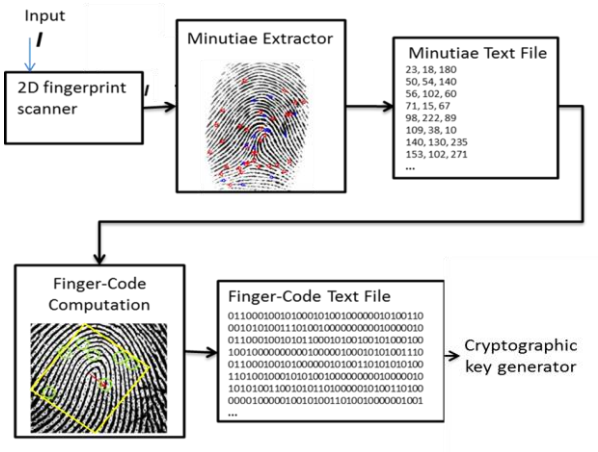

Fig. 3. Process of finger-code generation

IV. Comparison of Fingerprint Key Generation Algorithms

A. Comparison measures

The Performance measures that are used during this study are:

1) Computing time or speed of encryption key generator algorithm

2) Strength of fingerprint key measured using the length of the key

These performance measures are used because in a runtime systems are the main measures that affect the system performance. In addition, smart chips have limited resources in terms of memory and instruction sets.

B. Comparison table

This table shows comparison measures used with existing methods for generating bio-cryptographic keys. Different methods have been presented which generate key length of 128 and or 256 bytes and compared to the proposed method. These method in [13], [7], [9] and [10] were selected for comparison because of the availability of their algorithms and the consistency in the length of the data used.

TABLE I. Table for Comparison

Year [Ref]	**Comparison Measures**	
	Key length (bytes)	***Computation time (milliseconds)***
2012, [13]	128	0.032
2015 [7][9]	128	1.300
2016, [10]	256	2.000
Proposed Solution	256	0.170

V. Discussion

The presented method offers a few advantages over other methods, namely:

- The templates for the query and reference do not have to be of the same dimensions since the neighborhood is independent of the exact location of minutiae rather the relative relationship is encoded
- The binarization of the neighborhood is well suited for a light architecture platform.
- The neighborhood encoding is rotationally and transnationally invariant
- Measured computing time or speed of cryptographic key generation algorithm is 0.17 milliseconds
- The strength of fingerprint key measured using the length of the key; the key length varies depending on the number of minutiae and the size of square tessellation. The larger the square tessellation size, is the larger key length.
- This method can be used to generate keys for asymmetric or symmetric
- Does not depend on the type of minutiae

The proposed method has a significant advantage in terms of the key length because of its option to vary the length, which makes it suitable for the use in different cryptosystems with different key length requirements. In addition, there is no requirement for the cryptographic key to be stored in a protected memory, thus reducing the security threats to the minimum. The fingerprint is an inherent trait of every individual and is distinct. Hence, different cryptographic keys can be generated, making it very hard for an attacker to guess the key.

Conclusion

A new approach of encoding the minutiae neighborhood into binary form has been proposed for generating a cryptographic key that can be used as a key and for verification or authentication. The approach creates a square tessellation that contains sub-squares to encode the minutiae point. The tessellation is created such that no two or more minutiae points can be located within one sub-square. The banalization of the neighborhood is well suited for a light architecture platform with small memory. The variability in key length, large length, and type of our key gives advantages of the key to be used in different applications including, different methods of cryptography such as AES, RSA, and more. In addition, it can be used on smart cards as a biometric template for verification purpose. Measured computing time or speed of cryptographic key generation algorithm is 0.17 milliseconds.

Future works will be to optimize the algorithm in terms of computing time and test the proposed method in the environment of a smart card.

REFERENCES

[1] C. Barral, “Biometrics & Security: Combining Fingerprints, Smart Cards and Cryptography,” Biometrics, Ec. Polytech. Fed. Lausanne,Thesis, 2010.

[2] A. K. Jain, J. Feng, and K. Nandakumar, “Fingerprint matching,” IEEE Comput. Soc., vol. 43, no. 2, pp. 36–44, 2010.

[3] H. Guo, “Smart Cards and their Operating Systems,” Helsinki Univ. Technol. Softw. Multimed. Lab. Tech. Rep., pp. 1–15, 2002.

[4] S. Panjwani and A. Prakash, “Crowdsourcing Attacks on Biometric Systems,” SOUPS ’14 Proc. Tenth Symp. Usable Priv. Secur., pp. 257–269, 2014.

[5] U. Uludag and A. K. Jain, “Attacks on biometric systems: a case study in fingerprints,” Proc. SPIE 5306, Secur. Steganography, Watermarking Multimed. Contents, p. 622, 2004.

[6] S. Barman, S. Chattopadhyay, D. Samanta, and G. Panchal, “A novel secure key-exchange protocol using biometrics of the sender and receiver,” Comput. Electr. Eng., vol. 0, pp. 1–18, 2016.

[7] T. Z. Ismaeel, “A New Cryptosystem based on Fingerprint Features,” Internatinal J. Comput. Appl., vol. 132, no. 12, pp. 31–36, 2015.

[8] T. Zhao, Q. Ran, L. Yuan, Y. Chi, and J. Ma, “Image encryption using fingerprint as key based on phase retrieval algorithm and public key cryptography,” Opt. Lasers Eng., vol. 72, pp. 12–17, 2015.

[9] P. T. Zeyadismaeel and A. S. Names, “Data Encryption Algorithm using Asymmetric Key Derived from Fingerprint Biometric Features,” vol. 4, no. 7, pp. 594–599, 2015.

[10] K. Ankit and J. Rekha, “Biometrics as a Cryptographic Method for Network Security,” Indian J. Sci. Technol., vol. 9, no. 22, pp. 1–6, 2016.

[11] M. Marimuthu, “Dual Fingerprints Fusion for Cryptographic Key Generation,” Int. J. Comput. Appl., vol. 122, no. 23, pp. 20–25, 2015.

[12] S. Singh and J. A. Laxminarayana, “RSA Key Generation Using Combination of Fingerprints,” IOSR J. Comput. Eng., vol. 15, pp. 48–53, 2012.

[13] B. R. Rao, E. V. V. K. Rao, S. V. R. Rao, and M. Rama, “Finger Print Parameter Based Cryptographic Key Generation,” vol. 2, no. 6, pp. 1598–1604, 2012.

[14] M. T. Rashid and H. A. Zaki, “RSA Cryptographic Key Generation Using Fingerprint Minutiae,” Iraqi Comm. Comput. Informatics ,Iraqi J. Comput. Informatics, vol. 1, no. 1, pp. 66–69, 2014.