Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 69 (2014) 1459 – 1468

Procedia Engineering

www.elsevier.com/locate/procedia

24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013

New Enrollment Scheme for Biometric Template using Hash Chaos-Based Cryptography

Marius Iulian Mihailescu*

University of Bucharest, Str. Academiei nr.14 sector 1, Bucharest 010014, Romania

Abstract

In this article we propose a new enrollment scheme for biometric template based on hash chaos-based cryptography (more precisely Davies – Meyer scheme). The idea of the scheme is to create a strong and unique authentication process of the biometric templates and to guarantee the safety of the biometric data.

Keywords: biometrics; enrollment;chaos cryptography; davies-meyer; hash function

1. Introduction

The biometric process, despite its many advantages, it's also vulnerable to attacks that can diminish its security. These possible attacks were analysed by Ratha et al. [1], who separated them into eight classes. In addition, different suggests about the possible attacks on biometric process were made by Dimitriadis [5]. In this paper we approach only the template database attacks, in other words attacks on template database (e.g. modifying an existing template, removing template, adding new template etc.). The biometric template is being stored in sensing device, smart card and central repository. In the center of the system where are stored the biometric templates the imposer can insert a fake template or templates, in order to gain unauthorized access. As a consequence, the legitimate user faces denial of service. A method to prevent this from happening is to use smart cards, in which case the template would be

* Corresponding author. Tel.: +40740030310.
E-mail: mihmariusiulian@gmail.com;

doi:10.1016/j.proeng.2014.03.142

stored, written and erased or destroyed in case of altered technique. Another option is applying strong security controls or protection schemes to protect the template in case of the absence of the first option. [4]
Different scheme are used to the purpose of protecting the database from the imposter. The literature proposes two such classifiable categories of schemes:

- The feature transformation approach
- The biometric cryptosystem.

In this further approach a transformation function (F) is applied to the biometric template (T) while the database stores only the transformed template $(F(T;K))$. Usually, the parameters of the transformation function derive from a random key (K) or the password. Another way to classify these schemes is by categorizing them in invertible and non-invertible transforms. In the case of invertible transforms, an adversary has access to the key and to the transformed template, this way being able to recover the original biometric template (or a close approximation of it), having in mind that security of the invertible scheme is based on the secrecy of the key or password [27]. Yet, the non-invertible transformation schemes are more effective, considering that they apply a one-way function on the template, making so more difficult computationally to invert a transformed template, although the key may be known. In case the helper data is obtained by binding an independent (from the biometric features) key with the biometric template, this wares the name of biometric cryptosystem [29]. The biometric cryptosystem's initial function was to secure, using biometric features, a cryptographic key or to generate directly, from biometric features, a cryptographic key, functions which can also be called helper data-based methods. Depending on the process of obtaining the data, the biometric cryptosystems could also be classified in key binding and key generation systems. Bear in mind that it is difficult to computationally recover the key or the original template, when only the helper data is given [25,26,6].

In order to match in a key binding system it is imperative to recover the key from the helper data, and we do so by using the query biometric features. In the situation in which the cryptographic key is generated directly from the helper data and from the query biometric features and the helper data derivate only from the biometric template, this leads to a key generation biometric cryptosystem. It is also known that there are certain template protection techniques that use more than one basic approach. [11]

One of the invertible transformation biometric protection schemes approached, in which the transformation is realized through user specific key or password, is the bio-hashing or salting. This implies that the user must securely store the key or remember the password in the course of authentication [8]. At this point we can introduce and describe the notion of cancellable biometrics, which refers to the distortion that is systematically and intentionally repeated, with the purpose of protecting biometric template [14]. This type of biometrics is a non-invertible approach, which means that if a cancellable feature is compromised, the features of the distortion change while mapping the same biometrics to a new template, subsequently used [9]. The undistorted (original) biometrics can't be recovered, even in the case of knowing the transformation function and the resulting transformed biometric data. This requires an approach of steganography and watermarking. First of them involves the process of hiding information and its based techniques can be used to transfer critical biometric information from template storage to the matcher [12].

On the other hand, we can use a watermarking technique to protect database and to transfer on channel, because this is a technique that allows one pattern to be embedded into another pattern. But there is also a third technique that can be to prevent channel attack is challenge-response system.
This is the image based challenge-response method [2]. Here, the sensor is being presented with the challenge, which makes the response string to depend on the challenge string and the content of the acquired input image.

2. Preliminaries

The Merkle – Damgard represents the foundation of many hash functions, such as SHA, MD5 [2]. As a general start, a hash function creates a message digest of fixed length from an arbitrary message length. We have to retain the fact that the hash value sometimes refers to a hash-code or message digest [23-27]. Hash functions can be classified into two categories: *hash functions un-keyed* which are based on a single

input parameter, which is the message and *hash functions keyed*, based on two inputs (the message and a secret key).

The hash function presented in this article takes into consideration the following general properties [1]:

a. *Preimage resistance*: essentially, all outputs which pre-specified are computationally impossible to find and to match any input which can hash that output. This is very important as pre-image to find any x' in such manner that $h(x') = y$ when we have any y for which we have a known input.
b. *2nd pre-image resistance:* it is impossible computationally to find and to match a second input which is able to have the same output as any specified input, by having a x given and to find a 2nd – pre-image $x' \neq x$ in such way that $h(x') = h(x)$.
c. *Compression*: the input can be arbitrary finite from the point of view of bit length, and the output is fixed bit length, such as 512-bit, 256-bit, 128-bit.
d. *Collision resistance:* computationally is impossible to find two distinct inputs x, x' that are able to have the same output in a such way that $h(x') = h(x)$.

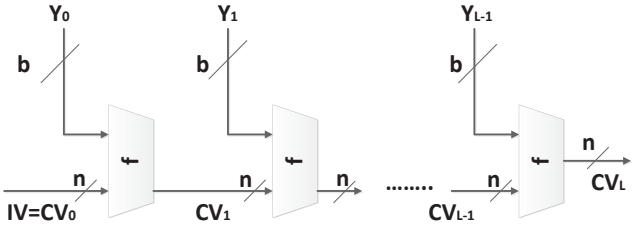

Fig.1. The Merkle-Damgard scheme for iterated hash function.

The Davies-May function represents a one-way compression function for a single-block-length which passes each block of the message (m_i) as a key to that block cipher. It passes the previous hash-value (H_{i-1}) as the plaintext that will be encrypted. The output resulted as a cipher text will be *XORed* $(\oplus)$ with the previous hash value(H_{i-1}), which will produce the next hash value (H_i). For the first round, when we don't have a previous hash value, we use a pre-specified value (H_0).

As a mathematical expression, Davies-May can be represented as:

$$H_i = E_{m_i}(H_{i-1}) \oplus H_{i-1} \quad (1)$$

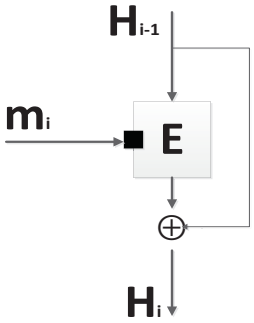

Fig. 2. The Davies-May compression function.

Yi combined in [11] the chaotic iteration with the scheme of Davies-May, using 75-times the iterations of a chaotic tent map under the form of a cipher, in order to generate the hash value. Messages are divided in blocks by this algorithm for further processing. When both the chaotic phase space and parameter space are influenced, the result is that message units are modulated into the chaotic iteration.
This algorithm presents a chaotic tent map $F_\alpha(x_i)$ with parameter α:

$$F_\alpha(x_i) = \begin{cases} \frac{x_{i-1}}{\alpha}, 0 \le x_{i-1} \le \alpha \\ \frac{1-x_{i-1}}{1-\alpha}, \alpha \le x_{i-1} \le 1 \end{cases} \quad (2)$$

where $0 < \alpha < 1$.

Using $F_\alpha(x_i)$, we can define a map G_α as follows:

$$G_\alpha : x_i = \begin{cases} F_{A(\alpha)}(x_{i-1}), & 0 < x_{i-1} < 1 \\ \beta, & others \end{cases} \quad (3)$$

$A(\alpha)$ is an affine mapping from [0,1) to a sufficiently small interval around 0.5, and $0 < \beta < 1$ is a constant. In this context, G_α^n represents $\overbrace{G_\alpha \circ G_\alpha \circ \overset{n}{\dot{G}_\alpha} \circ \dots \circ G_\alpha}$.

2.1. The Description of the Algorithm

The algorithms consist in several steps which are described below. From my personal point of view, the algorithm is not the most optimal one, but he do the job in our case very well.

- Primarily, a message M with arbitrary length is padded until the size of the message becomes multiple of l. Then we broke the padded message into blocks $M_0, M_1, \dots, M_{r-1}$, each having l bits (where the last l-bit block M_r represents the length of M in bits).
  We change the l-bit block $M_i = (p_{i1}p_{i2} \dots p_{il})$ (where $i = 0,1, \dots, r$) into a pair of binary fractions $(m_i, \widetilde{m_i})$, where $m_i = 0.p_{i1}p_{i2} \dots p_{il}$ and $\widetilde{m_i} = 0.p_{il}p_{i(l-1)} \dots p_{il}$.
- Let $i = 0$ and input a pair of common initial binary fractions(s_0, t_0), where $s_0 = 0.s_{01}s_{02} \dots s_{0l}$, $t_0 = 0.t_{01}t_{02} \dots t_{0l}$, in which case they necessarily must to be kept secret.
- Another step is the hashing of the quadruple binary fractions $(s_i, t_i, m_i, \widetilde{m_i})$ with a hash round function Hinto a pair of binary fractions (s_{i+1}, t_{i+1}). The symbols $\boxplus$ and $\otimes$ are defined in the following way, to describe the hash round function:

$$x \boxplus y = (x + y)(mod\ 1) \quad (4)$$

$$s \otimes y = G_{\min(x,y)}(\max(x, y)) \quad (5)$$

$G_\alpha^n(x_0)$ is used as a block cipher in the hash round function H, and in $G_\alpha^n(x_0)$, $x_0 = s_i \boxplus m_i, \alpha = t_i \boxplus \widetilde{m_i}$. In this context, let

$$z_i = G_\alpha^n(x_0) = G_{t_i \boxplus \widetilde{m_i}}^n(s_i \boxplus m_i) \quad (6)$$

The hash round function can be divided into two hash round functions H_1, H_2, equation (7) and (8):

$$H_1: t_{i+1} = s_i \boxplus z_i \quad (7)$$

$$H_2: s_{i+1} = t_i \otimes (\widetilde{m_i} \boxplus z_i) \quad (8)$$

- Let $i = i + 1$. If $i \leq r$ then return to previous step.
- The **Performance analysis** of the pair of binary fractions ($s_{r+1} = 0.a_1a_2 \ldots . a_l \ldots, t_{r+1} = 0.b_1b_2 \ldots b_l \ldots .$) which are the output of the last iteration concludes that the output $2l$-bit hash value $h = (a_1a_2 \ldots a_lb_1b_2 \ldots b_l)$.
- An important role in the hash round function plays Statistical analysis of $G_\alpha^n(x_0)$. Its two parameters α, n determine its characteristics, given that it evolved from the chaotic tent.

First test – Uniform distribution

In [11], the author proves the following:

1. For any given $0 \leq \alpha < 1$, the distribution of $x_1 = G_\alpha(x_0)$ for randomly chosen $0 < x_0 < 1$ is the standard uniform distribution;

2. For any $0 \leq \alpha < 1$ and n, the distribution of $x_n = G_\alpha^n(x_0)$ for randomly chosen $0 < x_0 < 1$ is the standard uniform distribution $U(0,1)$.

Second test – The determination of the minimum number of iterations

In this context, [11] has proved that, if the number of iterations $n \geq 73$, the distribution of $G_{\alpha+\Delta\alpha}^n(x_0)$ for even tiny $\Delta\alpha$ is independent from the distribution of $G_\alpha^n(x_0)$. In this case, we chose the minimum number of iterations in this hash algorithm to be $n = 75$.

In this section we will make a quick review of the most important work papers regarding the chaotic hash functions. We will focus on the construction of chaos-based hash algorithms. Trying to integrate chaos into hash functions represents a promising direction which attracts more and more attention. It's very interesting to see how, by borrowing some of the classical cryptography, we can see a summary of the instructions regarding the securing construction of chaos-based hash function which is beneficial for the design of hash function on chaos for the next future. The model behind the chaotic environment consists in setting the initial value and the parameter as the algorithm secret key and start iteration. Many of the chaotic models, until now, have been introduced into hash function construction, such as simple chaotic map, complex chaotic map, composite map and chaotic neural networks.

3. The proposed enrollment scheme

The protection scheme proposed in this article is starting from the idea of creating a strong and powerful scheme for enrolment in biometric systems. Each message that is exchanged in this scheme is hashed with the algorithm described above, in section 2.

One of the key aspects of this scheme is the generator of the session key. The generator is based on a chaotic phenomenon, composed from Rossler map and a pseudo random bit generator [11]. Rossler map is based on continuous time domain and have real space domain. The combination between the Rossler and [2] in the generator increase the power and the resistance on different cryptanalysis attacks, making a strong key and hard to recover. Other chaotic function properties and number generators are presented in [3,4].

The generator is based on the following general steps:

1. Generate three numbers *a, b,* and *c*. These numbers are used to calculate the Rossler differential equation.
2. These numbers are converted into binary form, using the *Binary Conversion* function from the session key algorithm.
3. The binary values are XORed ($x \oplus y \oplus z$) and the final value is passed to the *Session Key Stream Mapping* function.
4. A pseudo number is computed and XORed with the result from the session key stream mapping. This way, the key becomes more resistant to different attacks.
5. The session key is passed to the enrolment scheme which is used together with the messages that are exchanged between the components of the scheme. The session key generated by this algorithm is used in the hash function presented above. As a mathematical representation of how the algorithm in question is introduced in the enrolment scheme, let's consider that the *message component* represents actually the message *M* from the algorithm and *SK* is the session key generated above.

$$DMHashAlg(SK, message\ component) \quad (9)$$

The key {*x, y, z*} is composed of three sub-key's, in which each sub key is of the form sub-key = {initial value, *r* value}. The *r* value represents a positive number, which helps to identifying if there is a possibility to make a reproduction of the session key. First value, *x* value, is used for generating the 1st key stream, the second value, *y* value, is used for generating the 2nd key stream and the third key, *z* key, is used for generating the 3rd key stream. 2nd key stream `and 3rd key stream are illustrated in the next equation:

$$sessionkey_{stream} = b_1, b_2, \dots, b_n \text{ where } b_n \in [0,1] \quad (10)$$

We have three general types of behaviour for *r* value:

1. If *r* is situated between 0 and 1, the session key is easy to reproduce in proportion of 30%;
2. If *r* is situated between 1 and 2, the session key is difficult to reproduce in proportion of 50%;
3. If *r* is situated between 2 and 3, the session key is difficult to reproduce in proportion of 70%;

The session key stream mapping module for generating the 1st key stream is shown in the next equation:

$$sessionkey_{stream} = b_1, b_2, \dots, b_k, b_{k+1}, \dots, b_{2k}, b_n \text{ where } b_n \in [0,1] \quad (11)$$

We can observe in figure 3 of the chaotic session key representation, that the session key is passed also the logistic map [1, 7, 10], which is given by,

$$x_{n+1} = r \cdot x_n[1 - x_n],\ 0 < x < 1 \quad (12)$$

where x_0 $(n = 0)$ represents the initial value, *r* represents the bifurcation parameter and is dependent on the value *r,* x_0 represents the dynamics of the generated chaotic sequence which can change dramatically for $3.57 < r \leq 4$, this sequence being founded as non-periodic and non-converging [10]. The function for probability density of logistic map is symmetric and proven in [7] and the binary representation is done using the following equation:

$b_i = 0 \; for \; x_n < 0.5 \; and$ (13)

$b_i = 1 \; for \; x_n \geq 0.5$

where $0 < i < n$, n are the length of the chaotic sequence.

The XOR operation between logistic map and session key stream mapping represents a strengthening of the key. This way, the confusion property means that the relationship between the plaintext and the cipher text is complex and involved as much as possible.

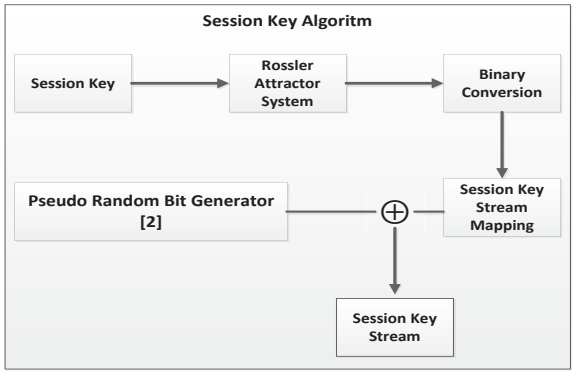

Fig. 3. Session Key Algorithm

In figure 3, we can observe that the permutation functions are applied on all the components that play a role in the enrolment scheme. The permutation function will allow the original value to be re-arranged in such a way that will be very difficult to understand something from permuted value.

Chaotic system is deterministic and sensitive to the initial values. According to this feature, it has complex active action, which can be used to protect data content. For example, the random sequence produced by chaotic phenomenon can be used to encrypt data in secret communication. This property makes the initial value suitable for the key that controls the data encryption or decryption. The one-way property makes neural network a suitable choice for hash function also. The model described in figure 4 stores the encrypted Biometric Template using Session Key. This approach is nothing but bio hashing or salting approach using key. A session key is an encryption and decryption key that is randomly generated to ensure the security of a communications session.

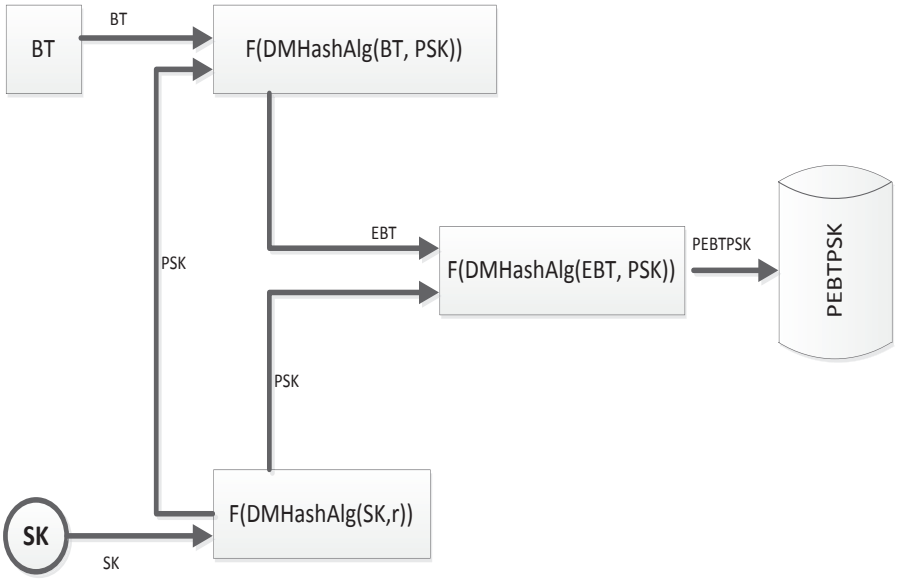

Fig. 4. Proposed Enrolment Scheme.

Notations:
BT – biometric template
SK – represents the session key, which is created using two chaotic function, Rossler and Logistic map. The process of creating the session key is presented above.
PSK – permuted session key, which is used to generate the expanded permuted transformation of the session key(SK), $F(SK)$ is a function for permutation which is used above the Davies-May hash function.
EBT – represents the biometric template which is encrypted. To generate biometric template encrypted, the Davies-May hash function $F(DMHashAlg(BT, PSK))$ is used. The hash function $F(DMHashALg(BT, PSL))$ is using simple XOR function and both functions $F(DMHashALg(SK, r))$ and $F(DMHashAlg(EBT, PSL))$ functions use beside of the hash function also permutations of bits by *SK, EBT* and ESK.
PEBTPSK – The permuted encrypted BT and also the permuted session key (SK) will be used to generate the final concatenated biometric template $F(DMHashAlg(EBT, PSL))$. The same session key and the functions are used for decryption of the encrypted biometric template.

The affiliation of this hash algorithm to the iterated hash function is obvious, so that if and attack occurs on the hash round function, it implies an attack of the same type, in other words with the same computation complexity, on the iterated hash function. In the algorithm we consider $G_\alpha^n(x_0)$ as a "block cipher", while the two sub-round functions H_1, H_2 resemble to the well-known Davies-Mays scheme. This can lead to believe that H_1, H_2 have almost similar complexities regarding the possible attacks as Davies-Mays scheme, which means that this hash function has at least the same computational security against the above mentioned attack.

In order to point out the need of usage of analysis models for principal components and cluster analysis in the segmentation process of consumers, we have analysed the behaviour, need of loans, of the bank's customers. Starting from a database of clients that have requested consumer loans through the two methods of analyse we have followed the main characteristics of the loan consumer; the whole process is directed on simplifying variables that describe the loan client's profile and identification of segments, considering that a narrower client identification can have its benefits as well as regarding business opportunity, that would lead in selling a credit card, and as for reducing costs.

References

[1]. Ratha, N.K., J.H. Connell, and R.M. Bolle (2001), *Enhancing security and privacy in biometrics-based authentication systems*, IBM Systems Journal, vol. 40, no. 3.

[2]. Marina Hentea (2007), *Intelligent system for information security Management: architecture & design issues*, Informing science & information technology vol. 4.

[3]. Radu Boriga, Ana Cristina Dascalescu (2011), *A New Method for Improving the Cryptographic Performances of the Ten Map*, MegaByte Journal, volume 2.

[4]. Radu Boriga, Ana Cristina Dascalescu (2010), *A Method for Increasing the Randomness of Lagged Fibonacci Generators*, Ovidius University Annals: Economic Sciences Series, Vol. 10, No. 2, pp. 51-55.

[5]. Christos K. Dimitriadis (2004), *Biometric risk and controls*, Information Systems control Journal Vol. 4.

[6]. Chen, Z., Huang, Y. (2001), *Chaotic one way hash function*, Communications Technology volume 7, pp. 96-98.

[7]. Liu, J., Xie, J., Wang, P. (2000), *One way hash function construction based on chaotic mappings*. Journal of Tsinghua University (Sci. & Tech.) 40(7), 55-58.

[8]. Wang, X., Wang, Y., Wang, M. (2001), *The collision problem of one kind of methods for constructing one-way Hash function based on chaotic map*. Acta Physica Sinica 55(10), 5048-5054.

[9]. Uludag U, Jain AK (2004*) Attacks on Biometric Systems: A Case Study in Fingerprints*. Proceedings of SPIE-EI 2004, Security, Steganography and Watermarking of Multimedia Contents VI 5306:622{633.

[10]. Uludag U, Pankanti S, Prabhakar S, Jain AK (2004) *Biometric Cryptosystems: Issues and Challenges*. Proc. IEEE, 92:948{960.

[11]. Yi X (2005), *Hash function based on the chaotic tent map*. Transactions on Circuits and Systems 52(6), 354-357.

[12]. Vinod Patidar, K.K. Sud (2009), *A Pseudo Random Bit Generator Based on Chaotic Logisitic Map and its statistical Testing*, Informatica volume 33, pp. 441-452.

[13]. Habutsu, T., Nishio, Y., Sasase, I. & Mori S. (1991), *A secret key cryptosystem by iterating a chaotic map. Advances of Cryptology* – EUROCRYPT'91, Lecture Notes in Computer Science, vol. 547, pp. 127–140 (Spinger-Verlag).

[14]. Pareek N. K., Patidar Vinod and Sud K. K. (2005), *Cryptography using multiple one-dimensional chaotic maps*. Communications in Nonlinear Science and Numerical Simulation, vol. 10, pp. 715-723.

[15]. Pareek N. K., Patidar Vinod and Sud K. K. (2006), *Image encryption using chaotic logistic map*. Image and Vision Computing, vol. 24, pp. 926-934.

[16]. Fridrich J. (1998), *Symmetric ciphers based on two dimensional chaotic maps*. International Journal of Bifurcation Chaos, vol. 8, pp. 1259–1284.

[17]. Tang, G., Liao, X. & Chen (2005), *Y. A novel method for designing S-boxes based on chaotic maps*. Chaos Solitons Fractals, vol. 23, pp. 413–419

[18]. Kocarev, L., Jakimoski, G., Stojanovski, T. & Parlitz (1998), *U. From chaotic maps to encryption schemes*. Proceedings IEEE Int. Symposium Circuits and Systems (ISCAS'98), vol. 4, pp. 514–517.

[19]. Guo, D., Cheng, L. M. & Cheng, L. L. (1999) *A new symmetric probabilistic encryption scheme based on chaotic attractors of neural networks*. Applied Intelligence, vol. 10, pp. 71–84.

[20]. Jakimoski G. & Kocarev L. (2001) *Chaos and cryptography: Block encryption ciphers based on chaotic maps*. IEEE Trans. Circuits Syst. I, vol. 48,pp. 163-169.

[21]. Menezes A.J., Oorschot P.C.V. and Vanstone S.A. (1997), *Handbook of Applied Cryptography*. CRC Press, Boca Raton.

[22]. Alvarez G. and Li S. (2001), *Some basic cryptographic requirements for chaos based cryptosystems*. International Journal of Bifurcation and Chaos, vol. 16, pp. 2129-2151.

[23]. Pareek N. K., Patidar Vinod and Sud K. K. (2003) *Discrete chaotic cryptography using external secret key*. Physics Letters A, vol. 309, pp. 75-82.

[24]. Boccaletti S., Grebogi C., Lai Y.-C., Mancini H. and Maza D. (2000), *The control of chaos: theory and applications*. Phys. Reports, vol. 329, pp. 103-197.

[25]. Schuster H. G. (Ed.) (1999) Hand book of chaos control. Wiley-VCH Verlag, Weinheim, Germany.

[26]. Kocarev, L., Halle, K. S., Eckert, K., Chua, L. O. & Parlitz, U. (1992) *Experimental demonstration of secure communications via chaotic synchronization*. International Journal of Bifurcation Chaos, vol. 2, pp. 709-713.

[27]. Wu, C. W. & Chua, L. O. (1993) *A simple way to synchronize chaotic systems with applications to secure communications systems.* International Journal of Bifurcation Chaos, vol. 3, pp. 1619–1627.

[28]. Cuomo, K. M., Openheim, A. V. & Strogatz, S. H. (1993), *Synchronization of lorenz-based chaotic circuits with applications to communications*. IEEE Trans. Circuits Syst. II, vol. 40, pp. 626–633, 1993.
[29]. Morgul, O. & Feki, M. (1999) *A chaotic masking scheme by using synchronized chaotic systems*. Phys. Lett. A, vol. 251, pp. 169–176.