Chaos-based hash function (CBHF) for cryptographic applications

Article *in* Chaos Solitons & Fractals · October 2009

DOI: 10.1016/j.chaos.2009.02.001

Mohamed Amin
Helwan University
8 PUBLICATIONS 269 CITATIONS
SEE PROFILE

Osama S. Faragallah
Menoufia University
193 PUBLICATIONS 1,893 CITATIONS
SEE PROFILE

Ahmed Abd El-Latif
171 PUBLICATIONS 4,074 CITATIONS
SEE PROFILE

Contents lists available at ScienceDirect

Chaos, Solitons and Fractals

journal homepage: www.elsevier.com/locate/chaos

Chaos-based hash function (CBHF) for cryptographic applications

Mohamed Amin [a], Osama S. Faragallah [b], Ahmed A. Abd El-Latif [a,*]

[a] *Dept. of Mathematics & Computer Science, Faculty of Science, Menoufia University, Shebin El-Koom 32511, Egypt*
[b] *Dept. of Computer Science and Engineering, Faculty of Electronic Engineering, Menoufia University, Menouf 32952, Egypt*

ARTICLE INFO

Article history:
Accepted 6 February 2009

ABSTRACT

As the core of cryptography, hash is the basic technique for information security. Many of the hash functions generate the message digest through a randomizing process of the original message. Subsequently, a chaos system also generates a random behavior, but at the same time a chaos system is completely deterministic. In this paper, an algorithm for one-way hash function construction based on chaos theory is introduced. Theoretical analysis and computer simulation indicate that the algorithm can satisfy all performance requirements of hash function in an efficient and flexible manner and secure against birthday attacks or meet-in-the-middle attacks, which is good choice for data integrity or authentication.

1. Introduction

Cryptographic hash function is the basic technique for information security and plays an important role in modern cryptography. A hash function H takes a long string (or "message") of any length as input (x) and produces a fixed length string ($h = H(x)$) as output, sometimes termed a message digest or a digital fingerprint [1]. In addition, it satisfies the following properties [2]:

- The input can be of any length.
- The output has a fixed length.
- $H(x)$ is relatively easy to compute for any given x.
- $H(x)$ is one-way.
- $H(x)$ is collision-free.

Different from the conventional cryptography as MD5, and SHA-1, new directions in cryptography have been widely developed in the past decade [3], for example, chaos-based cryptography. Chaos is a deterministic process, which is ubiquitously present in the world. Because of its random like behavior, sensitivity to initial conditions and parameter values, ergodicity, and confusion and diffusion properties; chaotic cryptography has become an important branch of modern cryptography and has huge potential in protecting the assets [4]. An algorithm for one-way hash function construction based on iterating a chaotic map is introduced in this paper. Theoretical analysis and computer simulation indicate that the algorithm can satisfy all performance requirements of hash function in an efficient and flexible manner, which is good choice for data integrity or authentication.

The remaining of the paper is organized as follows. The algorithm of hash function is proposed in Section 2. Its performance is analyzed in Section 3. Finally, conclusions are drawn in Section 4.

[*] Corresponding author.
E-mail addresses: mamin04@yahoo.com (M. Amin), osam_sal@yahoo.com (O.S. Faragallah), ahmed_rahiem@yahoo.com (A.A. Abd El-Latif).

0960-0779/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.chaos.2009.02.001

2. The hashing scheme

2.1. Analysis of the tent map

As a simple introduction to one-dimensional nonlinear discrete dynamical systems, consider the tent map [5],

$$T(x) = \begin{cases} rx, & 0 \leqslant x < 0.5, \\ r(1-x), & 0.5 \leqslant x \leqslant 1. \end{cases} \tag{1}$$

Define an iterative map by

$$x_{n+1} = T(x_n), \tag{2}$$

where $0 \leqslant r \leqslant 2, x_n \in [0,1]$. The tent map is constructed from two straight lines, which makes the analysis simpler than for truly nonlinear systems. The graph of the tent map function is plotted by MATLAB software and is given in Fig. 1.

Although the form of the tent map is simple and the equation involved is linear, for certain parameter values, this system can display highly complex behavior and even chaotic phenomena [5]. Fig. 2 shows the simulation of the tent map by MATLAB software. The simulation results include three portions. It can be described as follows: let the initial value: $x_0 = 0.3$, loop iterations = 30, 100, 1000, and 100,000. The parameter r can be divided into three segments, which can be examined by experiments on the following conditions:

- When $r \in [0,1]$ as shown in Fig. 2a, the calculation results come to the same value after several iterations without any chaotic behavior.
- When $r \in [1, \sqrt{2}]$, the system appears periodicity, as shown in Fig. 2b.
- While $r \in [\sqrt{2}, 2]$, it becomes a chaotic system with periodicity disappeared as shown in Fig. 2c.

Also, we use MATLAB software to graph the bifurcation diagram of the tent map as shown in Fig. 3.

2.2. The proposed chaos-based hash function (CBHF)

Here, we propose a chaotic hash algorithm based on chaotic tent map. An overview of CBHF module is depicted in Fig. 4. Chaotic tent map is used as the core of CBHF to achieve security requirements. The CBHF encodes the block length M_{n-1} (1024 bits) into fixed length H_n (128 bits). First, the message M is appended to multiples of 1024 bits. That is, one ‘1’ bit and some ‘0’ bits are appended to M. Second, it is partitioned into n blocks: $M_0, M_1, \ldots, M_{n-1}$. These, blocks are encoded a number of times until we reach the message digest H_n as shown in Fig. 5, which presents the general construction proposal. The final hash value is computed as

$$H_n = K_{n-1} \oplus H_{n-1}. \tag{3}$$

In general,

$$H_n = (K_0 \oplus H_1) \oplus H_2 \oplus \cdots \oplus H_{n-1}, \tag{4}$$

where K_0 is the initial value of the tent map, and it consists of x_0, r_0.

$$H_1 = C(K_0), \tag{5}$$

C is the chaotic tent map and H_n is the messages digest.

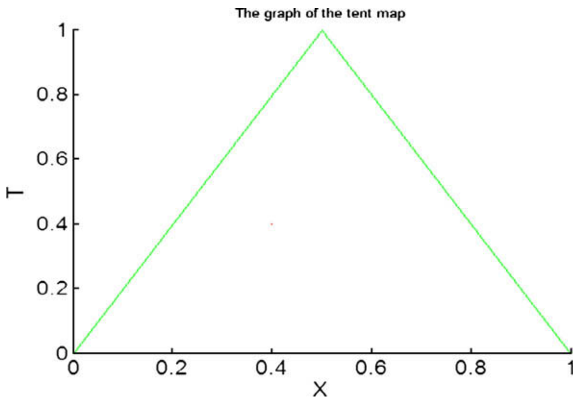

Fig. 1. The graph of the tent map.

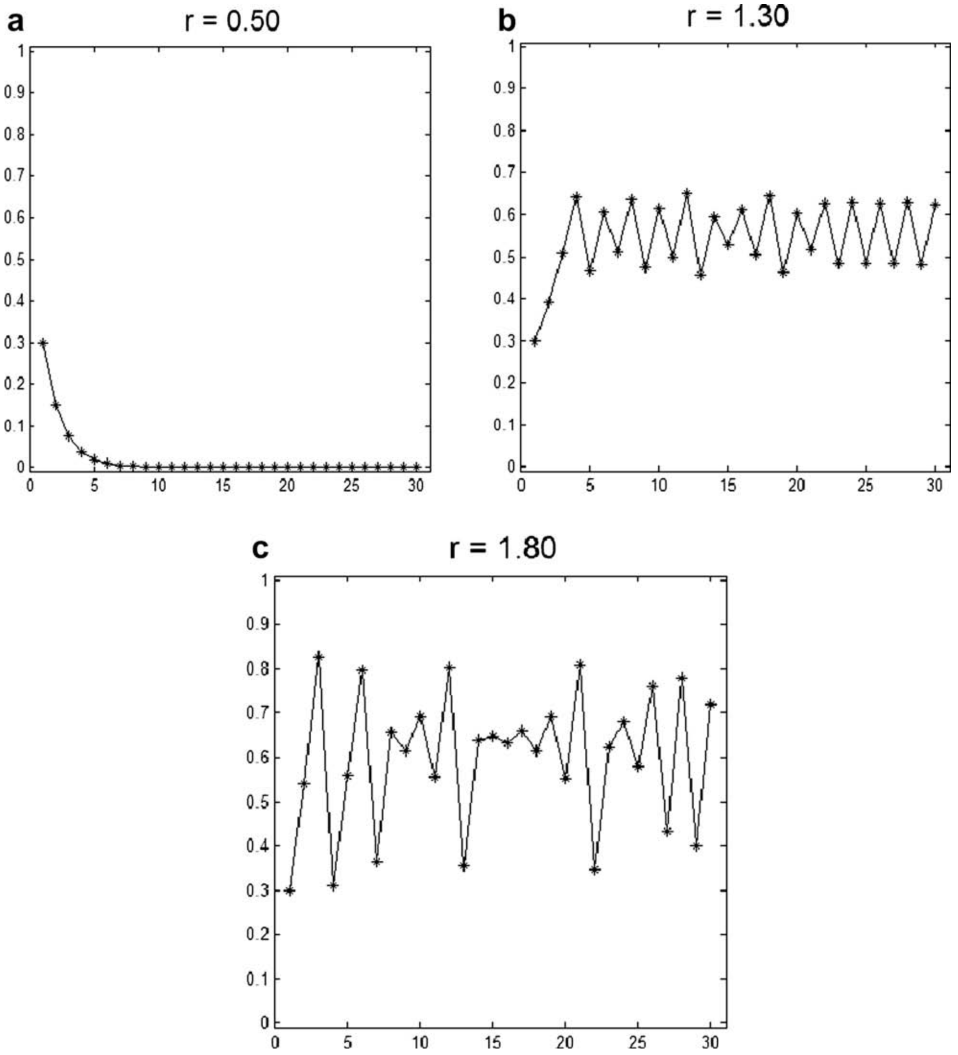

Fig. 2. (a) Iteration property when $r = 0.5$. (b) Iteration property when $r = 1.3$. (c) Iteration property when $r = 1.8$.

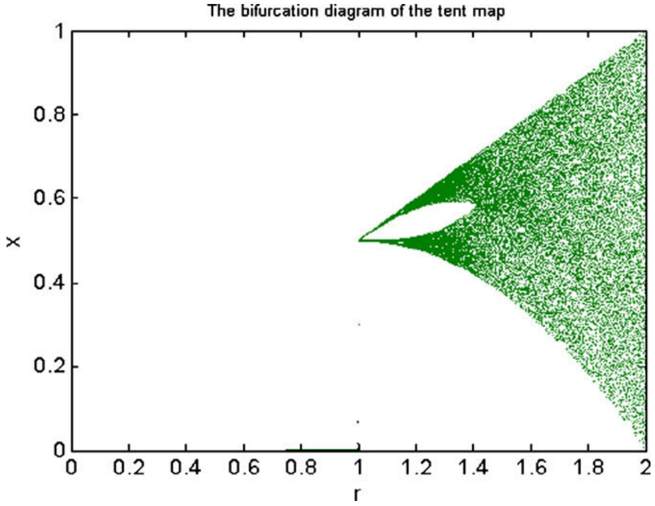

Fig. 3. The bifurcation diagram of the tent map.

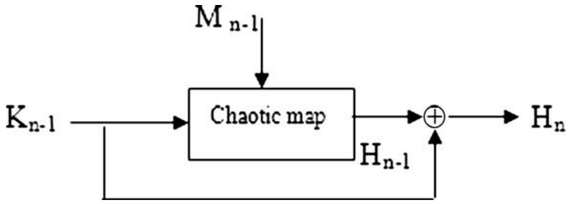

Fig. 4. Diagram of chaotic hash function.

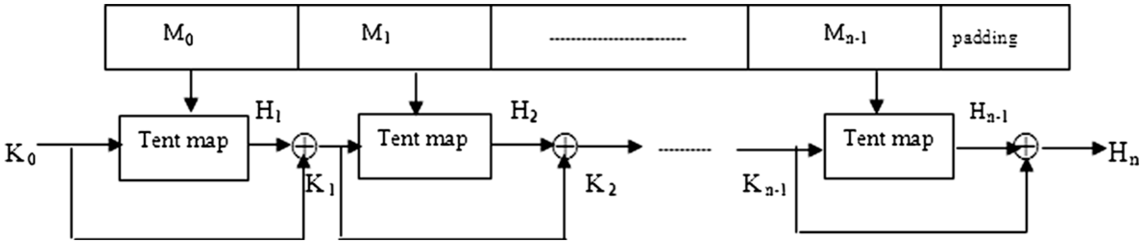

Fig. 5. Detail construction of CBHF.

3. Performance analysis

3.1. Hash result of messages

We use the proposed algorithm to do hash simulation under the following conditions:

Condition 1: The original message is "As the core of cryptography, hash is the basic technique for information security. Many of the hash functions generate the message digest through a randomizing process of the original message. Subsequently a chaos system also generates a random behavior, but at the same time a chaos system is completely deterministic".
Condition 2: Changes the first character A in the original message into C.
Condition 3: Changes the word technique in the original message into techniques.
Condition 4: Changes the full stop at the end of the original message into comma.
Condition 5: Replace the word chaos in the original message into disorder.
Condition 6: Adds the name Ahmed to the end of the original message.The corresponding hash values in hexadecimal format are:
Condition 1: D8F85A570935F72C59E6595FA2970E51
Condition 2: 5C85AF5F5023153FE25E15685E05DA7B
Condition 3: E5F6095BA47089CF23F45FA54FA14F54
Condition 4: B52F75DB0123E9F5943CF501286016FA
Condition 5: A574E95A6578A74CA657F21F6467A0D5
Condition 6: 0A693FD5FC5455E4450C45B45DF81F4E

The simulation result indicates that any least difference of the message will cause a substantial change in the final hash value.

3.2. Analysis of confusion and diffusion

Confusion and diffusion are two basic design criteria for cryptographic algorithms, including hash functions [6,7]. For the message digest in binary format, each bit is only 1 or 0. So, the ideal diffusion effect should be that any substantial changes in initial conditions lead to 50% changing probability of each bit.

We have performed the following diffusion and confusion test: the message is randomly chosen and hash value is generated. Then, a bit in the message is randomly selected and toggled and a new hash value is generated. Two hash values are compared with each other and the number of changed bit is counted as X_i. This kind of test is performed N (such as 128, 256, 512, 1024) times, and the corresponding distribution of changed bit number is shown as Fig. 6, where $N = 1024$.

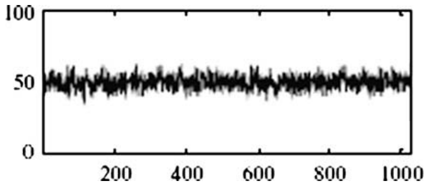

Fig. 6. Distribution of changed bit number.

Usually, four statistics are defined as follows:

$$\text{Mean changed bit number}: \quad \overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i, \tag{6}$$

$$\text{Mean changed probability}: \quad P = (\overline{X}/128) \times 100\% \tag{7}$$

$$\text{Standard variance of the changed bit number}: \quad \Delta X = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(X_i - \overline{X})^2} \tag{8}$$

$$\text{Standard variance}: \quad \Delta P = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(X_i/128 - P)^2} \times 100\%, \tag{9}$$

where, N is total statistic number, X_i changed bit number when the ith simulation is performed, X_i and P represent mean changed bit number and mean changed probability, respectively, ΔX and ΔP indicate the stability of diffusion and confusion. Through the tests with $N = 128$, 256, 512, and 1024, respectively, the corresponding data are listed in Table 1.

Based on the analysis of the data in Table 1, we can draw the conclusion: the mean changed bit number and changed percent is 63.84 and 49.88%, respectively, which are very close to the ideal value 64 and 50%. While ΔX and ΔP, indicating the stability of diffusion and confusion, is very little, which represent that the capability for diffusion and confusion is stable. The statistical effect guarantees that attacker absolutely cannot forge or reduce other plaintext–ciphertext pair from some known plaintext–ciphertext pairs.

3.3. Analysis of birthday attack

Birthday attack [1,8] is a typical attack method used to break a hash function. That is, to find a contradiction is similar to find two persons with the same birthday. Thus, for 64-length hash value, the attack difficulty is not 2^{64}, but much smaller 2^{32}. Considering of the practical computing ability, the hash value's length should be at least 128-bit, which keeps the attack difficulty above 2^{64}. Here, the proposed hash is 128-length, this keeps the scheme secure against this kind of attack.

3.4. Analysis of meet-in-the-middle attack

Meet-in-the-middle attack [1,8] means to find a contradiction through looking for a suitable substitution of the last plaintext block. If $M = [M_0, M_1, \ldots, M_{n-2}, M_{n-1}]$, the expected contradicted one is $M' = [M_0, M_1, \ldots, M_{n-2}, M'_{n-1}]$. That is, the attack process is just to replace M_{n-1} with M'_{n-1} and keep H_n unchanged, as is shown in Fig. 7. Because K_{n-1} is not known, and the chaotic map's parameter are not known. The attackers may attempt to use many plaintext-key-hash triples, but they cannot obtain K_{n-1} because it is in close relation with the key and the previous plaintext blocks. If $n = 0$, there is only one plain-block, which has been analyzed above. Thus, it is difficult to break the hash function with meet-in-the-middle attack.

3.5. Analysis of speed and flexibility

First, the proposed algorithm has to do little computations, one-dimensional chaotic tent map is chosen in our algorithm, which makes the analysis simpler than for truly nonlinear systems. On the one hand, its dynamical property is enough for the

Table 1
The statistical performance of our algorithm.

	$N = 128$	$N = 256$	$N = 512$	$N = 1024$	Mean
X_i	64.32	63.87	63.74	63.43	63.84
P (%)	50.25	49.90	49.80	49.55	49.88
ΔX	5.43	5.53	5.64	5.72	5.58
ΔP	4.23	4.34	4.53	4.41	4.3775

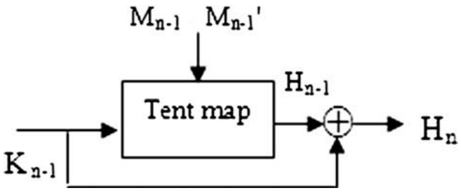

Fig. 7. Meet-in-the-middle attack.

algorithm security; on the other hand, its structure is so simple that only multiplication, and several additions (subtractions) are operated, which reduces the algorithm complexity and guarantees the high efficiency.

Second, although our algorithm aims at unkeyed hash function, it also can be used to construct keyed hash function simply treating x_0 and the initial values of bit sequences as secret keys. Through simply modifying the number of subintervals and the size of bit sequence, the length of the final hash value will be easily changed. Compared with the conventional hash algorithm such as MD5 with fixed 128-bit length, the proposed algorithm can adapt to the actual demand better.

4. Conclusion

In this paper, an algorithm for one-way hash function construction based on chaos theory is investigated. A chaotic tent map is chosen, for certain parameter values, this system can display highly complex behavior and even chaotic phenomena. The hash value is obtained by iterating the tent map. Owing to making full use of properties of chaos, such as ergodicity, nonlinearity and mixing the scheme can satisfy all performance requirements of hash function in an efficient and flexible manner. Theoretical analysis and computer simulation indicate that our algorithm can resist birthday attack and meet-in-the-middle attack, which is good choice for data integrity or authentication.

References

[1] Menezes A, van Oorschot P, Vanstone S. Handbook of applied cryptography. New York: CRC Press; 1996.
[2] Stallings W. Cryptography and network security: principles and practice. 4th ed. Englewood Cliffs (NJ): Prentice-Hall; 2005.
[3] Peng Fei, Qiu Shui-Sheng. One-way hash functions based on iterated chaotic systems. In: IEEE conference proceedings: communications, circuits and systems, 2007. ICCCAS 2007. International conference on 11–13 July; 2007. p. 1070–74.
[4] Muhammad Khurram Khan, Jiashu Zhang, Xiaomin Wang. Chaotic hash-based fingerprint biometric remote user authentication scheme on mobile devices. Chaos, Solitons & Fractals 2008;35(3):519–24.
[5] Stephen Lynch. Nonlinear discrete dynamical systems. In: Dynamical systems with applications using MATLAB. Boston: Birkhauser; 2004 [chapter 2].
[6] Baris Coskun, Nasir Memon. Confusion/Diffusion capabilities of some robust hash functions. In: IEEE conference proceedings: information sciences and systems, 2006 40th annual conference on 22–24 March; 2006. p. 1188–93.
[7] Di Xiao, Xiaofeng Liao, Shaojiang Deng. One-way hash function construction based on the chaotic map with changeable-parameter. Chaos, Solitons & Fractals 2005;24(1):65–71.
[8] Shiguo Lian, Jinsheng Sun, Zhiquan Wang. Secure hash function based on neural network. International Journal of Neurocomputing 2006;69(16–18):2346–50.