INT J COMPUT COMMUN, ISSN 1841-9836
9(1):24-36, February, 2014.

The Parallel One-way Hash Function Based on Chebyshev-Halley Methods with Variable Parameter

M. Nouri, M. Safarinia, P. Pourmahdi, M.H. Garshasebi

Mahdi Nouri
Department of Electrical Engineering,
Iran University of Science and Technology (IUST), Tehran, Iran
Mnouri@mtu.edu

Mahyar Safarinia, Payam Pourmahdi
Electrical Engineering Department
A.B.A Institude of Higher Education, Qazvin, Iran
Mahyar1986@gmail.com, payamict67pm@gmail.com

Mohammad Hossein Garshasebi
Communication Engineering Department
Basir Institude of Higher Education, Qazvin, Iran
E-mail: Mh.garshasebi@chmail.ir

Abstract: In this paper a parallel Hash algorithm construction based on the Chebyshev Halley methods with variable parameters is proposed and analyzed. The two core characteristics of the recommended algorithm are parallel processing mode and chaotic behaviors. Moreover in this paper, an algorithm for one way hash function construction based on chaos theory is introduced. The proposed algorithm contains variable parameters dynamically obtained from the position index of the corresponding message blocks. Theoretical analysis and computer simulation indicate that the algorithm can assure all performance requirements of hash function in an efficient and flexible style and secure against birthday attacks or meet-in-the-middle attacks, which is good choice for data integrity or authentication.
Keywords: Hash function; Chebyshev-Halley methods; Two-dimensional coupled map lattices; Spatiotemporal chaos; Chaotic nonlinear map; variable parameter.

1 Introduction

By the rapid development of Internet, the security is essential for modern communication [1][2].

Hash functions play an essential role in many areas of cryptographic applications such as digital signature, random number generation, data source authentication, key update and derivation, message authentication code, integrity protection and malicious code recognition and, etc. Generally, hash functions can be classified into two categories [3,4]: unkeyed hash functions with a single input parameter (a message) for data integrity, and keyed hash functions, usually known as message authentication code (MAC), with two distinct inputs. Conventional hash functions, such as MD5 and SHA, involve logical operations or multi-round iterations of some available ciphers. Although each step of the former is simple, the number of processing rounds could be massive even if the message is very short. Recently, spatiotemporal chaos has been attracting more and more interests among researchers engineering. After the conventional Hash function such as MD5, SHA was successfully attacked, the research on the design of secure and efficient Hash function remains a research hotspot. Compared with simple chaotic maps, spatiotemporal chaos possesses two additional advantages for cryptographic purpose. Due to the finite computing precision, chaotic orbits will eventually become periodic.

In particular, the period of chaotic orbits generated by a system with a large number of chaotic coupled oscillators is too long to be reached in practical communications. The period of spatiotemporal chaos is longer than that of simple chaotic maps [7]. Therefore, periodicity can be practically avoided in spatiotemporal chaotic systems [8,29,30]. Chaotic systems have vital characteristics like ergodic, mixed

and sensitive which are so important in this area [3,5]. Some algorithms for one-way Hash function based on chaotic map have already been brought forward [11,12]. Most algorithms for chaos-based hash function proposed by existing articles are based on dissipative chaotic systems. But the dissipative chaotic systems can lead to many hidden threats in the practical application for secure communication because of their potential warning. When the dissipative chaotic systems were used in the application of encryption, if the attacker gets a continuous of plaintext-ciphertext pairs, and the length of ciphertext meets certain conditions, the attacker cannot predict by reconstructing through ciphertext without attractor structure for the conservative chaotic systems [5,13,14,31,32]. So the conservative chaotic systems with high security are ideal model in cryptography application.

In this paper the aim is to design an unkeyed hash function based on spatiotemporal chaos, which has high efficiency. In this algorithm the entire message blocks perform some rounds of iteration through Standard map. One output of the iterations of each round determines one of the initial values of the next round is the output of previous block. The method of this design enhances the diffusion of Hash function, and overcomes the inherent defects of dissipative chaotic systems. So the Hash algorithm has a higher security. Theoretical analysis and computer simulation show that this algorithm has good effect in Anti-conflict and Avalanche effect. This algorithm is easy to express and to satisfy all the performance requirements of Hash function in an efficient and flexible manner.

2 Feasibility of Hash Function Construction Based on Proposed

2.1 Nonlinear Equations Based on Newtons Method

Non-linear equations are an important and basic method [15], which converges quadratically. A family of third-order methods, called Chebyshev Halley methods [16], is written as

$$x_{n+1} = x_n - \frac{f(x_n)}{f\prime(x_n)} \tag{1}$$

where

$$L_f(x_n) = \frac{f\prime\prime(x_n) f(x_n)}{f\prime(x_n)^2} \tag{2}$$

This family includes the classical Chebyshevs method (CM), ($\alpha = 0$) , Halleys method (HM) ($\alpha = 1/2$) and Super-Halley method (SHM) ($\alpha = 1$)) (for the details of these methods, see [17-18] or a recent review [19]).In order to improve the local order of convergence, Grau and Daz-Barrero[20] propose an improvement of Chebyshevs method with fifth-order convergence

$$\tilde{x}_{n+1} = x_n - \left(1 + \frac{f\prime\prime(x_n)(f(x_n) + f(x_{n+1}))}{2f\prime(x_n)^2}\right) \frac{f(x_n) + f(x_{n+1})}{f\prime(x_n)} \tag{3}$$

Analysis of Convergence shows that the error convergence can be modeled with sixth order of error [10].

$$\tilde{e}_{n+1} = \left[(6 - 4\beta)c_2^2 - 3c_3\right]\left[2(1-\alpha)c_2^2 - c_3\right] e_n^5 + \mathrm{O}\left(e_n^6\right) \tag{4}$$

2.2 The Chebyshev-Halley Method

The family includes the classical Chebyshevs method (CM) ($\alpha = 0$) , Halleys method (HM) ($\alpha = 1/2$) and Super-Halley method (SHM) ($\alpha = 1$) (for the details of these methods, see [9,10] or a recent review [11]).Here the logistic map is taken as the local map, given as

$$f(x) = \mu x(1 - x) \tag{5}$$

Where $x_i, y_i \in [0, 1]$ and μ , α are the controlled variables. The map is nonlinear and the parameter ensures that the map runs in a chaotic status .The chaotic nonlinear map also has the same belongings to proposed map that are fit for composing Hash function. The form of the map is complicated and the equation involved is nonlinear. Figure 1 shows the simulation of the chaotic nonlinear map iterating 64

times with the initial values $x_0, y_0 = 0.3423$ and parameter $\mu = 60.5$. The map has some properties, which are appropriate for constructing the Hash function, such as initial value sensitivity and parameter sensitivity, with variable parameter valued in the interval (0,120). Figure 1 & 2 displays the chaotic iteration property with variable parameter first Iteration valued in the interval (0,64), which initial values are $x_0, y_0 = 0.3423$.

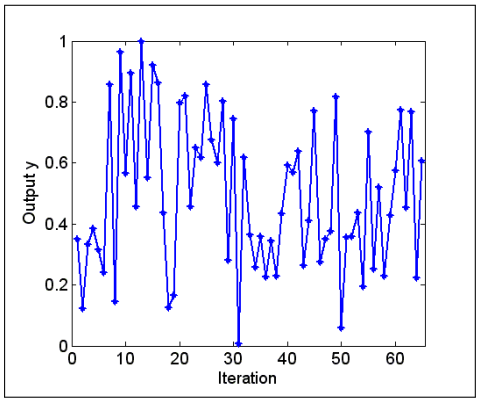


Figure 1: b

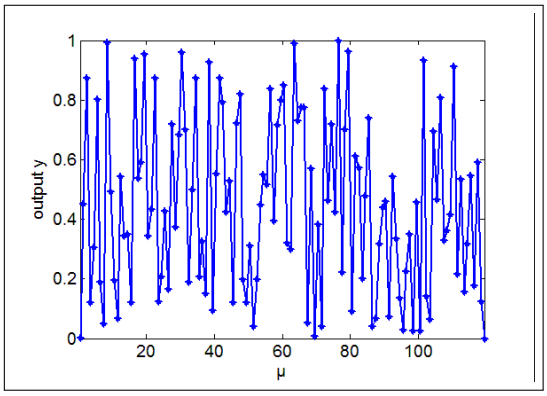


Figure 2: b

Figure 1 & 2 : Iteration property with changeable parameter First Iteration and μ and $x_0, y_0 = 0.3423$

3 Hash Function Based on Standard

Message expansion is significant and necessary; because it greatly improves the sensitivity of each bit in original message to the final Hash value [14]. The plaintext is an arbitrary message that is conveyed in a matrix M, for simple enlightenment of the extended message. Assume that M is a $N \times 16$ plain message matrix, each element with a size of 32 bits.

1) Padding the message: The purpose of padding is to ensure the padded message being a multiple of 128 bits. Suppose the total length of message is W bytes, compute d = (W mod 64), $0 \leq d \leq 12$. Pad as follows: if $12 \leq d < 16$ then pad 12d bytes, otherwise pad 28d bytes, the bytes been padded come from the head of message. The last four bytes are padded with message length. This method ensure at least one byte head of message been padded.

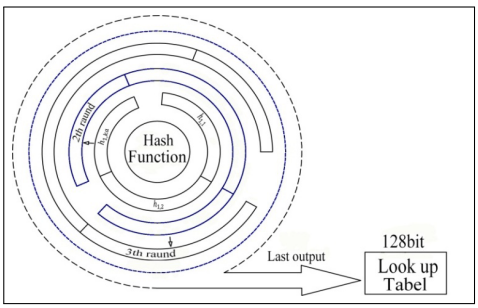


Figure 3: a

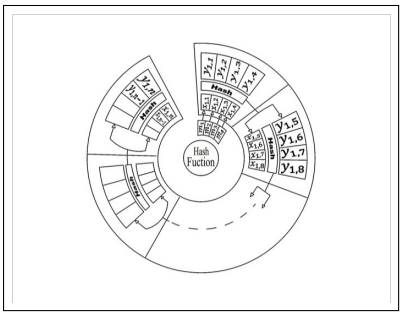


Figure 4: b

2) Parsing the padded message : The padded message is parsed into N 128-bit blocks, $M_1, M_2, \ldots, M_N$, $M_j (1 \leq j \leq N)$ is parsed into four 32-bit words

$$M_j = [m_{j,1}, m_{j,2}, m_{j,3}, m_{j,4}] \tag{6}$$

Figure 3 & 4 ; The whole algorithm ; a) All iterations of hash function structure . b) First iteration of hash function .

Input of the algorithm can have an input of arbitrary length output of the algorithm has a fixed length of 128 bits. Give a message M with length L. Take each letter of M as a plaintext block. Transform each plaintext block to the corresponding ASCII numbers; the ASCII numbers create the x_j, y_j which are the inputs of chaotic nonlinear map. Compression function inputs consider 16 lattice spaces. Let the initial iterative value of these inputs are:

$$\begin{cases} x_{j,1} = \left(\frac{m_{j,1}}{10^3}\right) + \left(\frac{m_{j,2}}{10^6}\right) + \left(\frac{m_{j,3}}{10^9}\right) + \left(\frac{m_{j,4}}{10^{12}}\right) \\ x_{j,2} = \left(\frac{m_{j,5}}{10^3}\right) + \left(\frac{m_{j,6}}{10^6}\right) + \left(\frac{m_{j,7}}{10^9}\right) + \left(\frac{m_{j,8}}{10^{12}}\right) \\ x_{j,3} = \left(\frac{m_{j,9}}{10^3}\right) + \left(\frac{m_{j,10}}{10^6}\right) + \left(\frac{m_{j,11}}{10^9}\right) + \left(\frac{m_{j,12}}{10^{12}}\right) \\ x_{j,4} = \left(\frac{m_{j,13}}{10^3}\right) + \left(\frac{m_{j,14}}{10^6}\right) + \left(\frac{m_{j,15}}{10^9}\right) + \left(\frac{m_{j,16}}{10^{12}}\right) \end{cases}$$

From $\{x_i\}$ and $\{y_i\}$, 4 groups of (y_i) can be reached. Determine the 32-bits Hash value by the position of the coordinates (y_i) falling into the region of [0, 1), then, finally, juxtaposes these bits from left to right to get a 128-bit hash value.[28]

TABLE 1
Algorithm for generating the hash

	Step		Operation
1	$q \leftarrow 1, j \leftarrow 1, i \leftarrow 2q-1$ go to Step 2	5	$i_1 \leftarrow 23, x'_{j,2ka_{(j)+2}} \leftarrow f$ *if* $f_{1,p-1} = 1 i_p \leftarrow (1.1)^p + [i_{p-1}]$, *elseif* $i_p \leftarrow i_{p-1}, p+1 \leftarrow p$ *end*
2	$k \leftarrow 1/\Pi$ $(x_{j,i+4}, y_{j,i+4}) \leftarrow f(x_{j,i}, y_{j,i})$, $i \leftarrow i+1$	6	*if* $p < 33$ go to Step 5
3	*if* $i \le 2k+2$ go to Step 2	7	ka_j=*101*, $ka_{j+1} \leftarrow ([a_{33}]) mod\ 23 + 61$
4	$p \leftarrow 2, k+1 \leftarrow k$, *if* $k \le ka$ go to Step 2	8	$d \leftarrow [1,4]$, $y_{j+1,d} \leftarrow y_{j,2ka_{j+2+d}}$

4 Performance Analysis

The proposed algorithm is used to perform hash simulation under the following five kinds of conditions:

Condition 1: A hash function is a fundamental building block of information security and plays an important role in modern cryptography. It takes a message as input and produces an output referred to as a hash value. Generally, hash functions can be classified into two categories: unkeyed hash functions for data integrity, and keyed hash functions, usually known as message authentication code (MAC).

Condition 2: Change the first character A in the original message to D.

Condition 3: Change the word function in the original message to functions.

Condition 4: Change the full stop at the end of the original message to a comma.

Condition 5: Add a blank space to the end of the original message.

The corresponding hash values in hexadecimal format are given as follows, followed by the equivalent number of different bits compared with the hash value obtained under Condition 1:

Condition 1 : 0484FCD104D3614104D5A185002832E9
Condition 2 : 3D87661B4F87F263FAEF3DAD5E7AAB
Condition 3 : 96A08482053A5537316D50181DFB7640
Condition 4 : D700EAE9701A4A55680C96A133F3B023
Condition 5 : 17F91FD2D4C22388C74C0811F5874687

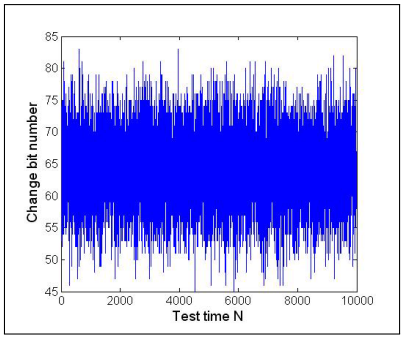


Figure 5 : Distribution of changed bit number B_i

This kind of test is performed N times, and the corresponding distribution of changed bit number is plotted in Figure 5 , where N = 10,000. Obviously, the changed bit number corresponding to 1 bit changed message concentrates around the ideal changed bit number, i.e., 64 bits. It indicates that the algorithm has very strong capability for diffusion and confusion.

4.1 Statistical Analysis of Diffusion and Confusion

Confusion and diffusion are two fundamental design criteria for encryption algorithms, including hash functions. Shannon introduced diffusion and confusion in order to hide message redundancy [18,19]. Hash function, like encryption system, requires the plaintext to diffuse its influence into the whole Hash space. This means that the correlation between the message and the corresponding Hash value should be as small as possible. Diffusion means spreading out the influence of a single plaintext symbol over many ciphertext bits so as hiding the statistical structure of the plaintext. Confusion means the use of transformations to complicate the dependence of ciphertext statistics on plaintext statistics. In the hash value in binary format each bit can be only 0 or 1. Therefore, the ideal diffusion effect should be that any tiny change in the initial condition leads to a 50% changes, probability of each bit. Usually six statistics are defined as follows:

Minimum changed bit number:

$$B_{min} = min\left(\{B_i\}_1^N\right) \tag{7}$$

Maximum changed bit number:

$$B_{max} = max\left(\{B_i\}_1^N\right) \tag{8}$$

Mean changed bit number:

$$\bar{B} = \frac{1}{N}\sum_{1}^{N} B_i \tag{9}$$

Mean changed probability:

$$P = \frac{\bar{B}}{128} \times 100 \tag{10}$$

Standard variance of the changed bit number:

$$\Delta B = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(B_i - \bar{B}\right)^2} \tag{11}$$

Standard variance:

$$\Delta P = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{B_i}{128} - P\right)^2} \times 100 \tag{12}$$

Where N is the total number of tests andB_i is the number of changed bits in the ith test. The following diffusion and confusion test has been performed that: A paragraph of message is randomly chosen and the corresponding hash value is generated. Then a bit in the message is randomly selected and toggled and a new hash value is generated. Finally, the two hash values are compared with each other. This kind of test is performed N times, and the corresponding distribution of changed bit number is plotted in Fig. 8, where N = 10,000. perceptibly the changed bit number corresponding to 1 bit changed message concentrates around the ideal changed bit number, i.e., 64 bits. It signifies that the algorithm has very strong capability for diffusion and confusion.The test results in N = 256, 512, 1024, 2048, 10,000 are listed in Table 1 respectively. Based on the results, the following conclusion was found that, the mean changed bit number $\bar{B}$ and the mean changed probability P are both very close to the ideal value 64 bits and 50% . While ΔB and ΔP are extremely small, which indicate the diffusion and confusion capability are very stable. Therefore a small difference in the plaintext will cause great changes in the hash value, which donates to the high plaintext-sensitivity of our hash function. This property is important in maintaining the secrecy against statistical attacks.

4.2 Collision Analysis

Birthday-attack

Collision resistance and birthday-attack are lying to each others roots. Both are derived from the probability problem that two random input data are found to hash to the same value. The results of the proposed algorithm are shown in TABLE 2:

TABLE 2
Statistical performance of proposed algorithm for 128 bits outputs with $\alpha = 0$ and $\mu = 60.5$

N	256	512	1024	2048	10,000
$\bar{B}$	63.79	64.14	64.0009	64.01	63.979
P(%)	49.84	50.11	50.0007	50.01	49.98
ΔB	4.10	4.39	4.31	4.34	2.38
ΔP(%)	3.21	3.43	3.37	3.39	1.86
B_{min}	49	49	49	44	44
$\boldsymbol{B_{max}}$	75	80	80	80	80

Collision resistance and birthday-attack are lying to each others roots. Both are derived from the probability problem that two random input data are found to hash to the same value. Given a function f the goal of the attack is to find two different inputs $f(x)$ such that $f(x_1) = f(x_2)$. Such a pair x_1, x_2 is called a collision. The method used to find a collision is simply to evaluate the function f for different input values that may be chosen randomly or pseudo randomly until the same result is found more than once. Because of the birthday problem, this method can be rather efficient. Specifically, if a function $f(x)$ yields any of H different outputs with equal probability and H is sufficiently large, then we expect to obtain a pair of different arguments x_1, x_2 with $f(x_1) = f(x_2)$after evaluating the function for about 1.25H different arguments on average. From a set of H values we choose n values uniformly at random there by allowing repetitions. Let $p(n; H)$ be the probability that during this experiment at least one value is chosen more than once. This probability can be approximated as

$$p(n; H) \approx 1 - e^{-\frac{n(n-1)}{2H}} \approx 1 - e^{-\frac{n^2}{2H}} \tag{13}$$

Let $n(p; H)$ be the smallest number of values we have to choose, such that the probability for finding a collision is at least p. By inverting this expression above, we find the following approximation

$$\mathrm{n}\left(\mathrm{p};\mathrm{H}\right) \approx \sqrt{2Hln\frac{1}{1-p}} \tag{14}$$

and assigning a 0.5 probability of collision we arrive at:

$$\mathrm{n}\left(0.5;\mathrm{H}\right) \approx 1.1774\sqrt{H} \tag{15}$$

Let $Q\left(H\right)$ be the expected number of values, it must be chosen before finding the first collision. This number can be approximated by

$$\mathrm{Q}\left(\mathrm{H}\right) \approx \sqrt{\frac{\pi}{2}H} \tag{16}$$

As an example, if a 64-bit hash is used, there are approximately 1.8×10^{19} different outputs. If these are all equally probable, then it would take only approximately 5.1×10^{9} attempts to generate a collision using brute force. This value is called birthday bound [4] and for n-bit codes it could be computed as 2^{n-1} [3]. Table shows number of hashes $n\left(p\right)$ needed to achieve the given probability of success, assuming all hashes are equally likely. The results of the proposed algorithm are shown in TABLE 3:

TABLE 3
probability of random collision of proposed algorithm for various bits outputs

Bits	Possible outputs (rounded)(H)	Desired probability of random collision (rounded) (p)				
		10^{-15}	10^{-9}	1%	50%	75%
64	1.8×10^{19}	1.9×10^{2}	1.9×10^{5}	6.1×10^{8}	5.1×10^{9}	7.2×10^{9}
128	3.4×10^{38}	8.2×10^{11}	8.2×10^{14}	2.6×10^{18}	2.2×10^{19}	3.1×10^{19}
256	1.2×10^{77}	1.5×10^{31}	1.5×10^{34}	4.8×10^{37}	4.0×10^{38}	5.7×10^{38}
512	1.3×10^{154}	5.2×10^{69}	5.2×10^{72}	1.6×10^{76}	1.4×10^{77}	1.9×10^{77}
1024	1.8×10^{308}	5.2×10^{69}	5.2×10^{72}	1.6×10^{76}	1.4×10^{77}	1.9×10^{77}

The state of proposed is related to each message bit. By iterations, significant changes are obtained at the final state even if there is only a one-bit change in the message. According to the above analysis, the proposed algorithm is secured against statistical attacks. For birthday-attack, the security of the cryptosystem is determined by the length of the hash value, which is 128-bit in this proposed function. According to the definition of birthday-attack [4, 22, 23] , the attack difficulty is 2^{64}

Meet-in-the-middle Attack

Meet-in-the-middle attack [10,11] means to find a contradiction through looking for a suitable substitution of the last plaintext block. In this paper we focus on random routing solutions because their consciousness is simple. The proposed algorithm is principally the same analyzed in [1,9].To clarify its operations, let ns represent the current attack represents the sections within the attacker belongs of M_i, and n_i represents the attacker in $\Phi\left\{n_s\right\}$which is in the straight path between ns and attacked n0. The basic idea of the random routing algorithm considered in this paper is very simple as it defines the following two phases:

1. Basic parameter of the principal phase is the probability p^*Actually, the current M_i immediately selects n_s, which represents the M_i in the shortest block near the attacked n_0, as the next M_i with probability p^*. If this occurs, then the algorithm stops here, otherwise, it performs the second phase.

2. In the second phase, the current M_i randomly selects any of the attackers in $\Phi\left\{n_s\right\}$as next one. Note that according to such algorithm M_i selects n_s, that is the attack in the shortest path with probability

$$P_S = p^* + \left(1-p^*\right) / \left|\Phi\left\{n_s\right\}\right| \tag{17}$$

where $\Phi\{n_s\}$ is the number of next M_i of n_s ; whereas any given one belonging to $\Phi\{n_s\} - \{n_0\}$ is selected with probability$(1-p^*)/|\Phi\{n_{CR}\}|$. Note that, the higherp^*, the higher the probability that packets will follow the shortest path, and, vice versa. Accordingly, on the one hand, if p^* decreases it can be expected longer end-to-end routes between packets source and the attacked n_0 and therefore larger energy consumption. However, on the other hand when p^* becomes smaller, it is more difficult for the attacker to intercept messages sent by the actual reader of target, and therefore, the level of security increases. In order to evaluate the impact of the countermeasures to the meen-in-the-middle attack, the output of the reading process is compared in normal conditions and when the meen-in-the-middle attack is performed. To evaluate the attack impact of the disturbing bits on the performance of the proposed scheme, in Figure 6, the bit error rate is shown versus the normalized amplitude of the disturbing bits. Note that when the normalized bits reach the unitary value, bit error probability is 0.5, which means that the meen-in-the-middle attack is not effective. Observe, however, that the increase in energy expenditure goes as the square of the disturbing bits.

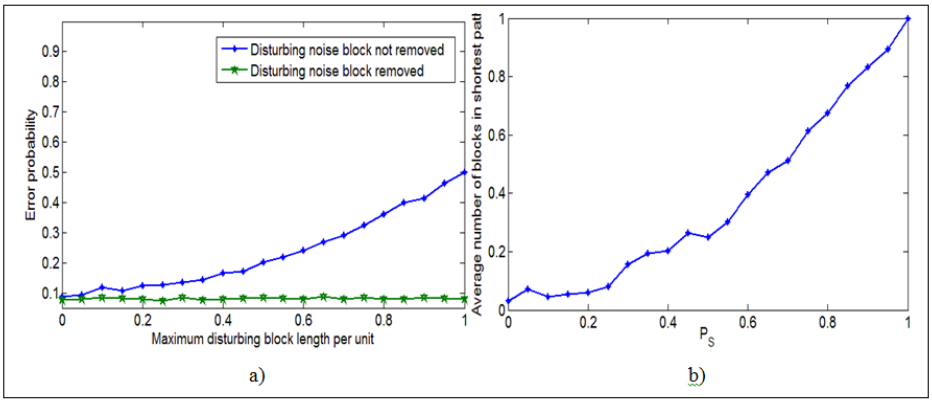


Figure 6 : a) Bit error probability versus the normalized length of the disturbing block, b) Block percentage of the shortest block included in the end-to-end path versus the probability p*

Accordingly, appropriate tradeoff must be identified to reduce energy consumption. In order to evaluate the effects of the adoption of random routing in Figure 6, it is shown the average number of blocks that are included in both the shortest block and in the block obtained by using random routing versus the value of the probabilityp^*. Obviously, this value increases as p^* increases and, therefore, the proposed scheme becomes less effective. However, it can be expected that as p^* increases, the energy consumption decreases. This is indeed demonstrated in Figure 9 where shows the average block length obtained by using random routing versus the value of the probabilityp^*. Note that as p^*increases, the average path length decreases. Note that Figure 7 have been obtained by randomly selecting the positions of the main and the attacked.

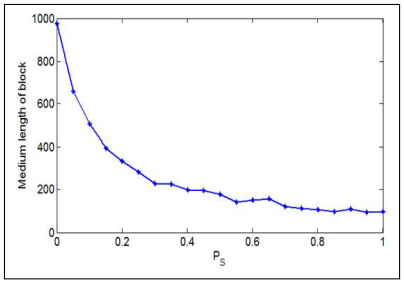


Figure 7 : Average Block length obtained by using random routing versus the value of the probability p*

Collision Test

The following test has been performed for the quantitative analysis on collision resistance [24]: first, the hash value for a paragraph of randomly chosen message is generated and stored in ASCII format. Then a bit in the message is selected randomly and toggled. A new hash value is then generated. The two hash values are compared and the number of ASCII characters with the same value at the same location is counted. Moreover, the absolute difference between the two hash values is calculated by the following formula:

$$d = \sum_{i=1}^{N} t\left(e_i\right) - t\left(e\prime_i\right) \tag{18}$$

Where e_i and $e\prime_i$ is thei_thASCII character of the original and the new hash value, respectively. The function t( ) converts the entries to their equivalent decimal values. This kind of collision test is performed 10,000 times. The maximum, mean, and minimum values of d are listed in Table 4 . The distribution of the number of ASCII characters with the same value at the same location in the hash value is given in Table 6. Notice that the maximum number of equal characters is only 2 and the collision probability is very low.

TABLE 4
Absolute difference of two hash value

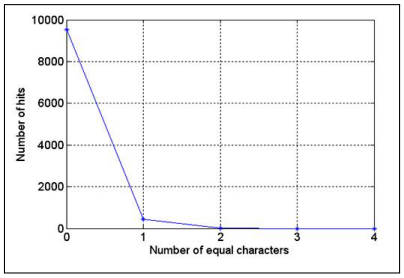
Absolute difference	Maximum	Minimum	Mean
Xiao's scheme	2221	696	1506
Zhang's scheme	2022	565	1257
MD5	2074	590	1304
Proposed scheme	2243	678	1579
Xiao's scheme	2221	696	1506
Zhang's scheme	2022	565	1257

Due to the progresses in technology, NIST plans to replace SHA-1 to longer and stronger hash functions (SHA-224, SHA-256, SHA-384 and SHA-512) by the year 2010 [25], as long hash values are needed in the future. To produce a long hash value, two simple modifications are introduced in the proposed algorithm: increase the size of proposed or extract more bits. For example, 256,512,1024-bit hash values are obtained by using a proposed model with extracting 8 hexadecimal numbers, by extracting 16 hexadecimal numbers and extracting 32 hexadecimal numbers from each lattice value in the origin proposed respectively which is shown in Table 5.

TABLE 5
Statistical performance of proposed algorithm for different output length 512 bits

N	256	512	1024	2048	10,000
$\bar{B}$	63.91	64	63.90	63.99	64.01
P(%)	49.92	50	49.92	49.99	50.01
ΔB	2.69	2.29	2.19	2.44	2.61
ΔP(%)	2.10	1.79	1.71	1.91	2.04
B_{min}	52	52	52	49	49
$\boldsymbol{B_{max}}$	73	73	73	76	81

Figure 8 : Distribution of the number of ASCII characters with the same value at the same location in the hash value.



Hereinto Statistic Analysis of Diffusion and Confusion for Variable Parameter is considered,is the controlled variable. The certain critical value of μ is 0.7 . The followed is some conclusion on chaotic nonlinear map. By varying this parameter as can be seen in the Figure 9, the Statistic analysis of diffusion and confusion are near the same for changing this parameter and it means this algorithm has a stable manner in all variable parameters.

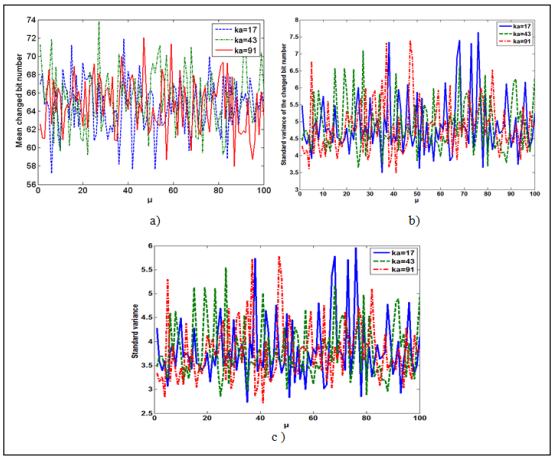


Figure 9 : a) Mean changed bit number for changing μ, b) Standard variance of the changed bit number for changing μ ,c) Standard variance for changing μ.

TABLE 6
Maximum number of equal characters at the same location in the hash value

	Zhang's scheme [27]	Xiao's scheme	MD5	Proposed scheme
Maximum number	2	3	2	2

Based on the result in Tables 1 and 6, the statistical performance of the proposed algorithm is as good as that of MD5. Moreover ΔB and ΔP of the proposed algorithm are smaller than those of MD5, which indicate that the result is more stable. Based on the data in Tables 4 and 6, the proposed algorithm has a bigger exact difference between two hash values than MD5. Meanwhile, the maximum numbers of equal character of MD5 and the proposed algorithm are the same, i.e., only two, which indicates the collision chance is very low.

Speed analysis

Since the proposed algorithm is based on a complex chaotic system, complicated computations are needed in producing a hash value and so this algorithm is slower than MD5 algorithms in [27]. Nevertheless, it still owns a sufficiently high hashing speed for practical use. The PROPOSED model is a kind of spatiotemporal chaos. Compared with the algorithm in [26], which is also based on spatiotemporal chaos, the proposed algorithm has a much higher efficiency. These two algorithms are applied to use Matlab which are running on a personal computer with a Pentium IV 2.66 GHz processor and 4 GB RAM. As monitored from these figures, the execution time of the proposed algorithm is much shorter, especially when the message is long.

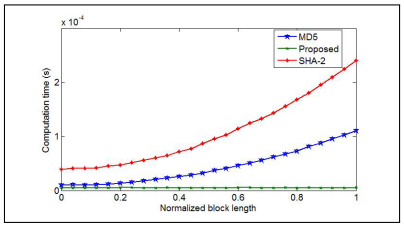


Figure 9: Computation time comparison between the proposed, MD5 and SHA-2 hash function for various normalized block length

5 Conclusion

Based on the Chebyshev Halley methods with variable parameters, a parallel Hash algorithm structure is proposed and analyzed. The algorithm alters the expanded message blocks into the equivalent ASCII code values. The two initial inputs and steps of iterations are generated by last round of iteration, which iterates the chaotic nonlinear map and wholly increases the rise influence of Hash function, and makes the final Hash value has high sensitivity to the initial values, increase the security of Hash function. The analysis designates that the algorithm can meet all the requisites of the Hash function efficiently. And the algorithm is easy to realize a swift and practical program to Hash function structure. The length of the final Hash value generated by this algorithm is 128 bits. The two primary inputs and steps of iterations are generated by last round of iteration, which iterates the chaotic nonlinear map and wholly increases the rise power of Hash function, and makes the final Hash value has high sensitivity to the initial values, increase the security of Hash function. Theoretical analysis and computer simulation show that the proposed algorithm presents numerous interesting features, such as high message, good statistical properties, collision resistance, and secure against meet-in-the-middle attacks that can assure the performance requirements of Hash function. Furthermore the proposed algorithm can present some extra advantages for having convenient controller by the variable parameters.

Bibliography

[1] Boris S. Verkhovsky; Information Assurance Protocols: Efficiency Analysis and Implementation for Secure Communication, *Journal of Information Assurance and Security*, 3(4): 263-269, 2008.

[2] B. Surekha G.N. Swamy, K. SrinivasaRao, A. Ravi Kumar; A Watermarking Technique based on Visual Cryptography Information Assurance Protocols, *Journal of Information Assurance and Security*, 470-473, 2009.

[3] W. Luo, D.C. et al, Hashing via finite field, *Information Sciences*, 176: 2553-2566, 2006

[4] A. Menezes, P. van Oorschot , S. Vanstone, *Handbook of applied cryptography*; CRC Press , 1996.

[5] X. Wang, D. Feng, X. Lai, H. Yu ; Collisions for hash functions MD4 , MD5 ; HAVAL-128 and RIPEMD, Rump Session of Crypto04 E-print, 2004.

[6] X. Wang, H. Yu ; How to break MD5 and other hash functions, *Proceedings of Eurocrypt05*, Aarhus ; Denmark, 19-35, 2005.

[7] S. Wang, W. Liu, H. Lu ; et al., Periodicity of chaotic trajectories in realizations of finite computer precisions and its implication in chaos communications, *International Journal of Modern Physics B*, 18: 2617-2622, 2005.

[8] P. Li, Z. Li, W.A. Halang; et al. G. Chen, A multiple pseudorandom-bit generator based on a spatiotemporal chaotic map, *Physics Letters A*, 349: 467-473, 2006.

[9] SPengFei, QiuShui-Sheng , One-way hash functions based on iterated chaotic systems , *IEEE conference proceedings: communications, circuits and systems, 2007. ICCCAS 2007. International conference on*, 11-13 July; p. 1070-74 , 2007.

[10] Schmitz R., Use of chaotic dynamical systems in cryptography", *Journal of the Franklin Institute*, 38(9):429-441, 2002.

[11] Deng S, Liao X F, Xiao D, A Parallel Hash Function Based on Chaos, *Computer Science*, 35(6): 217- 219, 2008.

[12] Wang X M, Zhang J S and Zhang W F, One way Hash function construction based on the extended chaotic map s switch , *Chin. Phys. Sin*, 52(11): 2737-2742, 2003.

[13] Gao J S, Sun B Y, Han W , Construction of the control orbit function based on the chaos theory, *Electric machines and control*, 2: 150-155, 2002.

[14] Parliz U, Junge L, Kocarev L , Synchronization-based parameter estimation from time series , *PhsRevE*, 4(6): 6253-6259, 1996.

[15] A.M. Ostrowski, *Solution of Equations in Eucilidean and Banach Space*, third ed., Academic Press, New York, 1973.

[16] J.M. GutieÂ´rrez, M.A. Hernandez, A family of Chebyshev Halley type methods in Banach spaces, *Bull. Austr.Math.Soc.*, 55: 113-130, 1997.

[17] J.F. Traub, *Iterative Methods for Solution of Equations*, Prentice-Hall, Englewood Cliffs, NJ,1964.

[18] J.M. GutieÂ´rrez, M.A. Hernandez, An acceleration of Newtons method: super-Halley method, *Appl. Math. Comput.*, 117: 223-239 , 2011.

[19] S. Amat, S. Busquier, J.M. Gutierrez,Geometric constructions of iterative functions to solve nonlinear equations, *J. Comput. Appl. Math.*, 157: 197-205, 2003.

[20] M. Grau, J.L. Daz-Barrero, An improvement of the Euler-Chebyshev iterative method, *J. Math. Anal. Appl.*, 315: 1-7 , 2006.

[21] Jisheng Kou, Yitian Li, Xiuhua Wang, A family of fifth-order iterations composed of Newton and third-order methods, *Appl. Math. Comput.*, in press, doi:10.1016/j.amc..07.150 ,2006.

[22] Secure Hash Standard, Federal Information Processing Standards Publication (FIPS PUB) 180-2, 2002.

[23] Security Requirements for Cryptographic Modules, Federal Information Processing Standards Publication (FIPS PUB) 140-1, 2002.

[24] K. Wong, A combined chaotic cryptographic and hashing scheme, *Physics Letters A*, 307:292-298 , 2003.

[25] NIST Brief Comments on Recent Cryptanalytic Attacks on Secure Hashing Functions and the Continued Security Provided by SHA-1, 2004,
http : //csrc.nist.gov/hash$_s$*tandards*$_c$*omments.pdf*

[26] H. Zhang, X. Wang, Z. Li, D. Liu, One way Hash function construction based on spatiotemporal chaos, *Act a PhysicaSinica*, 54(9): 4006-4011 (in Chinese) , 2005.

[27] J. Zhang, X. Wang, W. Zhang, Chaotic keyed hash function based on feed forward feedback nonlinear digital filter, *Physics Letters A*, 362: 439-448 , 2007.

[28] D. Goldberg, D. Priest, What every computer scientist should know about floating-point arithmetic, *ACM Computing Surveys*, 23(1): 548 , 1991.

[29] M.Nouri, S.Abazari Aghdam, P.Pourmahdi and M.Safarinia, Analysis of a Novel Hash Function Based upon Chaotic Nonlinear Map with Variable Parameter, *Journal of Computer Science and Information Security (IJCSIS)*, 221-228, 2011.

[30] M.Nouri, A.Khezeli, A.Ramezani and A.Ebrahimi , Dynamic Chaotic Hash Function Based upon Circle Chord Methods , *Sixth International Symposium on Telecommunications (IST)*, 1044 - 1049, 2012.

[31] Nouri, M.; Farhangian, N.; Zeinolabedini, Z.; Safarinia, M., Conceptual authentication speech hashing base upon hypotrochoid graph, *Sixth International Symposium on Telecommunications (IST)*, 1136 - 1141, 2012.

[32] Nouri, M. ; Zeinolabedini, Z.; Farhangian, N. ; Fekri, N.; Analysis of a novel audio hash function based upon stationary wavelet transform, *2012 6th International Conference on Application of Information and Communication Technologies (AICT)*, 1 - 6, 2012.