Analysis and Implementation of a Fast Hash Function Based upon Elliptic Curves

Mahdi Nouri, Sajjad Abazari Aghdam, Mostafa Nourzadeh, Mona Hatami, Somayeh Abazari Aghdam

Abstract— **Hash functions are probably the most popular component employed in cryptographic applications for their well known advantages they provide over digital communication links. Hash algorithms bearing the chaos and chaotic behaviors have attracted many attentions due to their full unpredictability. The proposed hash algorithm is based on elliptic functions with the employment of chaos and coupled map lattice with variable parameters. The floating point representation is used to prevent changing a message to different hash values in different environments. The study results show that the proposed hash function has irreversibility, collision resistibility as well as sensitivity to initial values. It can be implemented in parallel, in easy and fast processing for today's technology. This method is more secure than hash functions based on low-dimensional dissipative chaotic maps and it can be implemented much easier.**

Keywords- Hash function; Two-dimensional coupled map lattices; chaotic behaviors; Variable parameter

I. INTRODUCTION

With the rapid development of Internet, ever increasing security and confidentiality of information transfer in the field of electronic communication are required as an essential must [2], 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 [1,2]: unkeyed hash functions for data integrity, and keyed hash functions usually known as Message Authentication Code (MAC). Conventional hash functions such as MD5 and SHA are involved with logical operations or multi-round iterations of some available ciphers. Although each step of the performed iteration is simple the number of processing rounds could be enormous even if the message is very short. Moreover, recent investigations on the collision frequencies reveal many undiscovered flaws in the well-known methods such as MD5, SHA1, and RIPEMD [3–5]. As a ubiquitous phenomenon in nature, chaos is a kind of deterministic random-like process found in nonlinear dynamic systems. It is employed for data protection due to its attractive features such as the sensitivity to initial values, random-like and argotic [6]. Like chaotic cryptosystems, chaos-based hash functions have also interested many researchers. Based on Baptist's encryption method, Wong developed a scheme combining encryption and hashing [7] Although this method is able to encrypt messages and generates the corresponding hash value simultaneously, but the efficiency and security need further improvements [8]. Based on the Piecewise Linear Chaotic Map (PWLCM) or tent map some hash algorithms with higher efficiency are proposed [9–11]. Furthermore, many methods for predicting chaotic time series are published [12–14]. They all try to prevent attacks that breaks hash functions by predicting the chaotic series that employ the complex chaotic map.

Moreover, a one-way function h is a function that for each x in the domain of h it is easy to compute h(x); but for essentially all y in the range of h, it is computationally infeasible to find any x such that $y = h(x)$. Hash function is a special kind of one-way function that possesses the following properties [13]:

- *Compression*: h maps an input x of an arbitrary finite length to an output h(x) of fixed length n.
- *Irreversibility*: Given h and an input x make it easy to compute h(x). However, it is computationally infeasible to find any input which hashes to a specific output, i.e., to find any pre-image x such that $h(x) = y$ when any given y for which a corresponding input is not known.
- *Second pre-image resistance*: It is computationally infeasible to find some second input which has the same output as some specific input, i.e., given x, find a second pre-image ($x_0 \neq x$) such that $h(x) = h(x_0)$.
- *Sensitivity to input bits*: Each output bit is related to input bits. An avalanche property similar to that of good block ciphers is desirable whereby every input bit affects every output bit.

It is well known that Elliptic Functions have the following properties: sensitivity to tiny changes for initial conditions and coefficients, mixing, ergonomics, etc [23].

Recently, spatiotemporal chaos has been magnetizing more and more interests among researchers in the fields of mathematics, physics, and engineering. Compared with simple chaotic maps, spatiotemporal chaos has two supplementary merits for cryptographic purpose. Due to the finite computing precision, chaotic orbits will ultimately become periodic. The period of spatiotemporal chaos is longer than that of simple chaotic maps [6,12]. In particular, the period of chaotic orbits created by a system with a great number of chaotic coupled

oscillators is too long to be reached in practical communications. Consequently, periodicity can be practically avoided in spatiotemporal chaotic systems [6,13]. Furthermore, spatiotemporal chaotic system is a high dimensional chaotic system, which has a number of positive Lyapunov exponents that guarantee multifarious dynamical behavior. It is more difficult or even impossible to forecast the time series made by spatiotemporal chaos. Coupled map lattice model presents spatiotemporal nonlinear system a qualitative description through making space and time discrete and keeping states variable continuous. There are some great benefits of spatiotemporal hyper chaos in comparison with short dimensional chaos. In the former case there exist a huge number of spatial sites. When chaos is applied for treating message, each of which takes chaotic action and serves as a message operation, and then the message can be performed simultaneously and in parallel by many subunits, and thus the efficiency of message healing can be greatly increased.

The enlarged message blocks are converted into the resultant ASCII code values as the iteration times by the proposed algorithm, which the algorithm iterates the chaotic nonlinear map, continuously, with the variable parameters dynamically attained from the position index of the corresponding message blocks to create decimal portions and after that rounds the decimal portions to integers, and finally flows these integers to construct intermediate Hash value. Final Hash value with the length of 128-bit is created by four outputs of last iteration.

II. Analysis of the Hash function Based on Elliptic Function

Elliptic curves can be defined over any field K; the formal definition of an elliptic curve is a non-singular projective algebraic curve over K with order 1 at a given point defined over K. If the characteristic of K is neither two or three, then every elliptic curve over K can be written in the form [23].

A. :

$$y^2 = x^3 + ax^2 + b \quad (1)$$

As it is known, such algorithms have much advantage. The optimum values for the coefficients of the proposed algorithm are chosen within two derivation stages. In the first stage, the derivation is performed with respect to y, and in the second stage, the derivation is performed in respect to x as:

Derivation of x ($\frac{dy}{dx}$): $\longrightarrow$ $2yy' = 3x^2 + 2ax$ (2)

Derivation of y ($\frac{dx}{dy}$): $\longrightarrow$ $3x'x^2 + 2ax'x = 2y$ (3)

By substituting the computed derivations, the discrete form is shown as:

$$y' = y[n+1] - y[n] \quad (4)$$

$$x' = x[n+1] - x[n] \quad (5)$$

Hence:

$$y[n+1] = \frac{3x[n]^2 + 2ax[n] + 2y[n]^2}{2y[n]} \quad (6)$$

$$x[n+1] = \frac{2y[n] + 3x[n]^3 + 2ax[n]^2}{3x[n]^2 + 2ax[n]} \quad (7)$$

The proposed hash functions are performed on a 32-bit machine and consist of many stages based upon coupled map lattice. For simplicity, a two stage machine is investigated here, first, a side for y and another for x in a lattice form. To reach a better response, some changes can be made within the main algorithm to improve the proposed algorithm performances:

$$y[n+1] \to y[n+4], x[n+1] \to x[n+4] \quad (8)$$

$$a_y \to -\frac{1}{2}, a_x \to -\frac{1}{2x[n]} \quad (9)$$

Where $x_i, y_i \in [0,1)$ and a_y is the controlled variable and belongs to (0,10). The map is nonlinear and the parameter a_x ensures that the map runs in a chaotic status when $0 < a_y < 10$. It transforms an interval (0,10) onto itself and includes only one parameter. 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 $x_0, y_0 = 0.35$ and parameter $a_y = -0.5$. The map has some properties, which are appropriate for constructing the Hash function, such as initial value sensitivity and parameter sensitivity, with variable parameter a_x valued in the interval (0,1). Figure 1 displays the chaotic iteration property with variable parameter a valued in the interval (0,10), which initial values are $x_0, y_0 = 0.35$.

The map has some properties, which are appropriate for constructing the Hash function, such as initial value sensitivity and parameter sensitivity, with variable parameter *k* valued in the interval (0,1). Figure 2 displays the chaotic iteration property with variable parameter a valued in the interval (0, 10), which initial values are $x_0, y_0 = 0.35$. Only after several iterations, the sensitivity of chaotic systems can be shown. Take a test data as an example to test the sensitivity, the difference between the two initial values of each group, which

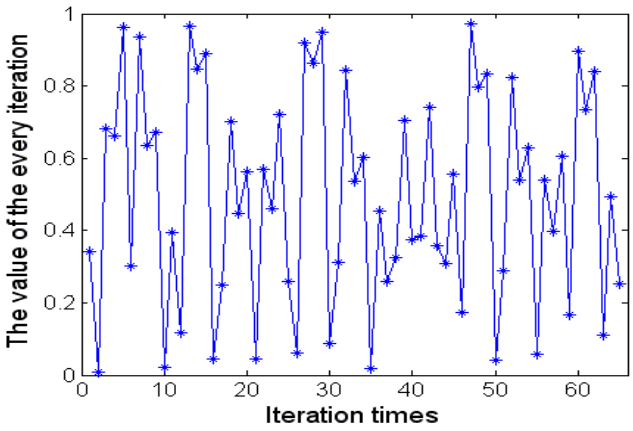


Fig. 1 Iteration property with changeable parameter a when $x_0, y_0 = 0.35$

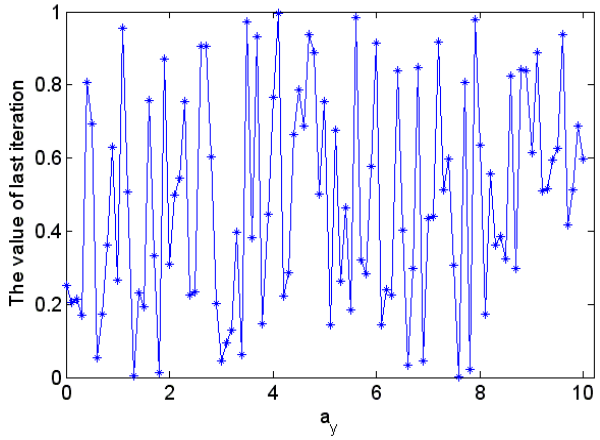


Fig. 2 Iteration property with changeable parameter k when $x_0, y_0 = 0.35$

TABLE I
THE DATA OF SENSITIVITY TEST FOR N=8,16

N	8	16
0.10000000000	0.585271953946239	0.151651625422085
0.10000000001	0.527569535841486	0.151717969326340
0.30000000000	0.234277690429915	0.521789230012578
0.30000000001	0.042692359899307	0.251349079835783
0.50000000000	0.799231271734095	0.322664392758495
0.50000000001	0.717425312838585	0.327979480128847
0.70000000000	0.580984761327791	0.734873200933088
0.70000000001	0.613307785478567	0.733537006543614
0.90000000000	0.912822703319106	0.296589693832238
0.90000000001	0.912830399754912	0.298913061238458

TABLE II
THE DATA OF SENSITIVITY TEST FOR N=32,64

N	32	64
0.10000000000	0. 492030760204249	0. 676206961312453
0.10000000001	0.739807354993843	0. 255690676291725
0.30000000000	0. 172741143820386	0. 491245180047975
0.30000000001	0.358467722762351	0. 930421713442810
0.50000000000	0.293420964172318	0. 981526544964812
0.50000000001	0.311091730191061	0. 952947038281160
0.70000000000	0.528835657010481	0.808479816199454
0.70000000001	0.674135883028605	0.769349336615293
0.90000000000	0.194016261781210	0. 625279969902671
0.90000000001	0.931917255574683	0. 160454188396437

is denoted by e, meets the condition $|e|<=10^{-12}$. Choose 8, 16, 32 and 64 respectively as the number of Iterations. The difference of the final value can be shown in Table 1. In this experiment, iteration operation on 2000 homogeneously distributed interval data in [0, 1) is performed several times, also choose 8, 16, 32 and 64 respectively as the number of iterations, list the statistics of the final value in each region. As can be seen from Table I, II, the sensitivity to initial values demonstrates obviously after 32 iterations. Therefore, in this algorithm, taking into account both the speed and security, we make the number of iterations range from 61 to 90.

III. HASH FUNCTION BASED ON STANDARD MAP

The whole structure of the algorithm can be shown in Figure 3.

A. Message expansion

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 ×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, computed =(*W* mod 64), 0 ≤ *d* ≤ 12. Pad as follows: if 12 ≤ *d* <16 then pad 12 − *d* bytes, otherwise pad 28 − *d* 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.

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

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

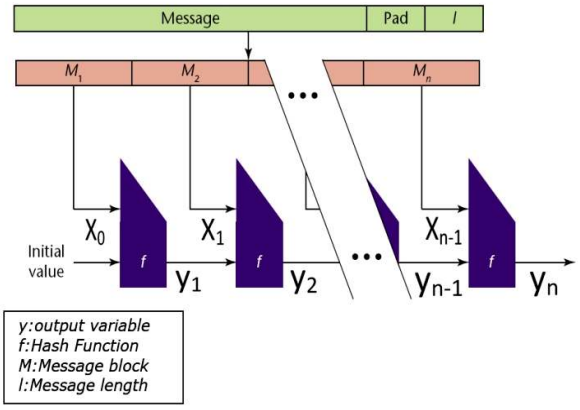


Fig. 3. Whole algorithm structure

3) *Setting the initial values*: The initial values *IV* is the following eight 32-bit words in hex same as SHA-256 [3]:

$$IV = 2^{-32} \times \begin{cases} 6A09E667 \\ BB67AE85 \\ 3C6EF372 \\ A54FF53A \end{cases} \quad (2)$$

4) *Float point number representation:* The new float point number representation in this paper is to represent each float point number between 0 and 1 with a 32-word $(b_1 b_2 \ldots b_{32})$ as follow:

$$b_1 \times \frac{1}{2} + b_2 \times \left(\frac{1}{2}\right)^2 + \cdots + b_{32} \times \left(\frac{1}{2}\right)^{32} \quad (3)$$

It uses this transform only at the end of every iteration for getting to our 128 bits Means four(32bits).

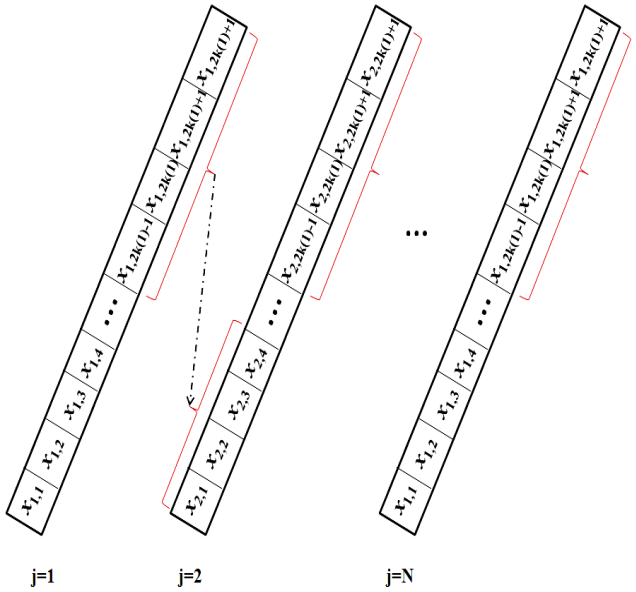


Fig .4. Iterate the proposed for *N* times and change the state

These words were obtained by taking the first 32 bits of fractional parts of the square roots of the first 4 Initial vectors.

B. Algorithm for generating the hash

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_jwhich are the inputs of chaotic nonlinear map.

Compression function inputs consider 16 lattice spaces. Let the initial iterative value of these inputs are:

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

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-bithash value.

TABLE III
ALGORITHM FOR GENERATING THE HASH

Step	Operation
1	$q \leftarrow 1, j \leftarrow 1, i \leftarrow 2q - 1$ go to Step 2
2	$a_y \leftarrow -0.5$ $(x_{j,i+4}, y_{j,i+4}) \leftarrow f(x_{j,i}, y_{j,i}), i \leftarrow i + 1$
3	*if* $i \leq 2k + 2$ go to Step 2
4	$p \leftarrow 2, k + 1 \leftarrow k$, *if* $k \leq ka$ 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*
6	*if* $p < 33$ go to Step 5
7	ka_j=47 $ka_{j+1} \leftarrow ([a_{33}]) mod\ 23 + 61$
8	$y_{j+1,1} \leftarrow y_{j,2ka_{j+3}}$ $y_{j+1,2} \leftarrow y_{j,2ka_{j+4}}$ $y_{j+1,3} \leftarrow y_{j,2ka_{j+5}}$ $y_{j+1,4} \leftarrow y_{j,2ka_{j+6}}$

IV. PERFORMANCE ANALYSES

A. Hash values of messages

Using the proposed algorithm in [14, 15] to conduct simulation analysis under the following 5 kinds of condition:

Condition 1: The original message is ‘‘a hash function is a fundamental building block of information security and plays an important role in modern cryptography. It takes a message as an 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 text message to ‘B’.

Condition 3: Change the word ‘‘input” in the original text message to ‘‘output”.

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:

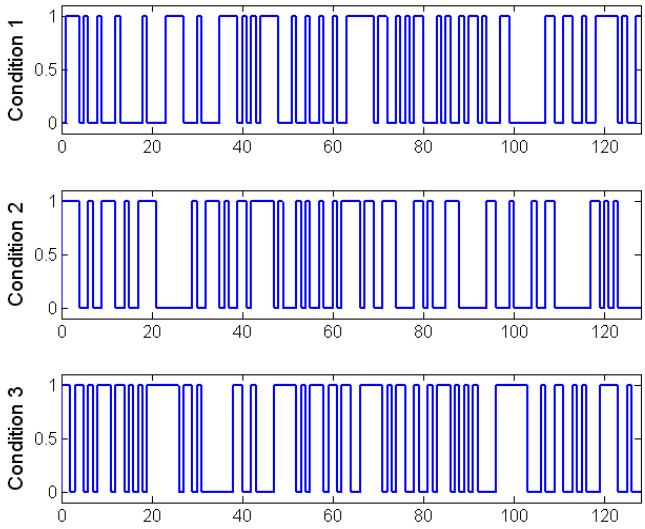
Condition1:86A1A1C12141C01841614686C1E4108116A41E821CE

Condition2:41A1AC14A1AC10064881A81616C1A162884018A81A14

Condition3:81A4114EE1221A1066E1A210E10116CC1C8168018161

Condition4:1814E14C161412121A6141416201E181C01A1412A1618

Condition5:A144A241E8188410106648C10A1E164101401C1E22



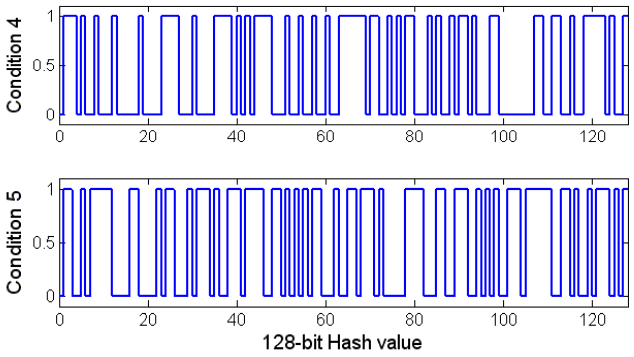


Fig. 5. Hash values under different conditions

The corresponding graphical display of binary sequences is shown in Fig. 4. The simulation results designate that a little difference in the message causes vast changes in the final hash value.

B. Statistic analysis of diffusion and confusion

Confusion and diffusion are two essential design criteria for encryption algorithms, including hash functions. Shannon initiated diffusion and confusion in order to conceal message redundancy [15,16]. Hash function, like encryption system, requires the plaintext to diffuse its effects 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 concealing the statistical structure of the plaintext. Confusion means the utilizing of transformations to make difficult 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 perfect diffusion effect should be that any minute change in the initial condition leads to a 50% changes, probability of each bit. Regularly six statistics are defined as follows:

Minimum changed bit number:

$$B_{min} = \min\left(\{B_i\}_1^N\right) \tag{5}$$

Maximum changed bit number:

$$B_{max} = max(\{B_i\}_1^N) \tag{6}$$

Mean changed bit number:

$$\bar{B} = \frac{1}{N}\sum\nolimits_1^N B_i \tag{7}$$

Mean changed probability:

$$P = \frac{\bar{B}}{128} \times 100 \tag{8}$$

Standard variance of the changed bit number:

$$\Delta B = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(B_i - \bar{B})^2} \qquad (9)$$

Standard variance:

$$\Delta P = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(\frac{B_i}{128} - P)^2} \times 100 \quad (10)$$

Where N is the total number of tests and B_i is the number of changed bits in the i_{th} test. The following diffusion and confusion test has been made that: A paragraph of message is arbitrarily chosen and the corresponding hash value is created. Then a bit in the message is arbitrarily chosen and toggled and a new hash value is created. At last, 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. 5, where N = 10,000. clearly the changed bit number corresponding to 1 bit changed message concentrates around the perfect changed bit number, i.e., 64 bits. It indicates that the algorithm has very strong capability for diffusion and confusion.

The test results in N = 256, 512, 1024, 2048, 5000, 10,000 are listed in Table IV respectively. 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%, based on the results. While ΔB and ΔP are extremely small, which indicate the diffusion and confusion capability are very constant. Therefore a small difference in the plaintext will cause great changes in the hash value, which donates to the high plaintext-sensitivity of the proposed hash function. This property is significant in maintaining the secrecy against statistical attacks.

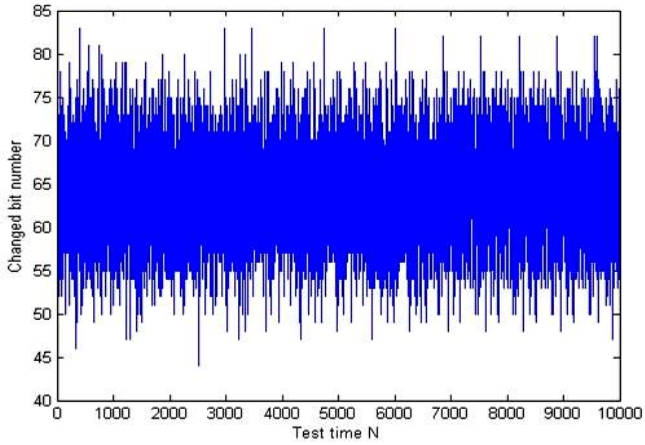


Fig.6. Distribution of changed bit number

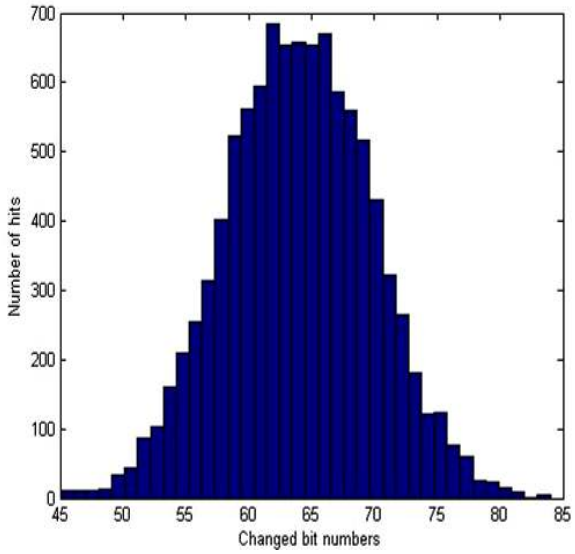


Fig.7. histogram of changed bit number

C. *collision resistant*

- *Birthday Attack*

A birthday attack is a type of cryptographic attack, so named because it exploits the mathematics behind the birthday problem in probability theory. Given a function f, the goal of the attack is to find two different inputs x_1, x_2 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 and x_2 with $f(x_1) = f(x_2)$ after evaluating the function for about $1.25\sqrt{H}$ different arguments on average.

- *Meet-in-the-middle attack*

The meet-in-the-middle attack is a cryptographic attack which, like the birthday attack, makes use of a space-time tradeoff. While the birthday attack attempts to find two values in the domain of a function that map to the same value in its range, the meet-in-the-middle attack attempts to find a value in each of the range and domain of the composition of two functions such that the forward mapping of one through the first function is the same as the inverse image of the other through the second function, quite literally meeting in the middle of the composed function. Collision resistance is one of the most important properties of hash functions. It means that it would be computationally infeasible that two different messages result in same hash value.

TABLE IV
STATISTICAL PERFORMANCE OF THE PROPOSED ALGORITHM

N	N=256	N=512	N=1024	N=2048	N=10000
B	63.84	64.05	64.01	64.01	64.04
P%	49.88	50.04	50.01	50.01	50.03
ΔB	5.40	5.42	5.45	5.44	5.58
ΔP	4.22	4.23	4.25	4.25	4.36
B_{max}	78	78	80	80	84
B_{min}	51	47	47	47	45

TABLE V
STATISTICAL PERFORMANCE OF ALGORITHM [19]

	N=256	N=512	N=1024	N=2048	N=10000
B	63.35	64.62	63.41	64.43	63.35
P%	49.32	50.53	49.49	50.46	49.32
ΔB	5.934	5.816	5.675	5.568	5.934
ΔP	5.013	4.927	4.684	4.506	5.013
B_{max}	69	73	78	85	69
B_{min}	51	48	47	44	51

TABLE VI
STATISTICAL PERFORMANCE OF ALGORITHM [21]

	N=256	N=512	N=1024	N=2048	N=10000
B	63.98	63.94	63.91	63.96	63.98
P%	49.98	49.95	49.92	49.97	49.98
ΔB	5.53	5.31	5.58	5.52	5.53
ΔP	4.33	4.15	4.36	4.32	4.33
B_{max}	81	81	80	85	81
B_{min}	44	46	46	44	44

TABLE VII
STATISTICAL PERFORMANCE OF ALGORITHM [20]

	N=256	N=512	N=1024	N=2048	N=10000
B	63.68	63.92	63.98	64.03	64.05
P%	49.75	49.93	49.98	50.02	50.04
ΔB	5.38	5.78	5.73	5.66	5.68
ΔP	4.20	4.36	4.48	4.42	4.44
B_{max}	78	82	81	86	83
B_{min}	49	49	42	44	42

D. Analysis of speed

The proposed algorithm is approximately comparative to the length of original message, the iteration steps range from 61 to 90, because the step of each round is arbitrary, the whole iteration steps of this algorithm are estimated approximately. While customary Hash algorithms like MD5, SHA need to structure the primitive plaintext together to the unchanging length. When original message is very small, the existing algorithms still need to do lots of computation, while the proposed algorithm only needs a few steps of iterations to save the computing time.

E. Analysis of collision resistance

The following test is performed to conduct quantitative analysis on collision resistance [6, 17,18]: first, the Hash value for a paragraph of message randomly chosen is generated and stored in ASCII format. Then a bit in the paragraph is selected randomly and toggled, and thus a new Hash value is then generated and stored in the same format. Two Hash values are compared, and the number of ASCII characters with the same value at the same location in the Hash values, namely, the number of hits, is counted. A plot of the distribution of the number of hits is given in Fig. 8, and it can be seen, there are 16 tests hit twice and 347 tests hit once, while in 9,637 tests, no hit occurs. It is noticed that the maximum number of equal character is only 2 and the collision is very low. Where $e_i$and e'_i is the i_{th} ASCII 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 VIII. The distribution of the number of ASCII characters with the same value at the same location in the hash value is given. Notice that the maximum number of equal characters is only 2 and the collision probability is very low.

$$d = \sum_{i=1}^{N} \left| t(e_i) - t(e'_i) \right| \tag{11}$$

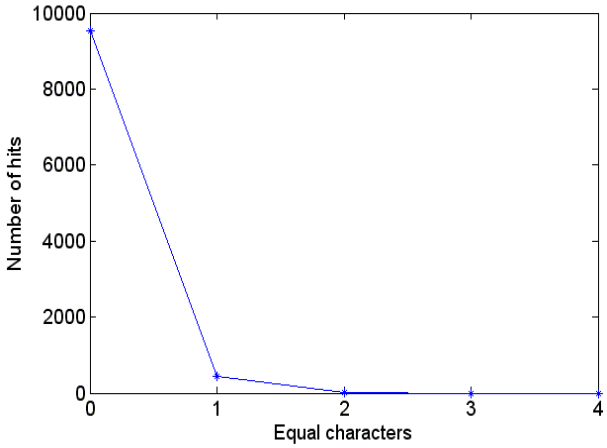


Fig. 8. Distribution of the number of ASCII characters with the same value at the same location in the Hash value

TABLE VIII
ABSOLUTE DIFFERENCE OF TWO HASH VALUES

Absolute difference	Maximum	Minimum	Mean
Xiao's scheme	2221	696	1506
Zhang's scheme	2022	565	1257
MD5	2074	590	1304
Proposed scheme	2457	678	1583
SHA-1	2730	795	1603

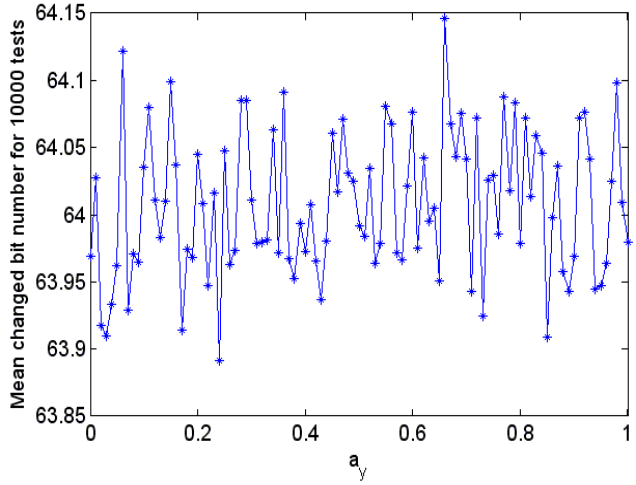


Fig. 9. Mean changed bit number for changing a_y

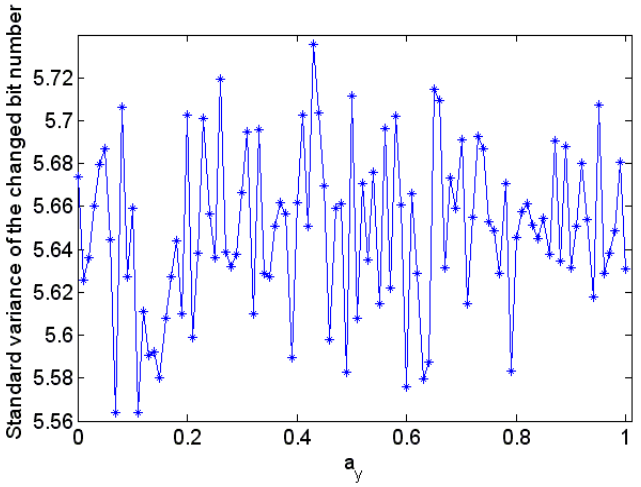


Fig. 10. Standard variance of the changed bit number for changing a_y

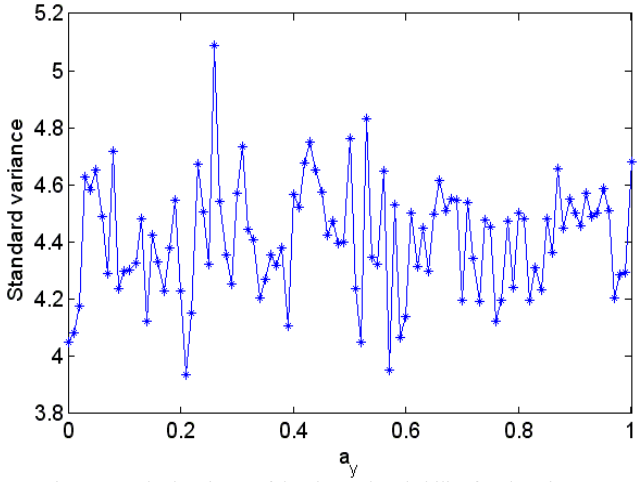


Fig.11. Standard variance of the changed probability for changing a_y

F. Statistic analysis of diffusion and confusion for variable parameter

Hereinto, k is the controlled variable. The certain critical value of a_x is 0.234.The followed is some conclusion on chaotic nonlinear map. By varying this parameter as can be seen in the Fig. 9, Fig. 10 and Fig. 11, 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.

IV. CONCLUSION

In this paper, a new hash algorithm based on elliptic functions, chaos and coupled map lattice has been proposed. The algorithm converts 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. Statistical analysis shows that the proposed hash function has a good confusion and diffusion properties and mere a bit change in message, results in about 50% change in hash value. And the two initial inputs and steps of iterations are generated by last round of iteration, which greatly enhances the proliferation effect of Hash function, and makes the final Hash value has high sensitivity to the initial values, increase the security of Hash function. 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 analysis indicates that the algorithm can meet all the requirements of the Hash function efficiently and the algorithm is easy to realize, which is a fast and practical program to Hash function construction. This shows that hash algorithm can stand against the attacks on hash functions such as meet in the middle attack and birthday attack. Analysis shows that proposed algorithm has three main properties of hash functions: irreversibility, collision resistance and sensitivity to initial values. Furthermore the proposed algorithm can give some extra advantages for having convenient controller by the variable parameters, where as the results of experiment do not change. Due to structure of lattice, the algorithm can be implemented in parallel.

REFERENCES

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

[2] B. Surekha G.N. Swamy, K. Srinivasa Rao, A. Ravi Kumar, "A Watermarking Technique based on Visual Cryptography Information Assurance Protocols", Journal of Information Assurance and Security, 4(6), pp. 470-473, 2009.

[3] Stallings W., Cryptography & Network Security Principles and Practices, Third Edition, Pearson Education, 2004.

[4] Schmitz R, "Use of chaotic dynamical systems in cryptography", Journal of the Franklin Institute, vol.38, no.9, pp.429-441, 2002.

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

[6] Bo Yang, Zhimin Li, Shihui Zheng, Yixian Yang, "Hash function construction based on coupled map lattice for communication security", Global mobile congress 2009, no.7, pp.1-7,2009.

[7] Y. Wang,X, Liao,D, Xiao,K, W. Wong, "One-way hash function construction based on 2D coupled map lattices" , Information Sciences, 2008, 178(5), pp.1391-1406 .

[8] Zhang Jia-shu, Wang Xiao-min, Zhang Wen-fang, "Chaotic keyed hash function based on feedforward- feedback nonlinear digital filter", Physics Letters, 1 (362), pp.439-448, 2007.

[9] 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), pp.2737-2742., 2003.

[10] Gao J S, Sun B Y, Han W, "Construction of the control orbit function based on the chaos theory", Electric machines and control, no.2, pp.150-155, 2002.

[11] Short K M, "Unmasking a modulated chaotic communications scheme", Bifurcation & Chaos, vol.6, no.2, pp.367-375, 1993.

[12] Parliz U, Junge L, Kocarev L, "Synchronization-based parameter estimation from time series", PhsRevE, vol.4, no.6, pp.6253-6259, 1996.

[13] Wang J Z, Wang Y L, Wang M Q, "The collision problem of one kind of methods for constructing one- way Hash function based on chaotic map", Chin. Phys. Sin, vol.55, no.10, pp.5048-5054, 2006.

[14] J. N. Liu, J. C. Xie, P. Wang. One way hash function construction based on chaotic mappings. Journal of Tsinghua University (Science and Technology), 2000, 40(7): 55-58.

[15] Xiao D, Liao X F, Deng S J, "One-way Hash function construction based on the chaotic map with changeable –parameter Chaos", Solitons & Fractals, vol.24 no.1, pp.65-71. 2005

[16] FIPS PUB180-2, Secure Hash Standard, NIST,US Department of Commerce, Washington D.C.

[17] Bo Yang, Zhimin Li, Shihui Zheng, and Yixian Yang. "HASH FUNCTION CONSTRUCTION BASED ON COUPLED MAP LATTICE FOR COMMUNICATION SECURITY", 2009 , Page(s): 1-7

[18] D. Xiao, X. Liao, S. Deng, One-way Hash function construction based on the chaotic map with changeable-parameter, Chaos Solitons & Fractals 24 (2005) 65–71.

[19] X. Yi, Hash function based on chaotic tent maps, IEEE Transactions on Circuits and Systems II 52 (6) (2005) 354–357.

[20] H. Zhang, X. Wang, Z. Li, D. Liu, One way Hash function construction based on spatiotemporal chaos, Acta Physica Sinica 54 (9) (2005) 4006–4011 (in Chinese).

[21] J. Zhang, X. Wang, W. Zhang, Chaotic keyed hash function based on feedforward–feedback nonlinear digital filter, Physics Letters A362 (2007),439–448.

[22] K. Wong, A combined chaotic cryptographic and hashing scheme, Physics Letters A 307 (2003) 292–298.

[23] I. Blake; G. Seroussi, N. Smart (2000). *Elliptic Curves in Cryptography*. LMS Lecture Notes. Cambridge University Press. ISBN 0-521-65374-6

Mahdi Nouri (S'09–M'11) received the B.S. and M.S. degrees in communication system engineering, the M.S. degree in communication secure system engineering from Iran University of Science and Technology (IUST), Tehran, in 2011. From 2007 to 2009, he was a Research Engineer and then Assistant Scientist, working on signal processing and DSP at the Institute of DSP, Tehran Academy of Sciences, Iran. Currently, His research interests are in the areas of Digital signal processing, Channel Coding, Channel Modeling and Cryptography. Email: [1]mnuri@elec.iust.ac.ir

Sajjad Abazari was born in Urmia, Iran, in 1984. He received the B.Sc and M.Sc degrees in Electrical engineering from IAU University of Urmia Branch, Iran, in 2006 and IAU University of South Tehran Branch, Tehran, 2011. He is working for Ph.d program. His research interests include, antenna and RF design, radars, DSP and Cryptography. E-mail: sajjadabazarei@gmail.com

Mostafa Nourzadeh (S'11) was born in 1989; he is student in the B.S degree in Information and Communication Technology engineering from A.B.A Institute of Higher Education, Abeyk, Iran. From 2011 till now, he was a Research Engineer in Cryptography and working on Hash Function, Currently, Her research interests are in the areas of Cryptography and Digital signal processing. E-mail: mnourzadeh@gmail.com,

Mona Hatami (S'11) was born in 1986; she is student in the B.S degree in Information and Communication Technology engineering from A.B.A Institute of Higher Education, Abeyk, Iran. From 2011 till now, she was a Research Engineer in Cryptography and working on Hash Function, Currently, Her research interests are in the areas of Cryptography and Digital signal processing. E-mail: mhatami@gmail.com.

Somayeh Abazari Aghdam she received B.S. Department of Phisics, Islamic Azad University Uromia and M.S. degrees in Phisics from Department of Phisics, Islamic Azad University Mahabad, Iran. From 2009 till 2011, she was a Research Engineer in Cryptography on Hash Function and signal processing working, Currently, Her research interests are in the areas of Fiber Optics, Cryptography and Digital signal processing. E-mail: somayeh.abazary@gmail.com