MALAYSIAN JOURNAL OF COMPUTING AND APPLIED MATHEMATICS (2020) 3 (2) 73-80

DOI: *https://doi.org/10.37231/myjcam.2020.3.2.56*

Article: Received 26.07.2020; Accepted 16.11.2020; Published 25.12.2020

SECURE MD4 HASH FUNCTION USING HENON

Amine Zellagui [a], Naima Hadj-Said [b], Adda Ali-Pacha [c]

[a,b,c] Department of Electronic, University of Sciences and Technology of Oran – Mohamed Boudiaf, USTO-MB, Algeria
aminegeT29@gmail.com

Abstract: Secure hash functions play a fundamental role in cryptographic and web applications. They are mainly used, in the context of digital signatures, to verify the integrity and authenticity of information, in recent years research have found weaknesses in a number of hash functions like MD4,MD5 and SHA-1 ,So in this paper a modified scheme of MD4 was proposed by replacing the original message index K and bit rotation S with new sequence using Henon chaos systems , this proposed scheme given high sensibility of any little change to the original message ,great statistical diffusion and confusion performance, high resistance to collision.

Keywords: MD4, Hash Function, Chaotic Maps, Confusion and Diffusion

1. INTRODUCTION

The Internet has evolved so much that it has become an essential communication tool. However, this communication increasingly involves strategic issues related to the activity of companies on the Web. Transactions made through the network can be intercepted, especially since the laws are difficult to set up on the Internet, so we must ensure the security of this information, it is the cryptography that takes care of it.

The hash functions are used to calculate an arbitrary size input data with a fixed size fingerprint. This size generally varies between 128 and 512 bits. The traditional hash functions algorithms such as MD4 [1] , MD5 [2] , ripemd [3] and haval-127 [4] has been successfully attacked by X Wang in 2004 [5],and SHA-1[6] by Marc Stevens in 2017 [7],It encourages researchers to find alternatives to synthesis efficient hash functions with ease of computations.

In recent years, the chaos system has become most important by researchers [11] [12], which are known to be highly sensitive to initial conditions and control parameters and desired confusion and diffusion properties. These properties has encouraged researchers to use chaos in crypto-systems and hash functions.

In this paper, we propose a modification of the hash algorithms the most used of the MD family like MD4 by using a chaos system, to increase security, and make hash function more sensitive.

.

2. CHAOS SYSTEMS

Chaotic system is a dynamic deterministic system that has unpredictable behavior in the long run. This unpredictability is due to sensitivity to initial conditions and he has a periodic behavior

Henon map

The Hénon map is a 2D chaotic map created by Michel Hénon [8]. It is a discrete-time dynamical system. The Hénon map takes a point (xn, yn) showed in Fig. 1, It is mathematically defined as:

$$x_{n+1} = 1 - a x_n^2 + y_n$$
$$y_{n+1} = b x_n$$

where a=1.4, b=0.3.

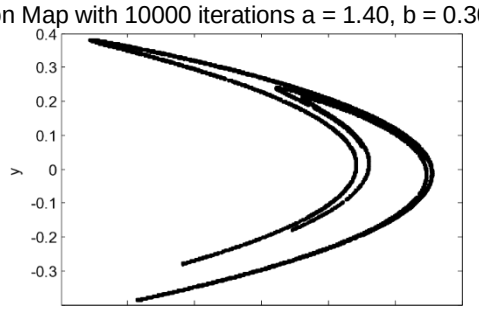

Fig. 1. Bifurcation diagrams of Hénon map

3. MD4 HASH FUNCTION

The MD4 algorithm considered as the origin of the MD-SHA family designed by Rivest in 1990, it is an iterative hash function running on 32-bit words. The round function takes as input a 4-word chaining variable and a 16-word message block and maps it to a new chaining variable. All operations are defined on 32-bit words. The transformation consists of 3 rounds, and each round consists of 16 steps [1], (see Fig. 2).

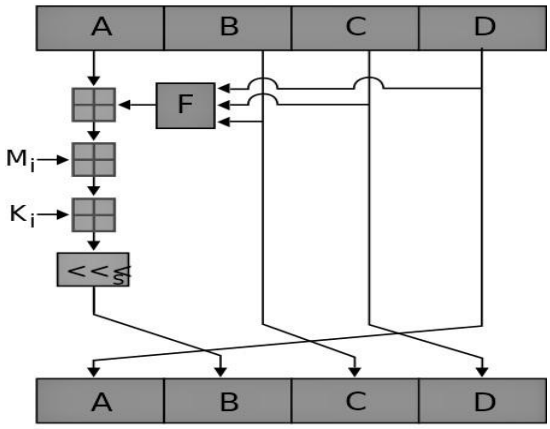

Fig. 2. An MD4 operation

where k[i] is the message index, with: 0 ≤k ≤15, and S is the bit rotation on the left with 0 ≤S ≤15. There are three possible functions F; a different one is used in each round

i. F(X, Y, Z) = XY v (-X)Z
ii. G(X, Y, Z) = XY v XZ v YZ
iii. H(X, Y, Z) = X + Y + Z

To calculate the message digest, we need to follow four steps:

a) Padding: One bit "1" is appended to the message, then "0" bits are appended so that the length in bits of the padded message becomes congruent to 448, modulo 512.Now we add 64-bit to the result of the previous step where 64-bits is the length of the message before the padding bits were added. the resulting message (after padding) has a length that is an exact multiple of 512 bits. Equivalently, this message has a length that is an exact multiple of 16 (32-bit).

   Example: Either the following message: 01100001 01100010 01100011 (in binary), length L = 24 bits.
   - Add '1' bit at the end of the message: 01100001 01100010 01100011 1 (the size becomes 25 bits).

- Then add '0' bits so that the bit length of the padded message becomes congruent to 448, module 512, (448-25 = 423 bits of zeros).
- 01100001 01100010 01100011 1 00 ... 0 (423 bits zeros to be added at the end, then the size becomes 448 bits).
- Now add 64 bits to the result of the previous step, take the original message size L = 24 in binary (24 => 00011000) + 56 bit of zeros.

b) Initialize a buffer: The main MD4 algorithm operates on a 128-bit state, divided into four 32-bit words, denoted A, B, C, and D. These registers are initialized to the following values:
- A = 67452301
- B = EFCDAB89
- C = 98BADCFE
- D = 10325376

Save A as AA, B as BB, C as CC, and D as DD. AA = A, BB = B, CC = C, DD = D.

c) Process Message in 16-Word Blocks: The main algorithm then successively uses each 512-bit message block to change the state. The processing of a message block consists of three similar steps, called rounds; each round is composed of 16 similar operations based on a non-linear function F, a modular addition and a left rotation S.

Proceed as follows:
Algorithm:
for i = 0 à N / 15
for j = 0 to 15 do
set X[j] to M[i*16+j]. end

MD4 uses two "magic constants" in round two and three. The constant of round two is $\sqrt{2}$ and the constant of round 3 is $\sqrt{3}$. Here are their values in octal and hexadecimal (with high order numbers given first) [9]
- In Round 2, constant ($\sqrt{2}$): 013240474631(octal), 5A827999 (hex)
- In Round 3, constant ($\sqrt{3}$): 015666365641(octal), 6ED9EBA1(hex)

Let [A B C D K S] designate the operation:
Round 1 (0j < 15): Let [A B C D i s] designate the operation.
A = (A + F (B, C, D) + x [i]) <<< S.
Such as:
S = [3,7,11,19] rotation to the left, X is the 32-bit message with 0 ≤ i <16 (i in ascending mode).
Round 2 (16 ≤ j < 31): Let [A B C D i s] designate the operation.
A = (A + G (B, C, D) + X [i] + 5A827999) <<< S.
Such as:
S = [3,5,9,13] rotation on the left and i = [0,4,8,12,1,5,9,13,2,6,10,14,3,7,11,15].
Round 3 (32 ≤ j < 47): Let [A B C D i s] designate the operation.
A = (A + H (B, C, D) + x [i] + 6ED9EBA1) <<< S.
Such as:
S = [3,9,11,15] rotation to the left and i = [0,8,4,12,2,10,6,14,1,9,5,13,3,11,7,15]

d) Exit: The output message summary is A, B, C, D 128 bits in size, that is, we start with the byte of the lower order of A and ends with the byte of the higher order of D. This completes the description of MD4.

4. PROPOSED SCHEME OF MD4

To increase the security of MD4 and make it more sensitivity, we will change the fixed values of K and S to the dynamic values by using Hénon map (see Fig. 3). The procedure for producing the values of K and S is as follow:

Step 1: Algorithm
- input: a,b,Ko,So,N>3000
- output: K, S

- Begin
- for i: = 1 to N do
- K(i+1): = (1-(a*K(1, i)^2) + S(1,i)*10^7)mod 16;
- S(i+1): = (b*K(1, i)*10^7) mod 32;
- End

Step 2: Take just the integer number of all values of K and S.
Step 3: In each round, select 16 Number of K and 4 number of S, where in every number should not be repeated.

Example:
Round 1: k = [1 16 12 6 13 11 9 15 2 7 8 3 5 4 10 14], S = [12 27 29 10].
Round 2: k = [1019132 6416578 1112 1415 3], S = [7 1 14 20].
Round 3: k = [419826571112 13 141510 163], S = [16 11 5 8].

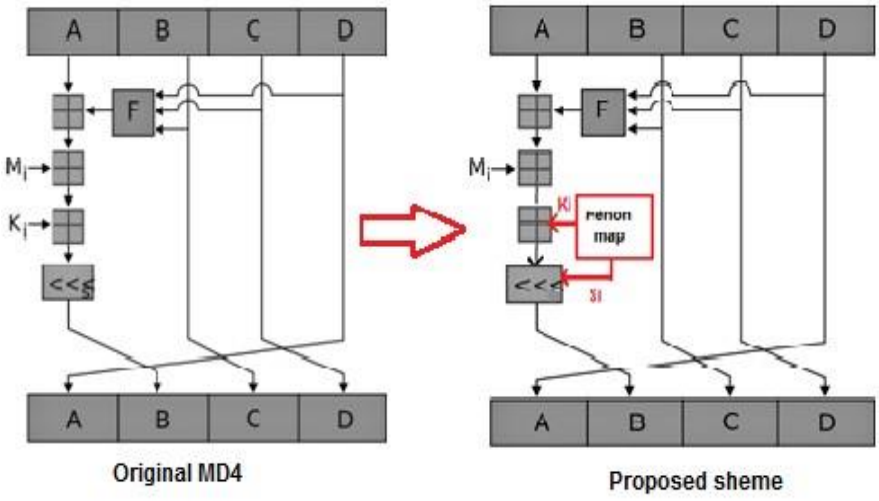

Fig. 3. Proposed scheme of MD4.

Note: if the block size of message superior of 512 bits (more block of length 512), we generate other value of K and S, where each number is different from the previous one.

5. PERFORMANCE ANALYSIS

In this section, we perform several tests to determine the performance of our modified hash function. Performance is evaluated against the scan suite that includes hash distribution, message sensitivity, diffusion and confusion, collision resistance. we also provide a comparison with the original MD4. The values assigned to initial conditions and parameters for simulation are: x0=0.02, y0=0.01, r=0.3, a=1.4.

Distribution Analysis: The uniform distribution of hexadecimal hash value is the crucial property of hashing scheme, we generate a message with random characters, then transformed to ASCII decimal codes plotted in Fig. 4 a), the hash values of the hashing scheme are uniformly spread over the possible range of hash values as illustrated in Fig. 4 b), This ensures that hash distribution is well uniform enough to hide information and act as a strong security measure.

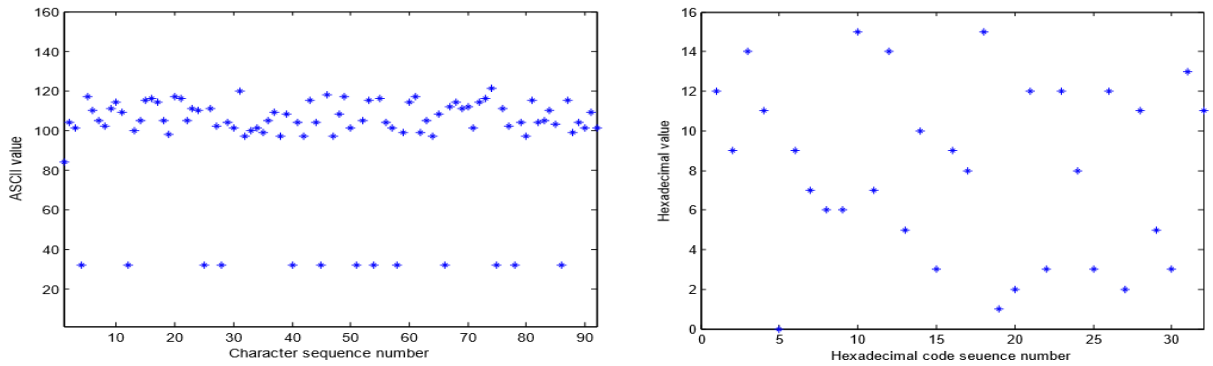

Fig. 4 a) random of message characters, b) hash value with proposed scheme of MD4.

Sensitivity to Small Changes in Message and Initial Conditions: in this subsection, we demonstrate the high sensitivity of the proposed scheme of MD4 hashing function to small changes in original messages and initial conditions. The simulation for sensitivity is done under following conditions.

- Condition 1: The original message is: « Secure hash functions play a fundamental role in cryptographic and web applications. ».
- Condition 2: Replace the first character of the original message “S“ by “s“.
- Condition 3: Replace the last character of the original message “.“ by “:“.
- Condition 4: change x0=0.02 to 0.021.
- Condition 5: change y0=0.01 to 0.001.

Fig.5. Hash value of proposed scheme with different conditions.

The corresponding 128-bit hash values in hexadecimal format are the following:
Condition 1: 08F8769488BE9823DE997EBEA21157DB
Condition 2: 069554335CC70CF3B90F8BC5F7915CBB
Condition 3: AE98822E37F373F748F4A5F11DF364B6
Condition 4: 0D069CCED560A6FFBAB7C7E7C68BB078
Condition 5: BCCDFEEFC5B9A11D128BC8B3E8603A35
All the hash value of proposed scheme with different conditions showed in Fig. 5.

The result of the binary representations of hash, demonstrate that a small modification in the message, initial condition and control parameter can change all the hash value, we can say that the proposed scheme has great sensitivity.

Statistical analysis of diffusion and confusion: Due to security of hashing, Shannon developed two feature named confusion and diffusion [10] to measure the performance of hashing algorithms and hide the message redundancy. Any slight changes in original message should have 50% changing probability for each bit of hash value.

To perform statistical analysis of confusion and diffusion, we need first to calculate the hash value of original message. Secondly. we change 1 bit of original message and calculate hash value, now the two hash values are then compared with each other in order to obtain the number of changed bits, this step will be repeated N times, where N=256, 512,1024 and 10000 (see Fig. 6).

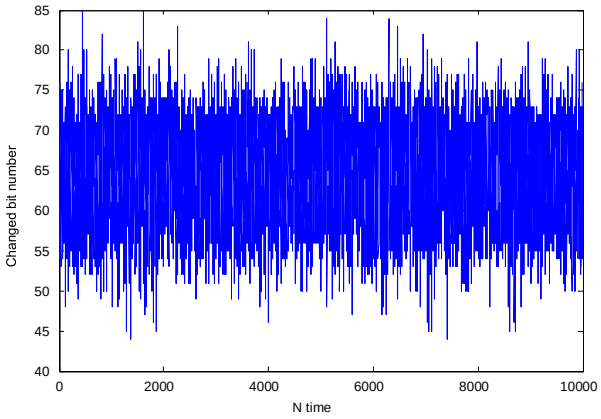

Fig. 6. Spread of changed bit number.

The evaluation of diffusion and confusion capabilities is accounted through the following measures:

Minimum changed bit number:

$$B_{min} = \min(\{B_i\}_1^N)$$

Maximum changed bit number:

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

Mean changed bit number:

$$\bar{B} = \sum_{i=1}^{n} \frac{B_i}{N}$$

Mean changed probability:

$$\mathrm{P} = \frac{\bar{B}}{128} \times 100\%$$

Standard variance of the changed bit number:

$$\Delta \mathrm{B} = \sqrt{\frac{1}{N-1} \times \sum_{i=1}^{N} (B_i - \bar{B})}$$

Standard variance of probability:

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

Table 1. Statistical outcomes for N = 256, 512,1024 and 10,000 of MD4

Parameter	256	512	1024	10000	Mean
$Bmin$	45	45	45	42	44.25
$Bmax$	80	80	83	84	81.75
B	63.894	63.863	64.077	63.958	63.94
P%	49.917	49.893	50.010	49.967	49.94
ΔB	5.9433	5.7402	5.6240	5.6403	5.73
ΔP	4.6432	4.4845	4.3937	4.4065	4.48

Table 2. Statistical outcomes for N = 256, 512,1024 and 10,000 of Proposed scheme of MD4

Parameter	256	512	1024	10000	Mean
$Bmin$	49	49	47	43	47
$Bmax$	80	81	83	83	81.75
B	64.039	64.095	64.230	63.954	64.08
P%	50.030	50.074	50.180	49.964	50.06
ΔB	5.4403	5.4045	5.6305	5.5960	5.51
ΔP	4.2502	4.2223	4.3988	4.3719	4.31

From the results in Tables 1 and 2, it can be observed that and P of proposed scheme are nearly the ideal values of n/2 and 50% respectively and better than original scheme. All values of ΔB and ΔP are very small, which signifies that diffusion and confusion capability of the proposed scheme is very strong and stable.

Resistance to collision attack: Collision resistance is a property of cryptographic hash functions a cryptographic hash function H is collision resistant if it is difficult to find two entries that give the same hash value, so in this subsection we test the Collision resistance of proposed scheme of MD4 by do the following: we choose a message randomly and we get the hash value then stored in ASCII format. Now we change and choose the one bit value in original message, the two hashes with different messages are compared by counting the number of same ASCII characters at the same location.

$$d = \sum_{i=1}^{N} |t(e_i) - t(e_i')|$$

where: ei and ei' are the ASCII characters at position i, d is the absolute difference and t is the thing that changing the ASCII code into corresponding decimal system value. The result is made in Table 3.

Table 3. Comparison on collision resistance for N=2048

Algo	min	max	mean	Mean/char
MD4	696	2322	1368	85.5322
Proposed scheme	538	2249	1371	85.7090

6. CONCLUSION

In this article, a new scheme of MD4 was proposed, we changed fixed value of K and bit displacement S to dynamic value using chaotic Hénon map, the chaotic maps provide high sensitivity to message and key such that even a little change results in dramatic changes in output hash. This proposed scheme given high sensibility of any little change to the original message, great statistical diffusion and confusion performance,

high resistance to collision, it can be considered as a keyed hash function by using initial value K0, S0 as a keys. The proposed scheme has simple structure and flexible, so we can used in MD5 hash function and make it more sensitive.

References

[1] Rivest R. (1992). The MD4 Message-Digest Algorithm," RFC 1320, MIT LCS and RSA Data Security, Inc.

[2] Rivest R. (1992). The MD5 Message-Digest Algorithm," RFC 1321, MIT LCS and RSA Data Security, Inc.

[3] Dobbertin H., Bosselaers A., Preneel B. (1996). RIPEMD-160: A strengthened version of RIPEMD, vol 1039. Springer, Berlin, Heidelberg.

[4] Zheng Y., Pieprzyk J., Seberry J. (1992). HAVAL - A One-Way Hashing Algorithm with Variable Length of Output. In: ASIACRYPT 1992. LNCS, pp. 83–104. Springer, Heidelberg.

[5] Wang X., Feng D., Lai X., Yu H. (2004). Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD. Cryptology ePrint Archive, report 2004/199.

[6] Eastlake D., Jones P. US Secure Hash Algorithm 1 (SHA1)", RFC 3174, September 2001.

[7] Stevens M., Bursztein E., Karpman P., Albertini A., Markov Y. (2017). The first collision for full SHA-1, CWI Amsterdam, Google Research.

[8] Hénon M. (1976). "A two-dimensional mapping with a strange attractor". Communications in Mathematical Physics. 50 (1): 69– 77.doi:10.1007/BF01608556.

[9] Knuth D. (1981). The Art of Programming, Volume 2 (Seminumerical Algorithms), Second Edition, Addison-Wesley.

[10] Shannon C. E. (1949). Communication theory of secrecy systems. Bell Systems Technical Journal, 28, 656–715.

[11] Ahmad M., Khurana S., Singh S., Al Sharari H. D. (2017). A Simple Secure Hash Function Scheme Using Multiple Chaotic Maps. 3D Research, 8(2), article id.13, 15 pp.

[12] Kanso A., Yahyaoui H., Almulla M. (2012). Keyed hash function based on a chaotic map, Information Sciences Volume, 186(1): 249-264.