SCITECH
RESEARCH ORGANISATION|

Volume 6, Issue 2
December 06, 2017

Journal of Information Sciences and Computing Technologies
www.scitecresearch.com/journals

Improving Data Integrity in Communication Systems by Designing a New Security Hash Algorithm

S. Hakim[1], M. Fouad[2]
[1]Department of Electronics, Communications Engineering, Zagazig University, Egypt
email: sarahakimhamdi@gmail.com
[2]Department of Electronics & Communications Engineering, Zagazig University, Egypt
email: fouadzu@gmail.com

Abstract

The objective of this paper is to design a new secure hash algorithm with final hash code length 512 bits. The proposed hash code algorithm based on the combination of SHA-256 algorithm -with modification in its message expansion- and MD5algorithm based on double-Davis-Mayer scheme to reduce the weakness existing in these functions. In this paper; we modified message expansion for SHA-256 in the proposed algorithm .By using MATLAB, The proposed algorithm has been simulated. And hash code for different messages is obtained using MD5, SHA-256, combination of MD5 and SHA-256 with final hash code length 265 bits algorithms and the proposed algorithm . Hash code of the proposed algorithm is different from hash code obtained by MD5, SHA-256 and combination of MD5 and SHA-256 with final hash code length 256 bits algorithms for the same messages. Avalanche test, with one bit difference and more than one bit difference, is applied toSHA-256, combination of MD5 and SHA-256 with final hash code length 256 bits and the proposed algorithm .The proposed algorithm passed avalanche test with higher probability than SHA-256 and combination of MD5 and SHA-256 with final hash code length 256 bit algorithms .The proposed algorithm is more complicated and more secure.

Keywords: Cryptography; Security; Message Digest; Secure Hash Algorithm.

1. Introduction

A cryptographic hash function h is an algorithm which takes a message of variable length as input and produces a fixed length string as output referred as hash code or simply hash of the input message [1]. The hash value is appended to the message at the source at a time when the message is assumed to be correct. the receiver authenticates that message by re-computing the hash value [2] . Hash algorithms, also called as message digest algorithms, are algorithms which generate a unique message digest for an arbitrary message. The digest can be considered as a fingerprint of the message and it must have the following properties: first, the hash must be easy to compute. Second, it must be very hard to compute the message from the digest and third, it must be hard to find another message which has the same message digest as the first message [3]. Hash algorithms are used widely in cryptographic protocols and Internet communication in general. Several widely used hash algorithms exist. One of the most famous algorithms is the MD5 message digest algorithm. MD5 is widely used in several public- key cryptographic algorithms and Internet communication in general. MD5 calculates a 128-bit digest for an arbitrary b-bit message, MD5 algorithm is developed by Ronald Rivest [4].In 1993, B. den Boer and A. Bosselaers found a kind of pseudo-collision for MD5 which consists of the same message with two different sets of initial values. This attack discloses the weak avalanche in the most significant bit for all the chaining variables in MD5 [5]. The other famous algorithm called secure hash algorithm SHA-256 [6] is one member of a family of cryptographic hash functions that together are known as SHA-2. The basic computation for the algorithm takes a block of input data that is 512 bits (64 bytes) and a state vector that is 256 bits (32 bytes) in size, and it produces a modified state vector. It is a follow-on to the earlier hash algorithms MD5 and SHA-1, and it is becoming increasingly important for secure internet traffic and other authentication problems. As the SHA-256 processing involves a large amount of

computations, it is critical that applications use the most efficient implementations available. SHA-256 is also not secure enough as it has an attack for 46 (out of 64) steps of the compression function with practical complexity [7] and preimage attacks on 41 steps SHA-256 [8].

2. Review of Literature

At 2007 it was a combination between MD5 and SHA-1 with hash code length 160 bits [9] . In 2012 it was a combination between MD5 and SHA-1 with hash code length 256 bits [10]. In 2013 it was a combination between MD5 and SHA-256 with hash code length 256 bits [11]. Here we combine the compression function of MD5 and SHA-256 to give final hash code 512 bits by using two SHA-256 to give a good diffusion so that the output in each round will be spread out and not to be equal with the same output in the next coming stages. This is done with XOR-ing each stage output with next input. At the same time Double-Davies-Meyer scheme will ensure the diffusion and will resist against hackers to reach the minimum distance. The 3rdsection explains how MD5 and SHA-256 operate. The 4thsection explains our hash algorithm, equations, block diagram and comparison between hash code of SHA- 256, MD5 and the proposed algorithm. 5thsection presents Results and discussion of the Security analysis and finally we end with conclusion and future work.

3. Description of MD5 And SHA-256

3.1. MD5

MD5 is an improved version of MD4. It is similar in design and also produces a 128-bit hash. After some initial processing, MD5 processes the input text in 512-bit blocks, divided into 16 32- bit sub-blocks. The output of the algorithm is a set of four 32-bit blocks, which concatenate to form a single 128-bit hash value. First, the message is padded so that its length is just 64 bits short of being a multiple of 512. This padding is a single 1-bit added to the end of the message, followed by as many zeros as are required. Then, a 64-bit representation of the message's length (before padding bits were added) is appended to the result. These two steps serve to make the message length an exact multiple of 512 bits in length, while ensuring that different messages will not look the same after padding [4]. Four 32-bit variables are initialized: A = 0x01234567, B = 0x89abcdef, C = 0xfedcba98, D = 0x76543210 these are called chaining variables. Now, the main loop of the algorithm begins and continues for as many 512-bit blocks as are in the message. The four variables are copied into different variables: a gets A, b gets B, c gets C, and d gets D. The main loop has four rounds, all very similar. Each round uses a different operation 16 times. Each operation performs a nonlinear function on three of a, b, c, and d. Then it adds that result to the fourth variable, a sub-block of the text and a constant. Then it rotates that result to the right a variable number of bits and adds the result to one of a, b, c, or d. Finally the result replaces one of a, b, c, or d. There are four nonlinear functions, one used in each operation (a different one for each round) [12].

Function 1: $F\ (X,Y,Z)\ =\ (X\ \wedge\ Y)\ \vee (\neg\ X\ \wedge\ Z)$
Function 2: $G\ (X,Y,Z)\ =\ (X\ \wedge\ Z)\ \vee\ (Y\ \wedge\ \neg\ Z))$
Function 3: $H\ (X,Y,Z)\ =\ X\ \oplus\ Y\ \oplus\ Z$
Function 4: $I\ (X,Y,Z)\ =\ Y\ \oplus\ (X\ \vee\ \neg\ Z)$ where:
($\oplus$ is XOR, $\wedge$ is AND, $\vee$ is OR, and $\neg$ is NOT).

These functions are designed so that if the corresponding bits of X, Y and Z are independent and unbiased, then each bit of the result will also be independent and unbiased. The function H is the bit -wise parity operator . If Mj represents the jth sub -block of the message (from 0 to 15), and << represents a left circular shift of s bits, the four operations are:

1. $FF\ (a,b,c,d,Mj\ ,s\ ,ti)$ denotes $a\ =\ b\ +\ ((a\ +\ F(b,c\ ,d\ )\ +\ Mj\ +\ ti)\ <<<\ s)$.
2. $GG\ (a,b,c,d,Mj\ ,s,ti)$ denotes $a\ =\ b\ +\ ((a\ +\ G(b,c,d\ )\ +\ Mj\ +\ ti)\ <<<\ s)$.
3. $HH\ (a,b\ ,c,d,Mj\ ,s\ ,ti)$ denotes $a\ =\ b\ +\ ((a\ +\ H(b,c,d)\ +\ Mj\ +\ ti)\ <<<\ s)$.
4. $II\ (a,b,c,d,Mj\ ,s,ti)$ denotes $a\ =\ b\ +\ ((a\ +\ I(b,c,d\ )\ +\ Mj\ +\ ti)\ <<<\ s)$.

Figure1. Illustrates one operation within a round. There are four possible functions F; a different one is used in each round: denote the XOR, AND, OR and NOT operations respectively.

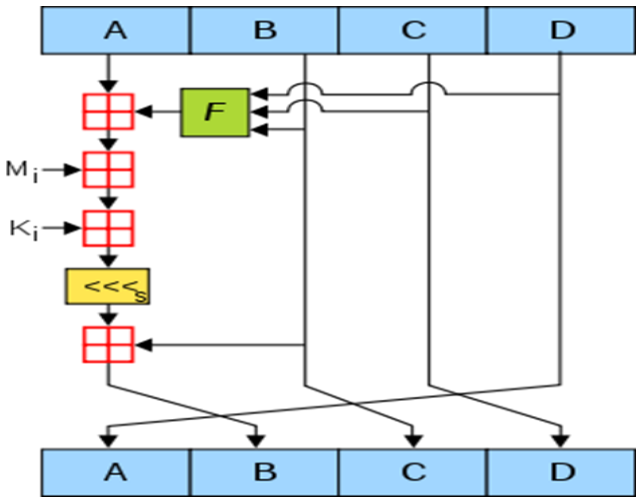

Fig 1: One MD5 operation. MD5 consists of 64 of these operations

Grouped in Four Rounds of 16 Operations. *F* is a nonlinear Function; One Function is Used in Each Round. M_i Denotes a 32-bit Block of The Message Input, and K_i Denotes a 32-bit Constant, Different for Each Operation. $<<<_s$ Denotes a Left bit Rotation by *s* Places; *s* Varies for Each Operation. ⊞ Denotes addition Modulo 2^{32}.

After all of this a, b, c and d are added to A, B, C and D respectively, and the algorithm continues with the next block of data. The final output is the concatenation of A, B, C and D.

3.2. SHA-256

SHA-256 operates in the manner of MD4,MD5 and SHA-1. The message to be hashed is 1st padded with its length in such a way that the result is multiple of 512 bit long , 2nd parsed into 512-bit message blocks $M^{(1)}$, $M^{(2)},\ldots,M^{(N)}$.And 3rd hash computation : SHA-256 uses a message schedule of sixty-four 32-bit words .The words of the message schedule are labeled W_0, $W_1,\ldots$, W_{63}. Expanded message blocks W0, W1,…,W63 are computed as follows via the SHA-256 message schedule:

$W_t = M_t^{(i)}$ for t = 0, 1,…, 15

$W_t = \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16}$ for 16≤ t ≤ 63

where :

$\sigma_1^{\{256\}} = ROTR^{17} \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$

$\sigma_0^{\{256\}} = ROTR^{7}(x) \oplus ROTR^{18}(x) \oplus SHR^{3}(x)$.

The initial hash values H(0) is the following sequence of 32-bit words in hexadecimal form : $H_1^{(0)}$= 6a09e667; $H_2^{(0)}$ = bb67ae85; $H_3^{(0)}$ = 3c6ef372; $H_4^{(0)}$ = a54ff53a; $H_5^{(0)}$ = 510e527f $H_6^{(0)}$ = 9b05688c; $H_7^{(0)}$= 1f83d9ab; $H_8^{(0)}$= 5be0cd19.

- Initialize the eight working variables, a, b, c, d, e, f, g and h, with the $(i-1)^{st}$ hash value for i = 1 to N :

For 0 ≤ t ≤63 {

T1= h + $\sum_1^{\{256\}}(e) + ch(e,f,g) + K_t^{\{256\}} + W_t$

T2= $\sum_0^{\{256\}}(a) + Maj(a,b,c)$

h = g, g = f, e = d+T1, d = c, c = b, b = a, a = T1+T2

}

Where:

$K_t^{\{256\}}$ is a sequence of 64 constant 32-bit words .

Ch(x, y, z)=(x ^ y) ⊕ (x ^ z) .

Maj(x, y, z)=(x ^ y) ⊕ (x ^ z) ⊕ (y ^ z) .

$\sum_1^{256}(x)$= (x>>> 6) ⊕ (x>>> 11) ⊕ (x>>> 25) .

$\sum_0^{256}(x)$= (x>>> 2) ⊕ (x>>> 13) ⊕ (x>>> 22) .

Where : >>> is ROTR

●After repeating steps one through four a total of N times (i.e., after processing M(N)), the resulting hash function is $H_0^{(N)} \| H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)} \| H_6^{(N)} \| H_7^{(N)}$[13] .

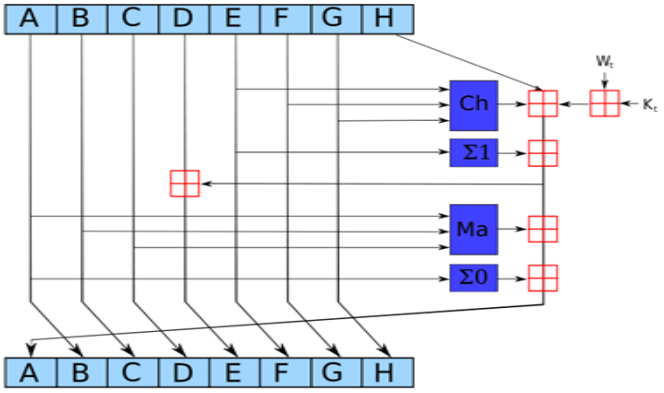

Fig 2: SHA-256 operation

4. Proposed Hash Algorithm

The proposed algorithm is a combination between compression functions of SHA-256 and MD5 Based on double Davis-Mayer scheme as shown in figure 3. which is called SHA256 ᴗ MD5.

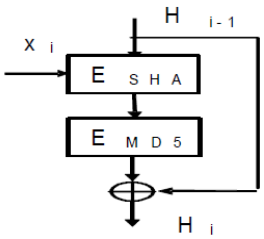

Fig 3: Double-davis-mayer scheme

The first part of our proposed algorithm is similar to SHA-256 process with message expansion modification but the second part looks like MD5 with extended compress function. Here we use different message expansion for sha-256 for helping have minimum hamming weight of the disturbance vector to be high. For 512 bit message blocks $M_{(1)},\ldots..M_{(N)}$ And The words of the message schedule are labeledW_0, $W_1,\ldots$, W_{63} we obtain them from new message expansion as follows :

$W_t = M_t(i)$ for t = 0, 1,…, 15

$W_t = \sigma1\ (W_t\text{-}1) + W_t\text{-}9 + \sigma2\ (W_t\text{-}15) + W_t\text{-}16$ for $(16 \leq t \leq 63)$

Where:

$\sigma1 = x \oplus X^{<<<7} \oplus X^{<<<22}$

$\sigma2 = x \oplus X^{<<<13} \oplus X^{<<<27}$

Here σ1 and σ2 are different for having minimum hamming weight of the disturbance vector high [14],[15],[16]. All other steps of SHA -256 are the same we us it as it is . here we use four MD5 as we use 2 SHA-256 which having 16 registers and each MD5 has four registers so we use 4 MD5. In each round the16 registers of 2 SHA-256 update the 16 registers of four MD5 and output of them updates the 16 registers of two SHA-256 in next round and so on until it gives final hash code with length 512 bits as two SHA-256 but the code of our algorithm is totally different from code of SHA-256 and different from code of the combination of one SHA-256 and 2 MD5 .

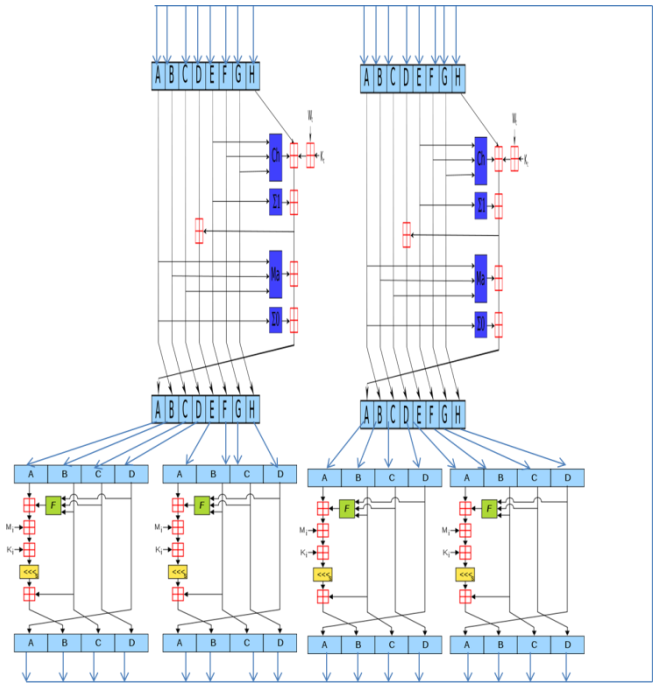

Fig 4: Block diagram of proposed hash algorithm

Where: F is a nonlinear function, Mi denotes a 32-bit block of the message input, Kt denotes a 32-bit constant, different for each operation, $<<<_s$ denotes a left bit rotation by s places; s varies for each operation, ⊞ Denotes Addition Modulo 2^{32}.

5. Experimental Results And Discussions

Like all cryptographic function, hashes are always vulnerable to be attacked by attackers. The longer the hash length (L) is , the greater effort of an attacker to make his attack . There are three important attacks on hashes like "collision attack" which allows an attacker to find two messages M1 and M2 with the same hash value in fewer than 2^(L/2) attempts , "first-preimage attack" which allows an attacker who knows the a desired hash value required to find the message that lead to this value in less than 2^L attempts and "second-preimage attack" if an attacker has a desirable message M1 ,he will find a message M2 that has the same hash value in less than 2^L attempts [10],[16] . Also in SHA-256 a collision in 28 and 64 rounds was founded by differential attack method [17]. A test will be performed on both SHA-256 and the combination between one SHA-256 & 2 MD5 with final hash code 256 bits and our proposed algorithm for a comparison between them .

Table 1: Comparison of hash function for random messages

Message	**Md5 (128bits)**	**Sha-256 (256bits)**	**Sha-256،Md5 (256bits)**	**Sha-256،Md5 (512bits) (proposed)**
.	*5058F 1AF83 88633 F609C ADB75 A75DC 9D*	*50DC91C4 8180BDBF 71114D82 8A950572 206D0703 71F8F4C9 9AA2257F 18A8D258*	*9C1A7DF2 604D5DA1 D566874E 891C0E19 D509F74E CFCA1B91 485A7170 B99A60C5*	*9C1A7DF2604D 5DA1D566874E 891C0E19D509 F74ECFCA1B9 1485A7170B99 A60C59C1A7D F2604D5DA1D 566874E891C0 E19D509F74E CFCA1B91485 A7170B99A60C5*

B	*92EB5 FFEE6 AE2FE C3AD 71C77 753157 8F*	*6EF1A2CA F31B99B5 7352D34E 781AB902 66486018 959B0101 E511C69A 80311972*	*AA69369C 56ABB417 CCA4DDC4 A965304B B072605D CD511F2B 4641750A 07DEFD1D*	*AA69369C56AB B417CCA4DDC 4A965304BB072 605DCD511F2B 4641750A07DE FD1DAA69369 C56ABB417CC A4DDC4A96530 4BB072605DCD 511F2B4641750 A07DEFD1D*
cryptography	*E0D00 B9F33 7D357 C6FAA 2F8CE AE4A6 0D*	*4C8F4ACC 3344427E 8EB2A7CC 69B81E49 7A733D6E CD9FC110 587DD7D7 1AFAB57E*	*BF2DD3D7 DFB86A42 549193EF B321A204 B6524060 F6A10018 6F1155F7 5F4BF1D1*	*BF2DD3D7DFB8 6A42549193EFB3 21A204B6524060 F6A100186F1155 F75F4BF1D1BF2 DD3D7DFB86A4 2549193EFB321A 204B6524060F6A 100186F1155F75 F4BF1D1*
13235989856 23898565665 65989263623 26986232326 88568743565 46949	*2BB30 408977 5F3A1 603E5 E3834 983C0 1*	*0954C63D 6CDECA55 1F289126 3E3BA086 360EB29A F6C3529A 9446E34A 47D56A11*	*C9CBE282 9E3656C1 07C1652E AFEF56FE 0BBF6151 13C0DA1C 806314BB 82A9C2F3*	*C9CBE2829E3656 C107C1652EAFEF 56FE0BBF615113 C0DA1C806314BB 82A9C2F3C9CBE2 829E3656C107C16 52EAFEF56FE0BB F615113C0DA1C8 06314BB82A9C2F3*
Abcdefghijkl mNopqrstuvw xyzABCDEF GHIJKLMNO PQRSTUVW XYZ	*2AD37 2C377 013BA A4EE3 2AB66 49D24 49*	*A238E742 0D4865A7 E3B04ADF D27DEF37 B1E217AE B230008B 210CCD5A CD447727*	*88DE5A6E 20C37444 963C9DF1 9D1AB808 E0CF0EAF 9F3E94D1 188EEAB0 26A77F44*	*88DE5A6E20C3744 4963C9DF19D1AB 808E0CF0EAF9F3 E94D1188EEAB02 6A77F4488DE5A6 E20C37444963C9 DF19D1AB808E0 CF0EAF9F3E94D 1188EEAB026A7 7F44*

From Table I we notice that when we run MD5 algorithm , SHA-256 algorithm , MD5 ب SHA-256 with final code length 256 bits algorithm and our proposed algorithm in mat lab , the final hash code is totally different from each other .

5.1. Avalanche Test

In cryptography, the avalanche effect is the desirable property of cryptographic algorithms . the avalanche test became clear when an input is changed slightly (for example, flipping a single bit) the output changes significantly (e.g., half the output bits flip). When the cryptographic hash function doesn't exhibit avalanche effect to a high degree (around 50%) , then hash function has poor randomization and predictions can be made about input by cryptanalyst , being given only the output . this may be sufficient to break the algorithm partially or totally .so , the avalanche effect is an important condition for the designer of cryptographic algorithm or device . we apply avalanche test first in one bit difference for SHA-256 , SHA-256 ب MD5 with final hash code length 256 bits algorithm and our proposed algorithm SHA-265 ب MD5 with final hash code 512 bits algorithm and 2nd we apply avalanche test in more than one bit different messages as shown :

Table 2: Messages with one bit difference

	Message	**Sha-256 (256bits)**	**Sha-256ب Md5 (256bits)**	**Sha-256ب Md5 (512bits) (proposed)**
1	*Electronic code book(ecb) electronic code book!(ecb)*	*43.359375%*	*50%*	*53.12500%*
2	*WtgenSHA400 WtgenSHA401*	*47.65625%*	*49.609375%*	*50.781250%*
3	*HMAC_GUI_E CE10101010 HMAC_GUI_E*	*48.828125%*	*51.171875%*	*52.343750%*

	CE00101010			
4	*noting deserve nothing!deserve*	*46.87500%*	*53.515625%*	*56.640625%*
5	*f=ans g=ansxortest(f,g) f=ans! g=ansxortest(f,g)*	*49.609375%*	55.8593%	*53.515625%*
6	*24438a59a09B5dB4 35563e1d 24438a59 a09B5dB4 35563e0d*	*45.3125%*	*48.0342%*	*48.046875%*
7	*differential cryptanalysis differential!crypta nalysis*	*47.65625%*	*50%*	*53.12500%*
8	*MMM 159 ^&*(ASDF MMM!159 ^&*(ASDF*	*47.65625%*	*55.46875%*	*52.734375%*
9	*Rpqldjdooldhvwgly lvdlqsduwhfwuhv Rpqld!jdooldhvwgl ylvdlqsduwhfwuhv*	*45.703125%*	*49.609375%*	*49.918750%*
10	*How does Cryptography Work123456789 How does Cryptography Work023456789*	*47.65625%*	*49.609375%*	*54.296875%*
11	*Table 1.22222222222222 222222222 Table 0.22222222222222 222222222*	*44.53125%*	*51.171875%*	*57.03125%*
12	*qqqaaasss 159632178945200 1 qqqaaasss!159632 1789452001*	*44.921875%*	*50.78125%*	*49.609375%*
13	*2000 2001 2002 2003 2004 2005 2000!2001 2002 2003 2004 2005*	*49.609375%*	*60.15625%*	*51.171875%*
14	*Message Authentication Codes Message!Authentic ation Codes*	*48.046875%*	*55.46875%*	*53.1250%*
15	*Workspace 8*8 char Workspace!8*8 char*	*50.390625%*	*50.78125%*	*50.78125%*

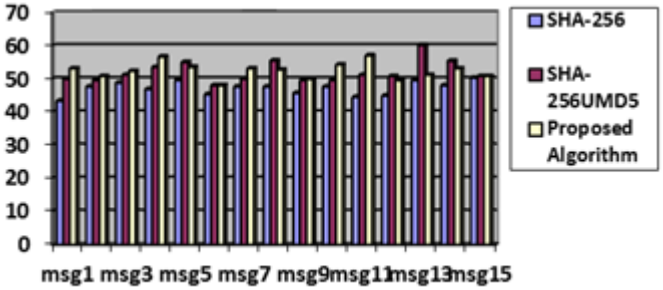

Fig5: Avalanche test with one bit difference

From figure 5 we notice that message 6 and 9 passed avalanche test with probability less than 50% for the proposed algorithm but this probability is still better than other compared algorithms. Also we notice that messages 1 , 2, 4 and message 11 passed avalanche test with probability more than 50% for SHA256UMD5 with final hash code 256 algorithm but this probability is still less than the probability of the proposed algorithm .
The shortest message(message 2) was with length 88 bits with probability percentage in proposed algorithm greater than other algorithms .The largest message(message 10) was with length 280 bits with probability percentage in proposed algorithm greater than other algorithms .
Messages with the same length ,for example messages 9 and 10, have the same length but with different probability percentage that is because proposed algorithm is more randomized than other compared algorithms .

Table 3: Avalanche test with change in more than one bit (small difference)

	Message	**Sha-256 (256bits)**	**Sha-256Md5 (256bits)**	**Sha-256Md5 (512bits) (proposed)**
1	*20* *20.*	*47.265625 %*	*51.5625%*	*51.5625%*
2	*f688b40d* *f688b40d.*	*51.171875%*	*51.171875 %*	*51.5625%*
3	*Bruteforce attack* *bruteforce attack.*	*43.359375 %*	*46.09375%*	*50%*
4	*communication lab* *communication lab.*	*47.65625%*	*52.34375%*	*52.34375%*
5	*i love my mother* *i love my mother.*	*50%*	53.125%	*53.125%*
6	*770515F5* *770515F5.*	*50%*	*50.78125%*	*51.953125%*
7	*i still have hope.* *i still have hope*	*52.29687%*	*53.90625%*	*53.90625%*
8	*7E6E2D6C* *E6E2D6C7.*	*48.4375%*	*48.4375%*	*50%*
9	*sara hakim* *sara hakim.*	*45.703125 %*	*46.8750%*	*51.5625%*
10	*.....""""""""{}{PO..* *.....""""""""{}{PO*	*48.828125 %*	*51.171875 %*	*52.734375%*
11	*combination of sha256 and md522* *combination of sha256 and md5*	*48.82800%*	*54.29687%*	*54.29687%*

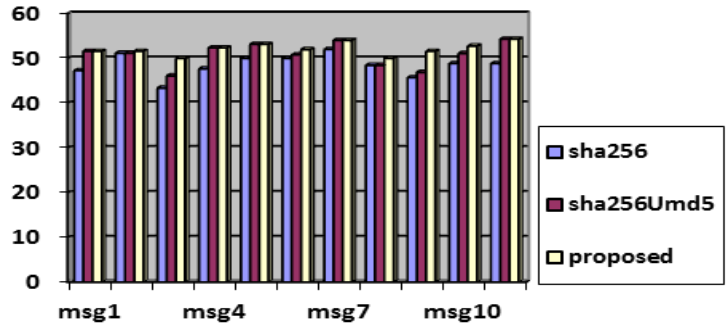

Fig 6: Avalanche test for more than one bit difference (small difference)

From figure 6 we notice that all messages passed avalanche test with probability percentage greater than 50% for proposed algorithm. Also all messages have probability percentage in proposed algorithm greater than SHA-256 algorithm .Messages 1,4,5,7 and 11 passed avalanche test with equal probability percentage to SHA-256UMD5 with final hash code length 256 bits algorithm but proposed algorithm is still better because it has final hash code length 512 bits which make it more complicated and more secure .Messages with the same length ,for example message 2, 6 and message 8, are having the same length but with different probability percentage in proposed algorithm that is because proposed algorithm is more randomized than other compared algorithms .So proposed algorithm is more complicated , secure ,randomized , immune to avalanche attack than SHA-256 and SHA-256ىMD5 with final hash code length 256 bits algorithms and bigger length than them.

6. Conclusions and future work

Here anew secure hash algorithm has been proposed based on MD5 and SHA-256 algorithms which can be used in signing applications or any message integrity as it's hash code length is 512 bits. The test results of proposed algorithm provided was more complex than the MD5, SHA256. and also was more secure and higher than the algorithms compared to it proposed algorithm passed avalanche test with probability greater than SHA-265 and SHA-256ىMD5 (with final hash code length 256)algorithms .

Proposed algorithm can be extended to have a bigger size of hash (768,1024 …) by increasing number of block size of compression function or extending them .

7. List of Abbreviations

MD	Message Digest
SHA	Secure Hash Algorithm

Acknowledgments

S. Hakim Author thanks Dr. M.fouad for his supporting and helping in calculating the functions in the algorithms and reviewing the paper.

References

[1] Praveen_Garavaram, "Cryptographic Hash Functions: Cryptanalysis Design and Application", Ph.D thesis, Information Security Institute, Faculty of Information Technology, Queensland University of Technology, 2007.
[2] W. Stalling "Cryptography and Network Security Principles and Practices", Prentice Hall, Fourth Edition, 2005 , P 353.
[3] B. Schneier., "Applied Cryptography." John Wiley & Sons, Inc., second edition, 1996.
[4] R.L. Rivest. The MD5 Message-Digest Algorithm. RFC 1321, MIT Laboratory for Computer Science and RSA Data Security, Inc., April 1992.

[5] X. Wang, H. Yu, "How to Break MD5 and Other Hash Functions", Advances in Cryptology, proceedings of EUROCRYPT 2005, Lecture Notes in Computer Science 3494,2005 , pp. 19–35.
[6] "Federal Information Processing Standards Publication 180-2 SECURE HASH STANDARD".
[7] M. Lamberger and F. Mendel, "Higher-order differential attack on reduced SHA-256", Cryptology ePrint Archive, Report 2011/037, 2011.
[8] Y. Sasaki, L. Wang, and K. Aoki., "Preimage Attacks on 41-Step SHA-256 and 46-Step SHA-512", IACR Cryptology ePrint Archive, Vol. 2009.
[9] H. Mirvaziri, K.Jumari and M.Ismail "A new Hash Function Based on Combination of Existing Digest Algorithms", The 5th Student Conference on Research and Development, SCOReD 2007, December 2007.
[10] A. kasgar, J. Agrawal and S. Sahu "New Modified 256-bit MD5 Algorithm with SHA Compression Function", International Journal of Computer Applications (0975 – 8887), Vol.42, No.12, March 2012.
[11] R. Roshdy1 International Journal of Engineering Sciences & Emerging Technologies, "Design and implementation a new security hash algorithm based on MD5 and SHA-256", August 2013.
[12] H. Dobbertin, (1996) "Cryptanalysis of MD5 compress" Announcement on Internet, 1996.
[13] NIST, "Secure Hash Standard (SHS)", FIPS PUB 180-2, 2002.
[14] J. Lee, D. Chang, E. Lee, H. Kim, D. Hong, J. Sung, S. Hong, and S. Lee, "A new 256-bit hash function DHA-256 − Enhancing the security of SHA-256," Presented at NIST Cryptographic Hash ,Workshop, 2005 .
[15] M. Juliato and C. Gebotys, "A Quantitative Analysis of a Novel SEU-Resistant SHA-2 and HMAC Architecture for Space Missions Security", IEEE Transactions on Aerospace and Electronic Systems, Vol. 49, July 2013, pp. 1536-54.
[16] G. Gupta, S. Sharma, "Enhanced SHA-192 Algorithm with Larger Bit Difference", International Conference on Communication Systems and Network Technologies (CSNT), 2013.
[17] Christoph Dobraunig; Maria Eichlseder & Florian Mendel "Analysis of SHA-512/224 and SHA-512/256" 2016.