IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661 Volume 2, Issue 2 (July-Aug. 2012), PP 18-22
www.iosrjournals.org

Evolution of Sha-176 Algorithm

Piyush Garg[1], Namita Tiwari[2]
[1,2]*(Computer Science and Engineering, M.A.N.I.T. Bhopal,INDIA)*

Abstract : *To maintain data integrity hash functions are generally used. Hash function is one type of cryptographic primitives, which provide an assurance for data integrity. NAS has designed SHA hash functions which are the set of cryptographic hash function. There are so many SHA function existing in which SHA-1 produces message digest that is of 160 bits long. Later in SHA-1 has been identified security flaws, namely that a possible mathematical weakness might exist. This point out that stronger hash function would be desirable. In this paper we are proposing a new hash function, say SHA-176 that has more strength than the existing. In this we are fulfilling basic security principle i.e. integrity. Basically this hash function is developed to improve the security. Presented results are showing the performance of the proposed SHA-176 in terms of efficiency and security.*
Keywords: *Computer Security, SHA, Hash, Message Digest*

I. Introduction

There are three type of cryptography algorithm like public key, symmetric key algorithms, and hash functions. While the first two algorithms are used for encryption and decryption of the data, and the hash functions are one-way functions that don't allow the reverse processed. To provide integrity on the messages that we send over the network we can use a hashing algorithm, these transform the text value into an alphanumeric value. Typically hashes are referred to as one way hashes; it is very difficult to reverse of hashes. There is impossible for two different messages to generate the same hash value. If the message is changed then the hash string will become invalid. Hashing is totally different from encryption process because the resulting hash is normally smaller than the original. There are three SHA algorithms, which has been structured differently and distinguished as SHA-0, SHA-1, and SHA-2. The SHA-2 family uses an identical algorithm with a variable digest size which is distinguished as SHA-224, SHA-256, SHA-384, and SHA-512 [5]-[7]. Later a new SHA algorithm was also proposed in many papers named SHA-192 [1].

In this paper we propose a new hash function say SHA-176 that focus on the security enhancement of existing SHA-1. Certain modifications are proposed in the existing one, the message digest length has been increased by extra 16 bits. There are some other changes too in the hash function in order to increase the security. Hash value generated through the hash function is appended to the message at the end at the time of transmission and the receiver authenticates that message by recompiling same hash value. If the hash value is not same at receiving end then message has been considered unacceptable. Hence the proposed modifications are enhancing the security and robustness of the existing algorithm. Organization of the paper is as follow: section two is the proposed work, section three is the result analysis and finally section four is the conclusion and future enhancement.

II. Proposed Work

2.1. Development of Proposed SHA-176

Eleven chaining variable of 16 bits is used in proposed algorithm hence the message digest generated by the hash function is of 176 bits which is 16 bits more than the SHA-1 message digest. The extended thirty two 16 bits into eighty 16 bits words are given as input to the round function. The SHA-176 algorithm has eighty steps in all and in each step there is an elementary function which calculates a message digest every time and sends it to the next step. We have proposed a new hash algorithm that undergoes a significant change in the elementary function of the secure hash algorithm and also gives us a message digest of length 176 bits. The word size is reconstructed from 32 bits to 16 bits while the number of rounds is same as that of SHA-1 [1]. In order to increase the security aspects of the algorithm the number of bits in message digest should be increased .To achieve this first, number of bits generated by message digest is considerably increased, which makes SHA-176 more complex in breaking. The modified structure of SHA-176 algorithm is given in Fig. 1

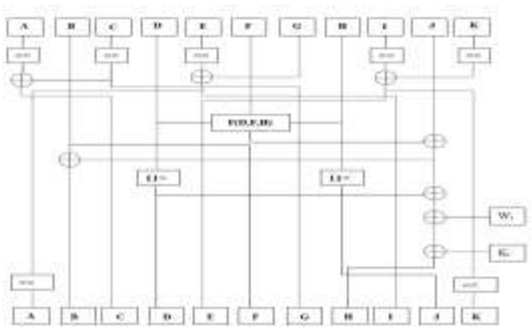

Fig.1 Proposed SHA-176 Architecture

2.2 Steps of algorithm are as follows

Step 1 Padding: The first step in SHA-176 is to add padding bits to the original message. The aim of this step is to make the length of the original message equal to a value, which is 64 bits less than an exact multiple of 512. We pad message M with one bit equal to 1, followed by a variable number of zero bits.

Step 2 Append length: After padding bits are added, length of the original message is calculated and expressed as 64 bit value and this 64 bits are appended to the end of the resultant message of step 1.

Step 3 Divide the input into 512 bit blocks: Divide the input message into blocks, each of length 512 bits, i.e. cut M into sequence of 512 bit blocks M^1, M^2…..M^N. Each of M^i parsed into thirty-two 16 bits words M^i_0, M^i_1……...M^i_{32}

Step 4 Initialize chaining variables: Before the hash function begins, the initial hash value H must be set. The hash is 176 bits used to hold the intermediate and final results. Hash can be represented as eleven 16 bits word registers, A,B,C,D,E,F,G,H,I,J,K.. Initial values of these chaining variables are:

A = 6745
B = 2301
C = EFCD
D = AB89
E = 98BA
F = DCFE
G=1032
H=5476
I=C3D2
J=E1F0
K=4038

The compression function maps 176 bits value H=(A,B,C,D,E,F,G,H,I,J,K) and 512 bit block M^i into 176 bits value. The shifting of some of the chaining variables by 11 bits in each round will increase the randomness in the bits which will change in the next successive routines. If the minimum distance of the similar words in the sequence is raised then the randomness will significantly raises. A different message expansion is employed in this hash function in such a way that the minimum distance between the similar words is greater compared with existing hash functions.

Step 5: Processing: After pre-processing is completed each message block is processed in order using following steps:

I) For i = 1 to N prepare the message schedule.
$M^i t$, $0 \le t \le 31$
Wt = (Wt-6 Wt-16 Wt-14 Wt-32) <<1
II) Initialize the eleven working variables A,B,C,D,E,F,G,H,I,J,K with (i-1)st hash value.
III) For t = 0 to 79

```
{
        Q1= ROTL2 (E)
        Q2= ROTL2 (I)
        Q3= ROTL2 (K)
        Q4= ROTL11 (D)
        A= ROTL2 (Q1 + G)
        F=B
        C= ROTL2 (B) + ROTL2 (C)
        B= (F1 (D, F, H) + J ) + B
        I= Q1 + G
        G= ROTL2 (D)
```

```
    D= Q4 + ((F1 (B, C, D) + J)
    E= Q2 + Q3
    H= J + (F1 (D, F, H) + Q4 + Wt + Kt
    J= ROTL11 (H)
    K= ROTL2 (Q2 + Q3)
}
```

Where Kt is a constant defined by a TABLE 1, F1 is a bitwise Boolean function, for different rounds defined by,
F1 (D, F, H) = IF D THEN F ELSE H
F1 (D, F, H) = D XOR F XOR H
F1 (D, F, H) = MAJORITY (D, F, H)
F1 (D, F, H) = D XOR F XOR H
Where the "IF….THEN……ELSE "function is defined by
IF D THEN F ELSE H = $(D \wedge F) \vee ((\neg D) \wedge H)$
and " MAJORITY " function is defined by
MAJ (D, F, H) = $(D \wedge F) \vee (F \wedge H) \vee (H \wedge D)$
Also, ROTL is the bit wise rotation to the left by a number of positions specified as a superscript.

IV) $H0^{(i)} = A + H0^{(i-1)}$
$H1^{(i)} = B + H1^{(i-1)}$
$H2^{(i)} = C + H2^{(i-1)}$
$H3^{(i)} = D + H3^{(i-1)}$
$H4^{(i)} = E + H4^{(i-1)}$
$H5^{(i)} = F + H5^{(i-1)}$

TABLE 1: Coefficients of each round

Rounds	Steps	F1	Kt
1	0-19	IF	FA92
2	20-39	XOR	6ED9
3	40-59	MAJ	8F1B
4	60-79	XOR	CA62

2.2. Strength of Proposed SHA-176 algorithm: It should be noted that security is linked to the ability to guess the value of hash function. Cryptanalysis of secure hash algorithm focuses on the internal structure of the compression function and is based on attempt to find efficient techniques for producing collision for single execution of the compression function. The proposed SHA-176 algorithm will satisfy the following properties

1. It is easy to compute the hash value of any input. (i.e. For any given input A, it is easy to compute h(A)).
2. Input bits of any length will be hashes to 176 bit of fixed length.
3. It is computationally infeasible to find any input which hashes to that output (i.e. to find any input say x that is equal to hash output of A (x) = h (A).)
4. It is computationally infeasible to find any second input which has same hash output, as any specified input.
5. It is computationally infeasible to find any two distinct input say A, A' which has same output (i.e. h (A) =h (A').

III. Performance Analysis

This section is providing analysis of the existing SHA and proposed SHA algorithms on the basis of different parameters like number of round; block size, maximum message size and word size all are measurable in bits. Dot Net implementation has used to test these SHA Algorithms. For experiment, Intel Core i5 2.40 Ghz, 4 GB of RAM and Window-7 Home Basic SP1, have used in which performance data is collected.

The hashing algorithms SHA-1, SHA-192 [1, 2] and Proposed SHA-176 were tested based on the security and time needed to generate message digests for the text data. Based on the simulation results, it was found that proposed SHA-176 needs more time to generate a message digest when compared with SHA-1 because the message digest generated by the proposed algorithm longer than the existing SHA-1. But as compare SHA-192 it's taking less amount of time. We have an extra block of 16 bits in the primary function and so it produces message digest of length 176 bits. Hence the security of the existing algorithm gets improved. It time to break 176 bit message digest will be more when compared with the exiting SHA-160. When comparing the bit difference, it is found that the bit difference in SHA-176 is more than existing SHA-1 after changing the single word in the message. Simulation results of compared SHA algorithm is given in TABLE 2-5. TABLE 2 is showing bits wise comparisons between existing SHA and Proposed SHA-176. TABLE 3 is showing execution time comparison between existing and proposed SHA. TABLE 4 showing the strength of proposed algorithm is more than existing, it showing the bit difference after changing a single letter in the sentence. If the bit difference is high, strength of algorithm is also high. And TABLE 5 is showing block size wise comparison between existing SHA and Proposed SHA-176.

TABLE 2: Bits wise comparison between Existing SHA-160, SHA-192 and SHA-176 algorithm

Algorithms	**Output size (bits)**	**Rounds**
SHA-176	176	80
SHA-1	160	80
SHA-192	192	80

TABLE 3: Timing Comparison between Existing SHA-160, SHA-176 and SHA-192 algorithm for 15 KB file

Algorithms	**Hash Timing (in second)**	**Message digested**
SHA-160	0.791	D4110AD3 B2F62A06 C80E3AF1 1FA81B8D F54866E6
SHA-176	0.940	FBE4 B793 BA05 02FB 2EC2 10D2 F64A 058B 0D83 10B7 2CC9
SHA-192	2.12	B32375FD 55232859 EBCD790C CF669A20 6D8B3930 1649593E

TABLE 4: Bit Difference Comparison between existing SHA-160, SHA-192 and SHA-176 algorithm after changing a single character

Message	**SHA-160**	**SHA-176**	**SHA-192**
the quick brown fox jumps over lazy dog	65	95	79
the quick brown fox jumps over lazy mog			

TABLE 5: Characteristic Comparison between Existing SHA-160, SHA-192 and SHA-176 algorithm

Algorithm	**Block size (bits)**	**Max message size (bits)**	**Word size (bits)**
SHA-176	512	$2^{64} - 1$	16
SHA-160	512	$2^{64} - 1$	32
SHA-192	512	$2^{64} - 1$	32

A graphical representation for the TABLE 3 and TABLE 4 is shown in Fig.2 and Fig.3 respectively. According to the graph in Fig.2, the execution time of Proposed SHA-176 is more than SHA-160, but very small than SHA-192. It shows that SHA-176 is highly time efficient than SHA-192. In Fig.3, the avalanche effect is high in SHA-176. When compare the bit difference after changing a single character in the message it is observed that in SHA-160 only 65 bits changed while in SHA-176 95 bits changed. This shows that the strength of SHA-176 is much higher than existing SHA-160.

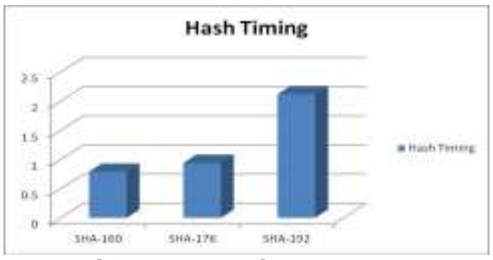

Fig.2 Execution time (in Seconds) comparison between SHA-160, SHA-192 with SHA-176

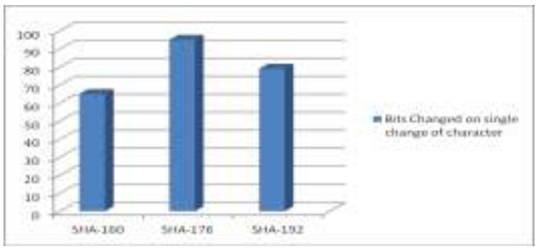

Fig.3 Bit Difference comparison between SHA-160, SHA-192 with SHA-176

3.1 Cryptanalysis of Proposed SHA-176 algorithm: It is already known that security is directly depended on the ability to guess the value of hash function. Cryptanalysis of secure hash algorithm focuses on the internal structure of the compression function and is based on attempt to find efficient techniques for producing collision for single execution of the compression function [1-3]. The proposed SHA-176 algorithm will satisfy the following properties:

1. It is easy to compute the hash value of any input. (i.e. For any given input HELLO, it is easy to compute h (HELLO)).
2. Input bits of any length will be hashes to 176 bit of fixed length.
3. It is impossible to find any input from the given Hash.
4. It is impossible to find any second input which has same hash output. As any specified input.
5. It is impossible to find any two distinct input say A, A' which has same output (i.e. h (A) =h (A').

IV. Conclusion

This paper gives the overall view about the exiting algorithms and the newly proposed of SHA-176. It focuses on the security enhancement of existing one due proposed change. The proposed SHA-176 hashing algorithm has been observed to be better than the already existing SHA-160 hashing algorithms in terms of the number of brute force attacks needed to break it and moreover it is fast when compared to the other secure hash algorithms (SHA-192). At the end we have come to the conclusion that the use of proposed system will increase the security and integrity of the message during the transmission of the message from the sender to the receiver end. Future work can be made on this to optimize the time delay.

REFERENCES

[1] L.Thulasimani and M.Madheswaran, Security and Robustness Enhancement of Existing Hash Algorithm IEEE International Conference on Signal Processing Systems 2009

[2] Ricardo Chaves, Georgi Kuzmanov, Leonel Sousa, and Stamatis Vassiliadis Cost-Efficient SHA Hardware Accelerators IEEE Transactions On Very Large Scale Integration(VLSI) Systems, VOL. 16, NO. 8, AUGUST 2008

[3] Harshvardhan Tiwari and Dr. Krishna Asawa "A Secure Hash Function MD-192 With Modified Message Expansion" (IJCSIS) International Journal of Computer Science and Information Security, Vol. VII , No. II, FEB2010 .

[4] P.P Charles & P.L Shari, "Security in Computing: 4th edition", Prentice-Hall, lnc., 2008.

[5] Charles Fleeger, Cryptography and network Security Principles and Practices.

[6] Secure Hash Standard: Federal Information Processing Standards Publication180-2

[7] Ilya Mironov, Hash functions: Theory, attacks, and applications Microsoft Research, Silicon Valley Campus.