An Implementation on Secure Hash Algorithm in Wireless Algorithms to Ensure the Integrity
Neetesh Tiwari, Amit Sinhal
Department of Computer Science & Engineering, Technocrats Institute of Technology, RGPV Bhopal
ANAND NAGAR, BHEL, 21, Bhopal, Madhya Pradesh 462022, INDIA

Abstract—“One way hash functions” plays an important role in data integrity, message authentication, and digital signature in modern information security. This paper proposed a fast one-way hash function to optimize the time delay with strong collision resistance, assures a good compression and one-way resistance. It is based on the standard secure hash function (SHA-1) algorithm. The analysis indicates that the proposed algorithm which we call OSHA is time efficient and proven for better compression function.

Keywords—Computer Security, SHA, Hash, Hash Function, Message Digest

I. INTRODUCTION
Hash functions were introduced in cryptography to provide data integrity, message authentication, and digital signature [1, 2]. A function that compresses an input of arbitrary large length into a fixed small size hash code is known as hash function [3, 4]. The input to a hash function is called as a message or plain text and output is often referred as message digest, the hash value, hash code, hash result or simply hash. Hash function is defined as: A hash function $H$ is a transformation that takes an input $m$ and returns a fixed size string, which is called the hash value $h$. One-way hash function must have the following properties: (1) one-way resistance: for any given code $h$, it is computationally infeasible to find $x$ such that $H(x) = h$, (2) weak collision resistance: for any given input $x$, it is computationally infeasible to find: $H(y) = H(x)$, and strong collision resistance: it is computationally infeasible to find any pair $(x,y)$ such that $H(y) = H(x)$. It is noted that for normal hash function with an $m$-bit output, it requires $2m$ operations to find the one way and weak collision resistance and the fastest way to find a collision resistance is a birthday attack, which needs approximately $2m/2$ operations [6,7].

The SHA-1 is called secure because it is computationally infeasible to find a message which corresponds to a given message digest, or to find two different messages which produce the same message digest. Any change to a message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify [6,7]. In this paper, a fast hash one-way function is proposed to optimize the time delay with strong collision resistance, assures a good compression and one-way resistance feature. The remainder of this paper is organized as follows. Section (2) shows methodology of proposed work. Section (3) discusses performance of the proposed hash function. Conclusions are discussed in Section (4).

II. METHODOLOGY OF OSHA
Internal structure of OSHA is different than SHA-1. OSHA algorithm uses eleven chaining variable of 16 bits and hence the message digest generated by the hash function is of 176 bits which is 16 bits more than the SHA-1 message digest. In OSHA extended thirty two 16 bits into eighty 16 bits words are given as input to the round function where as SHA-1 passes sixteen 32 bits into eighty 32 bits as input to the round function. The word size of OSHA is also different than SHA-1. SHA-1 uses 32 bits word size while OSHA uses 16 bit word size. The modified structure of OSHA algorithm is given in FIGURE 1. Steps of OSHA are as follows:

A. Padding
The first step in OSHA 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. Padding is done on message $M$ by inserting one bit equal to 1, followed by a variable number of zero bits.

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

C. Divide the input into 512 bit blocks
Dividing 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 $M_{i0}$, $M_{i1}$, …., $M_{i32}$.
D. Initialization of chaining variables
Before the hash function begins, the initial hash value \( H \) must be set. The OSHA used 176 bits buffer 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.

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 proposed hash function in such a way that the minimum distance between the similar words is greater than the existing hash functions.

E. Processing
After completion of pre-processing, each message block is processed using following steps:

**I)** For \( i = 1 \) to \( N \) prepare the message schedule.

\[
W_t = M_t, \quad 0 \leq t \leq 31. \\
W_t = (M_t^{16} \oplus M_t^{14} \oplus M_t^{10} \oplus M_t^{8} \oplus M_t^{32}) << 1, \quad 32 \leq t \leq 79.
\]

**II)** Initialize the eleven chaining variables \( A,B,C,D,E,F,G,H,I,J,K \) with \( (i-1) \)th hash value.

**III)** For \( t = 0 \) to 79

\[
\begin{align*}
\text{Temp1} &= F1(D,F,H) + J + \text{ROTL3}(B) + W_t + K_t \\
\text{Temp2} &= I \\
I &= G \\
G &= E \\
E &= \text{ROTL14}(C) \\
C &= K \\
K &= F \\
F &= \text{Temp1} \\
J &= H \\
H &= D \\
D &= B \\
B &= A \\
A &= \text{Temp2}
\end{align*}
\]

Where \( K_t \) is a constant defined by a TABLE 1, \( F1 \) is a bitwise Boolean function, for different rounds defined by:

\[
\begin{align*}
F1(D, F, H) &= \text{IF D THEN F ELSE H} \\
F1(D, F, H) &= D \oplus F \oplus H \\
F1(D, F, H) &= \text{MAJORITY}(D, F, H) \\
F1(D, F, H) &= D \oplus F \oplus H
\end{align*}
\]

Also, ROTL is the bit wise rotation to the left by a number of positions specified as a superscript.

**IV)** \( H_0(i) = A + H_0(i-1) \)

\[
\begin{align*}
H_1(i) &= B + H_1(i-1) \\
H_2(i) &= C + H_2(i-1) \\
H_3(i) &= D + H_3(i-1) \\
H_4(i) &= E + H_4(i-1) \\
H_5(i) &= F + H_5(i-1)
\end{align*}
\]

TABLE 1

<table>
<thead>
<tr>
<th>Coefficients of each round</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rounds</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

III. PERFORMANCE ANALYSIS
This section is showing a comparative analysis between four algorithms on the basis of different parameters like size in bits, number of rounds; block size, maximum message size and word size all are measured in bits. We have compared the proposed algorithms OSHA with standard SHA-1, SHA-192 [2], and SHA-192 [1]. These algorithms were tested based on the avalanche effect and timing. Dot Net implementation is used to compare these algorithms. For experiment, Intel Pentium Dual Core E2200 2.20 Ghz, 1 GB RAM and Windows-XP SP2 are used while performance data is collected.

**Timing Analysis:** Time analysis is one of the parameter used to measure the efficiency of an algorithm. An algorithm is considered efficient if it takes less time to calculate the digest. An experimental result is shown in TABLE 2. The experimental results have been taken after testing it on 100 sample files of same size for each different file size.

**TABLE 2**

| Timing Comparison between OSHA, SHA-1, SHA-192[1], and SHA-192[2] algorithms |
|-------------------------------|------------------|------------------|------------------|
| **File Size in KB**           | **OSHA**         | **SHA-1**        | **SHA-192[1]**   |
| 1 KB                          | 0.015            | 0.026            | 0.087            |
| 5 KB                          | 0.212            | 0.240            | 0.582            | 1.349            |
| 10 KB                         | 0.496            | 0.622            | 1.384            | 2.853            |
| 20 KB                         | 1.915            | 2.028            | 3.559            | 6.516            |
Graphical representation of TABLE 2 is shown in FIGURE 2. Here, color line shows the execution time in seconds of different algorithms for a 1 KB file, 5 KB file, 10 KB file and 20 KB file. After comparing OSHA with other algorithms it is clearly concluded that OSHA takes less time in comparison with other algorithms hence OSHA is more time efficient.

Security Analysis: Another important factor which is included to calculate the efficiency of hash algorithm is its security. Security of any hash algorithm can be measured with the help of avalanche effect. Avalanche effect state that two similar message having difference of single bit only produces a digest which results in 50 percent bits different from each other. It is an ideal condition, algorithms closer to this condition is considered more secure and the algorithms far from this conditions considered less secure.

To check the strength of internal structure of OSHA or to check the security of OSHA avalanche effect is calculated and compared with other algorithms.

Results retrieved after comparing OSHA with other algorithms is shown in TABLE 3 and its graphical representation is shown in FIGURE 3

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Avalanche Effect (in Percentage)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSHA</td>
<td>48.98</td>
</tr>
<tr>
<td>SHA-1</td>
<td>52.5</td>
</tr>
<tr>
<td>SHA-192[1]</td>
<td>42.078</td>
</tr>
<tr>
<td>SHA-192[2]</td>
<td>47.395</td>
</tr>
</tbody>
</table>