PAPER • **OPEN ACCESS**

A comparative study of Message Digest 5(MD5) and SHA256 algorithm

IOP Conf. Series: Journal of Physics: Conf. Series **978** (2018) 012116 doi:10.1088/1742-6596/978/1/012116

A comparative study of Message Digest 5(MD5) and SHA256 algorithm

D Rachmawati[1*], J T Tarigan[1*] and A B C Ginting [1*]

[1]Departemen Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Sumatera Utara, Jl. Universitas No. 9-A, Medan 20155, Indonesia

[*]Email: dian.rachmawati@usu.ac.id, jostarigan@usu.ac.id, realardi@gmail.com

Abstract. The document is a collection of written or printed data containing information. The more rapid advancement of technology, the integrity of a document should be kept. Because of the nature of an open document means the document contents can be read and modified by many parties so that the integrity of the information as a content of the document is not preserved. To maintain the integrity of the data, it needs to create a mechanism which is called a digital signature. A digital signature is a specific code which is generated from the function of producing a digital signature. One of the algorithms that used to create the digital signature is a hash function. There are many hash functions. Two of them are message digest 5 (MD5) and SHA256. Those both algorithms certainly have its advantages and disadvantages of each. The purpose of this research is to determine the algorithm which is better. The parameters which used to compare that two algorithms are the running time and complexity. The research results obtained from the complexity of the Algorithms MD5 and SHA256 is the same, i.e., Θ (N), but regarding the speed is obtained that MD5 is better compared to SHA256.

1. Introduction

Cryptography is the science and art which aims to maintain the security of the message [9]. The primary objectives of cryptography are authentication, integrity, and non-repudiation. The process of disguising the substance of a message called encryption. The result of encryption process is a cipher text. While the methods used to restore the cipher text to plain text is called decryption. There are three kinds of cryptographic function: hash function, private key functions, and public key functions[4].

One way hashing is the topic of cryptography [2]. One way hash function 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 merely hash of the input message [1]. A robust one-way hash function is usually expected to satisfy some requirements, namely collision resistance, preimage resistance, second preimage resistance[3]. There are a lot of hash function. Two of them are MD5 and SHA256.

The MD5 algorithm takes as input a message of arbitrary length and produces as output a 128-bit message digest of the input. The authentication algorithm computes a digest of the entire data of the secret message, used for authentication [6]. MD5 consists of 64 operations, grouped into four rounds of 16 operations [10]. The MD5 algorithm is designed to be quite fast on 32-bit machines. This algorithm found by Professor Ronald L. Rivest[2].

The SHA256 algorithm is a cryptography hash function and used in digital certificate as well as in data integrity. SHA256 is developed by N.I.S.T[5]. The SHA256 algorithm takes as input a message of arbitrary length that smaller than 2^{64} bits and produces as output a 256-bit message digest of the input[7].

2. Method

2.1. MD5 Algorithm

Step 1: Append padded bits

The message is filled so that its length is congruent to 448, modulo 512. This padding is single 1 bit added to the end of the message, followed by as many zeros are required so that the length of bits equals 448 modulo 512.

Step 2: Append length

A 64-bit representation of the message's length is appended to the result. This stage to make the message length an exact multiple of 512 bits in length.

Step 3: Divide the message

MD5 processes the input string in 512-bit blocks, divided into 16 32-bit sub-blocks. The output of the algorithm is set of four 32-bit blocks, which concatenate to form single 128-bit hash value.

Step 4 : Initialize MD Buffer

Four 32-bit variable are initialized:
A = 0x01234567
B = 0x89ABCDEF
C = 0xFEBCDA98
D = 0x76543210
These are called chaining variables.

Step 5: Process message

The main loop of the algorithm begins and continues for as many 512-bit blocks as are in the message. The four copied into the different variable: *a* gets *A*, *b* gets *B*, *c* gets *C*, and *d* gets *D*. the main loop has four rounds, all very similar. Each series 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 right a variable number of bits and adds the result to one of *a*, *b*, *c*, and *d*. Finally, the result replaces one of *a*, *b*, *c*, and *d*.
There are four nonlinear functions:
$F(X,Y,Z) = (X \wedge Y) \vee ((\sim X) \wedge Z)$
$G(X,Y,Z) = (X \wedge Z) \vee (Y \wedge (\sim Z))$
$H(X,YZ) = X \oplus Y \oplus Z$
$I(X,Y,Z) = Y \oplus (X \vee (\sim Z))$
($\vee$ is OR, $\wedge$ is AND, $\oplus$ is XOR, $\sim$ is NOT)

Step 6: Output

The message digest produced as output is A, B, C, D. That is, output begins with low-order byte of A, and end with the high-order byte of D.

2.2. SHA256 Algorithm

Step 1: Append padded bits

The message is filled so that its length is congruent to 448, modulo 512. This padding is single 1 bit added to the end of the message, followed by as many zeros are required so that the length of bits equals 448 modulo 512.

Step 2: Append length

A 64-bit representation of the message's length is appended to the result. This step to make the message length an exact multiple of 512 bits in length.

Step 3: Parsing the message

The padded message is parsed into N 512-bit message blocks, $M^{(1)}$, $M^{(2)}$,…, $M^{(N)}$, by appending 64-bit block.

Step 4: Initialize Hash Value

The initial hash value, $H^{(0)}$ is set, consist of eight 32-bit words, in a hexadecimal form.

Step 5: Prepare the message schedule

SHA256 uses a message schedule of sixty-four 32-bit words.The words of the message schedule are labeled W_0, W_1,…, W_{63}.[8]

$$Wt = \begin{cases} M_t^{(t)} & 0 \le t \le 15 \\ \sigma_1^{(256)}(W_{i-2}) + W_{i-7} + \sigma_0^{(256)}(W_{i-15}) + W_{i-16} & 16 \le t \le 63 \end{cases}$$

Where:

$\sigma_1^{(256)}(W_{i-2})$ = $((W_{i-2})$ *ROTR* 17) ⊕ $((W_{i-2})$ *ROTR* 19) ⊕ $((W_{i-2})$ *SHR* 10)

$\sigma_0^{(256)}(W_{i-15})$ =$((W_{i-15})$ *ROTR* 7)⊕$((W_{i-15})$ *ROTR* 18) ⊕ $((W_{i-15})$ *SHR 3)*

Step 6: Initialize the eight working variables, a, b, c, d, e, f, g, and h, with the(i-1)st hash value

For *t*=0 to 63:

{

T_1 = $h + \sum_1^{(256)}(e) + Ch(e,f,g) + K_1^{(256)} + W_t$

T_2 = $\sum_0^{(256)}(a) + Maj(a,b,c)$

H = *G*

G = *F*

F = *E*

E = $d + T_1$

D = *C*

C = *B*

B = *A*

A = $T_1 + T_2$

}

Where:

$\sum_{\mathbf{1}}^{(\mathbf{256})}(\boldsymbol{e})$ = (*e ROTR* 6) ⊕ (*e ROTR* 11) ⊕ (*e ROTR* 25)

$\sum_{\mathbf{0}}^{(\mathbf{256})}(\boldsymbol{a})$ = (*e ROTR* 2) ⊕ (*e ROTR* 13) ⊕ (*e ROTR* 22)

Ch(*e*,*f*,*g*) = $(e \wedge f) \oplus (\sim e \wedge g)$

Maj(*a*,*b*,*c*) = $(a \wedge b) \oplus (a \wedge c) \oplus (b \wedge c)$

Step 7 : Output

After repeating steps one through four a total of N times, the resulting hash function is $H_0^{(N)} \,||\, H_1^{(N)} \,||\, H_2^{(N)} \,||\, H_3^{(N)} \,||\, H_4^{(N)} \,||H_5^{(N)} \,||H_6^{(N)} \,||H_7^{(N)}$

3. Results and Discussions

The experiments were performed on Windows 10 Pro with Intel(R) Core(TM) i5-2430M CPU @ 2,40GHz (4CPUs),~2.4 GHz architecture, and 8.00 GB RAM. The development environment used for coding C# scripts is Visual Studio 2015. Each Algorithm was tested to three sample documents that are looped seven times. The results of the experiments of each set are presented in as follows.

3.1. MD5 Algorithm Trial

3.1.1. MD5 Algorithm First Trial

File size = 11539 bytes
Hash Value = 98B62AB5CD7D3BBF13D529FE5BDAC629
Average Running Time = 3.3644 millisecond

3.1.2. MD5 Algorithm Second Trial

File size = 22528 bytes
Hash Value = 577CA8666C87EAE17E6A053FE3812E4C
Average Running Time = 3.674242857143 millisecond

3.1.3. MD5 Algorithm Third Trial

File size = 47104 bytes
Hash Value = 14206DAB6EA6E8092D536DC36C26E040
Average Running Time = 4.7833 millisecond

3.2. SHA256 Algorithm Trial

3.2.1. SHA256 Algorithm First Trial

File size = 11539 bytes
Hash Value :
32E527EBBFD81F6B7739A87F88A4E4AB208DBD50D170DE0D0DF772A60C79529E
Average Running Time = 8.198342857143 millisecond

3.2.2. SHA256 Algorithm Second Trial

File size = 22528 bytes
Hash Value :
25B0F833564169D56741059DCCFA09ACCF891DCFC8042323090D83BD08D29FA4
Average Running Time = 10.25057142857 millisecond

3.2.3. SHA256 Algorithm Third Trial

File size = 47104 bytes
Hash Value :
42A0E1BCE1DFC2889BB0EC4A442BDFAF924C901971E187FFEFBD32734BF91CE7
Average Running Time = 16.8093142857143 millisecond

3.3. Comparison of Running Time between MD5 Algorithm and SHA256 Algorithm

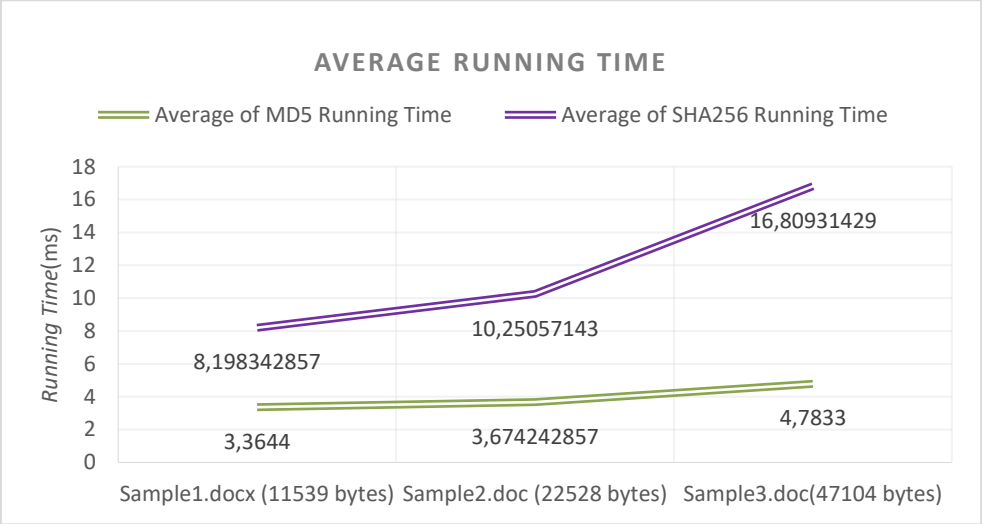

Figure 1. Graph about average running time MD5 and SHA256

It is clear that the running time of MD5 is faster than a SHA256 algorithm.

3.4. Complexity of Message Digest 5 (MD5) and SHA256

MD5 Complexity

$\sum = T(n)$
$= (C1 + 73C2 + 64C3 + C4 + 3C5)N^0 + (456C2+64C3+C4+2C6+64C7)N$
$= \Theta (N)$

SHA256 Complexity

$\sum = T(n)$
$= (29C1 + 4C2)N^0 + (10140C1 + 2C3 + 64C4 + 2C5)N^1$
$= \Theta(N)$

Both MD5 and SHA256 have a same complexity that is $\Theta (N)$

4. Conclusions

In conclusion, we know that complexity of the MD5 algorithm and SHA256 is equal and the value is $\Theta(N)$, but the running time of MD5 is faster than SHA256.

5. Acknowledgments

The authors gratefully acknowledge that the present research is supported by Fund Dissemination IPTEKS Research Results for Lecturers / Researchers Universitas Sumatera Utara.

References

[1] Gauravaram, Praveen 2007 *Cryptographic Hash Functions: Cryptanalysis Design and Application.* Ph.D. thesis, Information Security Institute, Faculty of Information Technology, Queensland University of Technology.

[2] Gupta, Piyush, and Kumar, Sandeep 2014 *A Comparative Analysis of SHA and MD5 Algorithm* International Journal of Computer Science and Information Technologies **5** (3) 4492-4495

[3] Handschub, H. and Gilbert, H. 2002 *Evaluation Report Security Level of Cryptography – SHA-256.* Technical Report, Issy-les-Moulineaux.

[4] Hossain, M.A, Islam, M. K, Das S. K., and Nashiry, M. A. 2012 *Cryptanalyzingof Message Digest Algorithms MD4 and MD5* International Journal on Cryptography and Information Security(IJCIS) **2**(1) 1-13

[5] Kasgar, A.K, Agrawal, J., and Sahu, S. 2012. *New Modified 256-bi MD5 Algorithm with SHA Compression Function.* International Journal of Computer Applications **42** (12) 47-51.
[6] Mishra, S, Mishra, S, and kumar, N. 2013. *Hashing Algorithm: MD5*. International Journal for Scientific Research and Development **1** (9) 1931-1933.
[7] NIST 2002 *Secure Hash Standard (SHS)*, FIPS PUB 180-2.
[8] Roshdy, R., Fouad, M., and Dahab, M. A. 2013 *Design and Implementation A New Security Hash Algorithm Based on MD5 And SHA-256*. International Journal of Engineering Sciences & Emerging Technologies 6 (1) 29-36.
[9] Schneir, Bruce 1996 *Applied Cryptography 2nd ed* Cryptography: Protocols, Algorithms, and Source Code in C"; 1.1 MD5 15-20.
[10] Thomas, C.G, and Jose, R.T. 2015 *A Comparative Study on Different Hashing Algorithms*. International Journal of Innovative Research in Computer and Communication Engineering **3**(7) 170-175.