A New Hash Function Based On The Hybrid Of Existing Message Digest Algorithms

Suarga Suarga, Muslim Muslim

ABSTRACT--- Hash functions are modern cryptography that have been developed by cryptographers to make security applications for protecting the integrity of data. However, almost all hash functions are either unsecure or inefficient. Therefore, another better hash function is needed to be used as an alternative hash function. In this paper, authors aim to re-explore message digest algorithms then proposes a new hash function was developed to combine the optimized internal strength elements of the algorithms, namely MD5 and SHA-1 to obtain a basic operation of the new hash that overall enhance the security, but still efficient. The proposed new hash function has been evaluated and the results shown that its execution time are slightly shorter than the combination of (SHA-192+MD5) but almost as fast as with the combination of (SHA-1+MD5). In addition, its avalanche effects exceeds 50% (the ideal condition is around 50%). These are secure enough for security applications.

Keywords: Hash function; Message digest function; Authentication; Data integrity; Merkle-Damgard construction.

I. INTRODUCTION

The hash function is one of the modern cryptologist introduced by Diffie-Hellman in 1976 to overcome authentication issues in an effort to protect the authenticity and integrity of the message [1]. The hash function compresses any size message into information known as a hash-value whose its size is always fixed and shorter than the message itself, usually 128-512 bits [2]. Besides easy to calculate, the hash functions are very sensitive to changes in input messages although only one bit changes, and is resistant to collision events [3]. Therefore, the hash function is suitable for checking the integrity of messages sent over an unsecured communication network.

There are many hash functions that have been created by cryptographers. MD4, MD5, SHA-1, and SHA-2 families (SHA-256, SHA-384, and SHA-512) are examples of dedicated hash functions that are widely used to develop information security applications [4]–[7]. MD4 is a message digest algorithm developed by R. Rivest in 1990 with a Merkle-Damgard construction whose its operation is performed iteratively by a compression function consisting of three rounds in which each round performs 16 basic operations that receives a message of any length as input and returns a 128-bit hash value as output. Although unsecure and have been breached, MD4 is widely used as a basis for designing other message digest algorithms such as MD5, SHA-0, SHA-1 and many SHA-2 families in use today.

In 1992, R. Rivest proposed MD5 with 128-bit hash-value as a more robust version of improvement than MD4. Initially, MD5 was thought to be difficult to find collisions, but there were some successful attacks that find the collisions, such as the attacks by Marc Steven and Kuznetsov [8], [9], so that MD5 is no longer regarded as a collision resistant message digest. The next attempt, the message digest algorithm SHA-1 was developed by the NSA in 1995 that produced stronger a 160-bit hash-value than MD5 although still designed with the same approach as used in making MD4 with some internal elements fixed to improve its security. However, 10 years since it was published, Wang, Ying, and Yu discovered the collision after performing 269 SHA-1 basic operations as well as by Karpman, Pierre, Peyrin, Thomas, and Stevens [10], [11]. SHA-2 is the name of a family consisting of three algorithms, namely SHA-256, SHA-384, and SHA-512. Although the structure is similar to SHA-1 but they are inefficient. Even their full versions are not proved to be breakable, but are not an option as a tool to check authenticity and test message integrity. From this brief review, there are some efficient message digest algorithms but unsecure, while the SHA-2 family is considered secure but unfortunately inefficient. Therefore, the authors propose a new hash function based on the hybrid of the existing message digest algorithm.

II. NEW HASH ARCHITECTURE

In this paper, the proposed hash function is built with hybrid architecture1 which incorporates elements of the main strength of the optimized MD5 and SHA-1 message digest algorithms. This function compresses any arbitrary message into 192-bit hash-value. This function compresses any arbitrary message into 192-bit hash-value. Prior to the hash process, the first message M is converted into message blocks: M_1, M_2, ..., M_N with a multiple of 512-bit size, in which the last block is formed by adding a number of bits, namely bits 1 and some bits 0 such that it is congruent to 448 mod 512, and then added with 64 bits representing the original message length. The author uses six 32-bit registers to accommodate intermediate hash-values and final result. Each register is initialized with a value taken from the part of the cube root of the first six primes, namely: α = 9aecce02; β = 1a5c3162; γ = 2a515f8a; δ = 30b44550; ε =

Manuscript published on 30 May 2019.
* Correspondence Author (s)
Suarga Suarga, High School Informatics and Computer Management Dipanegara, Makassar 90231, Indonesia.
Muslim Muslim, Faculty of Computer Science, Indonesia Muslim University, Makassar 90231, Indonesia.(Email: muslim@umi.ac.id)

8580a649; and λ = d1696b3d. Each 512-bit message block is processed along with the initial values of the register into a 192-bit output. This process consists of 2 rounds and each round performs 36 times basic operations such as Figure 1. The basic operation can be written in equation: u, v, w, x, y, z = (CLS(5, Ɵr(w, x, y)) + CLS(15, z) + Wk + Ck), CLS(5, u), CLS(10, v), w, x, y where u, v, w, x, y, z = six 32-bit registers containing initial values α, β, γ, δ, ε, λ; k = the basic operation iteration variable, $0 \leq k \leq 71$; Ɵr = function logic (Majority and Conditional); CLS = Circular Left Shift as much s bits; W = 32-bit word obtained from 512-bit message blocks, for $0 \leq k \leq 15$ W = Wk and W = CLS(7, W(k-16)) ^ W(k-14) ^ CLS(13, W(k-8)) ^ W(k-3) for $16 \leq k \leq 71$; Ck = the adder constant obtained from the integer part of the Sin(x) function made propagating, where $0 \leq. k \leq 71$. After rotation 71th, u, v, w, x, y, z XORed with α, β, γ, δ, ε, λ and then the algorithm processes the next message blocks. To minimize the inherent weakness of the iterative process of the Merkle-Damgard1 construction used, the authors modified the construction by adding an extra process as shown in Figure 2. The final output of this hash function is the result of concatenating the bits in α, β, γ, δ, ε, and λ. To determine the efficiency and strength of this hash function, the authors have implemented it and compared it with the combined hash function (SHA-192 + MD5)11 and the combined hash function (SHA-1+MD5) [12], [13].

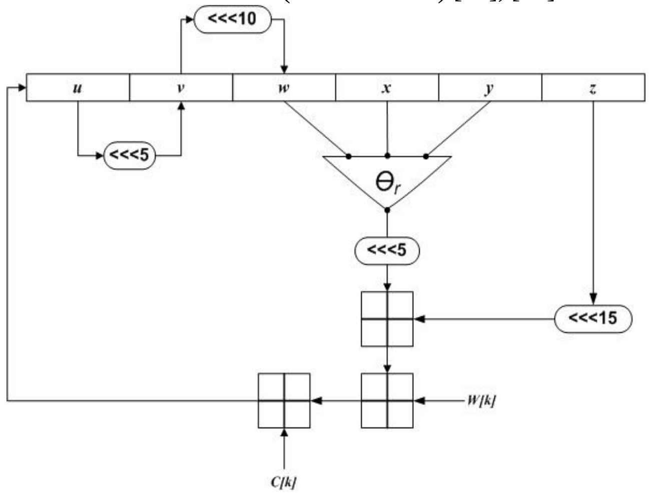

Fig. 1: Hash basic operation

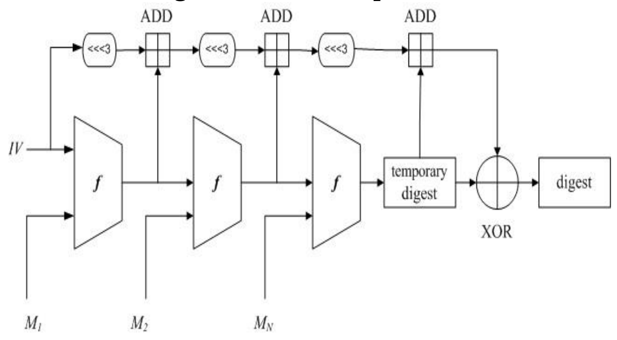

Fig. 2: Hash construction

III. RESULT AND DISCUSSION

Table 1 shows the time required by three hash functions to compute hash-values of messages of various sizes. Figure 3 is a graphical representation of the data in Table 1. Based on the data and graphs it is clear that the time required by the proposed hash function is slightly faster than the combined hash function (SHA-192 + MD5) but almost as fast as the combined (SHA-1 + MD5). Considering basic operations and post-iteration operations, it can be said that the proposed new hash function can actually be considered more efficient than its comparators.

Table 1: Time of message hash-value process

Message (kByte)	Time		
	Hash (proposed)	(SHA-1 + MD5)	SHA-192 + MD5)
0.25	2.160	2.120	2.220
1.00	2.630	2.500	2.700
1.50	2.790	2.620	2.850
2.00	2.970	2.730	3.122
3.00	3.660	3.210	3.570

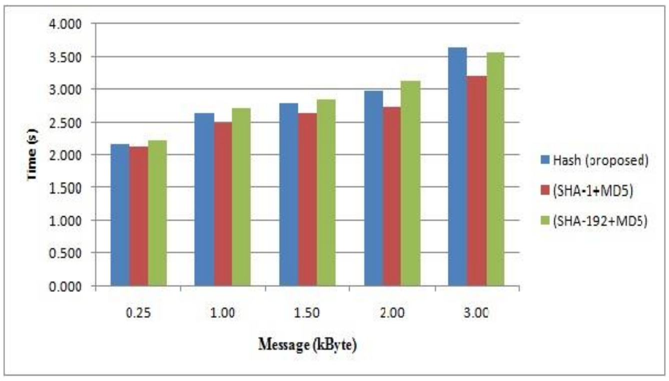

Fig. 3: Graphical represent of Table 1

Table 2 shows the avalanche effect1 of the proposed new hash function and its comparator. The Avalanche effect is an event that can show the randomizing properties of a hash function in which if the input message of a hash function is changed though it changes only 1 bit. A hash function that has a low avalanche effect (<50%) allows an attacker to guess the input message. The 192-bit hash value, a birthday attack requires the operation of at least 2^{69} hash operations.

Table 2: Avalanche test

Difference of bit number	Avalanche effect		
	Hash (proposed)	(SHA-1 + MD5)	SHA-192 + MD5)
1	52.50	45.10	46.31
5	53.65	47.31	48.42
2 Word	52.01	50.02	49.66

IV. CONCLUSION AND FUTURE NETWORK

In this paper, the authors have shown that after briefly reviewing message digest algorithms and optimizing the strength elements such as initialization values, message words, adder constants, logic and shift operations, logic functions, and number of rotations of MD5 and SHA-1 then design a basic operation that is slightly more complicated but still efficient as shown in Figure 1. So with the basic operation which is then placed at the heart of a modified Merkle-Damgard.1 construction such as Figure 2, the proposed new hash function is certain to be difficult to be breached because the avalanche effect exceeds 50% which indicates the hash function is approaching the random hash function. Further research that can be done is to decrease the execution time and at the same time improve the construction mode.

V. ACKNOWLEDGEMENT

The author wish to express my gratitude to all the lecturers who have willing to discuss about this work. I wish to thanks Dr. Ir. H. Dirgahayu A. Lantara as Dean of Computer Science Faculty who is always encouraging to write papers. I also appreciate to Indonesia Muslim University of Makassar for providing financial support.

REFERENCES

1. P. Gauravaram, "Cryptographic hash functions: cryptanalysis, design and applications," Queensland University of Technology, 2007.
2. A. P. U. Siahaan et al., "Combination of levenshtein distance and rabin-karp to improve the accuracy of document equivalence level," Int. J. Eng. Technol., vol. 7, no. 2.7, pp. 17–21, 2018.
3. R. Sriram, S. Sheeja, and H. Alexander, "Efficient data cleaning algorithm using decision tree classification model approach and modified new unique user identification algorithm using hashing techniques with a new error factor," Int. J. Eng. Technol., vol. 7, no. 1.9, pp. 54–63, 2018.
4. Z. Yong-Xia and Z. Ge, "MD5 research," in 2010 Second International Conference on Multimedia and Information Technology, 2010, vol. 2, pp. 271–273.
5. K. Matusiewicz and J. Pieprzyk, "Finding good differential patterns for attacks on SHA-1," in International Workshop on Coding and Cryptography, 2004.
6. Federal Information Processing Standards Publication (FIPS), "Secure Hash Standard." United States, 2002.
7. M. Stevens, A. K. Lenstra, and B. De Weger, "Chosen-prefix collisions for MD5 and applications," Int. J. Appl. Cryptogr., vol. 2, pp. 322–359, 2012.
8. A. A. Kuznetsov, "An algorithm for MD5 single-block collision attack using high-performance computing cluster," in IACR Cryptology ePrint Archive, 2014, p. 871.
9. X. Wang, Y. L. Yin, and H. Yu, "Finding collisions in the full SHA-1," in Annual international cryptology conference, 2005, pp. 17–36.
10. P. Karpman, T. Peyrin, and M. Stevens, "Practical free-start collision attacks on 76-step SHA-1," in Annual Cryptology Conference, 2015, pp. 623–642.
11. Q.-A. Kester, "A HYBRID CRYPTOSYSTEM BASED ON VIGENÈRECIPHER AND COLUMNAR TRANSPOSITION CIPHER," Int. J. Adv. Technol. Eng. Res., vol. 3, no. 1, pp. 141–147, 2013.
12. H. Mirvaziri, K. Jumari, M. Ismail, and Z. M. Hanapi, "A new hash function based on combination of existing digest algorithms," in 2007 5th Student Conference on Research and Development, SCORED, 2007, pp. 1–6.
13. A. Jawahir and H. Haviluddin, "An audio encryption using transposition method," Int. J. Adv. Intell. Informatics, vol. 1, no. 2, pp. 98–106, 2013.