

# The Analysis of MD5 and SHA-1 Hash Algorithms

Muhammad Rizki Fonna (13516001)<sup>1</sup>

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia

<sup>1</sup>13516001@std.stei.itb.ac.id

**Abstract**—This paper discusses the analysis of MD5 and SHA-1 hash algorithms. These algorithms are variants that develop from the development and the improvement of the previous algorithms, that are MD4 and SHA-0. MD5 hash function algorithm was developed by Ron Rivest at MIT for RSADData Security. The differences are discussed in this paper in terms of the complexity, performance, and security level of each algorithm. For the level of security, we will discuss how far cryptanalysts can penetrate and find collisions for each algorithm. A software is used to try various input variations for the MD5 and SHA-1 algorithms.

**Keywords**—MD2, SHA-1, hash algorithm, message digest

## I. INTRODUCTION

Cryptography is part of science that relies on mathematical techniques that are used as a form of security for the content of information so that the security of the content can be guaranteed and the content cannot not known by the party who wants to steal the information. In order to achieve that, security techniques or data integrity are required to safeguard information such as detecting data that has been manipulated. One example of them is to find out the authenticity of a digital image, for example a certificate. A certificate is a very important thing that is used by many people as proof that a person has completed his studies, but nowadays there are many problems that often arise and are often faced with one of them being damaged or missing. Currently, many people misuse the certificate, which can be in the form of signature forgery so that it looks like the original. The process of identifying the authenticity of a certificate is very difficult to do because the process to find out the authenticity can only be done by the party entitled to the certificate. The method used to verify the authenticity of certificate data at this time is by making direct contact with the party making the certificate regarding the authenticity of the certificate and the signed legalization process of the party making the certificate. One of these phenomena is in the problem in education. In cryptography there is a function that is suitable for security, one of which is authentication [2]. The function is used as a hash function. A hash function algorithm is called as digest algorithm. Hash functions are widely used in the process of digitally signing documents, but hash functions can usually be used widely in a variety of applications. The hash function that is often used are Message Digest (MD) and Secure Hashing Algorithm (SHA). Broadly speaking, the researchers compared the MD2 and SHA-1 methods, in both methods there is a comparison in the Hash

function, namely MD2 message block size is 128 bits, while SHA-1 message block size is 512 bits.

## II. HASH FUNCTION ALGORITHMS

### A. Hash Functions

Hash functions are very useful and appear in almost all information security applications, not only in the cryptography world. Practical applications include message integrity checking, digital fingerprinting, authentication, and various other information security applications using hash functions.

Hash function is a mathematical function that converts numeric input values into compressed numeric values. Its purpose is to compress the input numeric value. The hash function input has various lengths, but the hash value output will always have a fixed length. The value returned by the hash function is called a message digest or just the hash value.

Cryptographic hash functions are designed to prevent returning the hashed checksum back to the original text. However, while it is almost impossible to reverse it, that doesn't mean a hash value is completely guaranteed to be secure.

As there are many software that allows to re-translate hash functions, such as Rainbow Table which can be used to crack the plaintext checksum. Rainbow Table is basically a dictionary that takes out thousands, millions, or even billions of these in addition to the corresponding plaintext values.

But the reality is that the Rainbow Table sometimes doesn't list every possible checksum, usually it only "helps" find simple phrases like weak passwords. But passwords that are more complex or contain symbols may not work. That's why only those who memorize the passwords know.

Types of Hash function algorithms:

- a) MD2 (Message-Digest 2)
- b) MD4 (Message-Digest 4)
- c) MD5 (Message-Digest 5)
- d) RIPEMD
- e) RIPEMD-128/256
- f) RIPEMD-160/320
- g) SHA-0 (Secure Hash Algorithm)
- h) SHA-1 (Secure Hash Algorithm)
- i) SHA-256/224 (Secure Hash Algorithm)
- j) SHA-512/384 (Secure Hash Algorithm)
- k) WHILPOOL

The MD4 and MD5 algorithms were developed by an MIT

Professor named Ronald L. Rivest. The term "MD" which is used is the abbreviation of Message Digest. The development of MD5 has gone through 5 revisions, where the first and second generation MDs are designed to help the RSA algorithm compute the signature of secret messages that will be sent and encrypted by RSA. The third & fourth generation of MD came about due to competition from another hash algorithm called Snefru, which has a speed advantage in the computation process over MD2. In 1992, a security gap was discovered in both Snefru and MD4, then Professor Rivest immediately overcame this security gap and replaced it with the fifth generation Message Digest, namely MD5. Out of these five generations, the first & third generation MDs were not published because they had weak points.

Cryptographic hash functions must have the following properties:

1. The same message always produces the same hash value (ie the function is deterministic).
2. The hash value is calculated on the fly.
3. It is impossible to have two messages with the same hash value (known as a "collision").
4. It is impossible to intentionally create messages that return a specific hash value.
5. Slight changes to the message should change the resulting hash value widely, so that it doesn't appear to be correlated with the original hash.

The most commonly used cryptographic hash functions include MD5, SHA-1, and SHA-2.

The uniqueness of each hash is critical to the integrity of the cryptographic hash function. This is what really sets cryptographic hash functions apart from other hash functions - the certainty that a particular message is identified in a unique and non-duplicable way.

Digital signature schemes (such as for document signing, code signing, or S / MIME e-mail) generally require that a cryptographic hash be computed from the message and included in the signature. The receiving software then independently computes the hash to verify message integrity.

Websites also frequently publish hash values for downloadable files. When users download files, they can use their own software to independently compute hashes, verify file integrity.

Password security also relies on cryptographic hashes. The password presented by the user is hashed and then compared to the stored hash.

Cryptographic hash functions are widely used in security protocols such as SSL / TLS and SSH, and in other applications that rely on data integrity. Cryptocurrencies use a hashing algorithm to update the blockchain with new blocks of secure and verifiable transaction data. (BitCoin, for example, uses SHA-2 for transaction verification.)

#### B. Secure Hash Algorithm-1 (SHA-1)

SHA was developed by the National Institute of Standards and Technology (NIST) and published as Federal Information Processing Standards (FIPS 180) in 1993. The Secure Hash Standard (SHS) specifies SHA-1 to compute the hash value of a

message or file. SHA-1 has a maximum message length of  $2^{64}$  bits and has an output of 160 bits which is called a message digest or hash code. The message digest can be used as input for the Digital Signature Algorithm (DSA), which is used to produce a signature to verify the message. When a weakness was found in the SHA-0 algorithm, various revisions and improvements were made to make a better algorithm. The results of these improvements were published in 1995 and were used as a reference for making the SHA-1 algorithm. The SHA-1 algorithm is a technical revision of the SHA algorithm. The SHA-1 algorithm can be said to be safe because the calculation process does not allow to find the actual message from the resulting message digest. Any changes that occur to messages while sending will result in a different message digest. The SHA-1 algorithm is based on the MD4 algorithm and its design is very similar to that algorithm.

The SHA-1 algorithm is a technical revision of the SHA algorithm. The SHA-1 algorithm can be said to be safe because the calculation process does not allow to find the actual message from the resulting message digest. Any changes that occur to messages while traveling will result in a different message digest. The SHA-1 algorithm is based on the MD4 algorithm and its design is very similar to the algorithm.

SHA-1 is only one of four algorithms in the Secure Hash Algorithm (SHA) family. Most of them were developed by the US National Security Agency (NSA) and published by the National Institute of Standards and Technology (NIST).

SHA-0 has a message body size of 160-bit (hash value) and is the first version of this algorithm. The SHA-0 hash value is 40 digits. It was published under the name "SHA" in 1993 but fell into disuse in many applications as it was quickly replaced by the SHA-1 in 1995 due to a safety flaw.

SHA-1 is the second iteration of this cryptographic hash function. SHA-1 also has a 160 bit message digest and attempts to increase security by fixing a vulnerability found in SHA-0. However, in 2005, SHA-1 was also found to be unsafe.

After cryptographic flaws were discovered in SHA-1, NIST made a statement in 2006 encouraging federal agencies to adopt the use of SHA-2 in 2010. SHA-2 was stronger than SHA-1 and attacks carried out against SHA-2 were unlikely to have occurred with computing power today.

#### C. Secure Hash Algorithm-2 (SHA-2)

SHA 2 is a development of the SHA 1 algorithm and is a development project of the United States government. Cryptographers modify the algorithm with two hash functions at once, namely SHA 256 and SHA 512. Each hash uses 32 bits and 64 bits of words in the encryption process.

SHA 2 has a block encryption size of up to 1024 bits, which is the bit length of the hash encryption when sending data online. SHA 2 was developed from the weakness of SHA 1 which made the encryption value collision process too frequent and resulted in encryption failure.

One of the standard attacks for the development of SHA 1 into SHA 2 is a brute force attack (the most popular password target hacker attack used by hackers). The bit length given by SHA 1 is still not sufficient to protect against brute-force attacks, so SHA 2 appears with a hash bit length exceeding SHA 1.

SHA-256 was designed by The National Institute of Standards and Technology (NIST) in 2002. SHA-256 produces a message digest with a length of 256 bits. SHA-256 is a one-way hash function, because it is impossible to find messages from the resulting message digest.

The steps for making a message digest with SHA-256 are as follows:

1. Adding padding bits.
2. Increase the length of the original message.
3. Initialize the initial hash value.
4. Messing messages in 512 bit blocks.

SHA is used to compute message diggest from the message or data file provided as input. A message or file is considered a collection of bits. The length of the message is the number of bits in the message (empty messages have length 0). If the number of bits in the message is a multiple of 8, it can be displayed in hexadecimal format for easy reading.

The purpose of message padding is to make the total length of the message body a multiple of 512 bits. SHA sequentially processes 512 bit blocks when counting the message digest. In the message padding, add a single "1", followed by an m "0" followed by a 64-bit integer at the end of the message to produce a message  $512 * n$  in length. The 64-bit integer is the length of the original message before message padding.

SHA-2 includes significant changes from its predecessor, SHA-1. The SHA-2 family consists of six hash functions with digests (hash values) that are 224, 256, 384 or 512 bits: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256. SHA-256 and SHA-512 are novel hash functions computed with 32-bit and 64-bit words, respectively. They use different shift amounts and additive constants, but their structures are otherwise virtually identical, differing only in the number of rounds. SHA-224 and SHA-384 are truncated versions of SHA-256 and SHA-512 respectively, computed with different initial values. SHA-512/224 and SHA-512/256 are also truncated versions of SHA-512, but the initial values are generated using the method described in Federal Information Processing Standards (FIPS) PUB.

SHA-2 was first published by the National Institute of Standards and Technology (NIST) as a U.S. federal standard (FIPS). The SHA-2 family of algorithms are patented in US patent. The United States has released the patent under a royalty-free license.

With the publication of FIPS PUB 180-2, NIST added three additional hash functions in the SHA family. The algorithms are collectively known as SHA-2, named after their digest lengths (in bits): SHA-256, SHA-384, and SHA-512.

The algorithms were first published in 2001 in the draft FIPS PUB 180-2, at which time public review and comments were accepted. In August 2002, FIPS PUB 180-2 became the new Secure Hash Standard, replacing FIPS PUB 180-1, which was released in April 1995. The updated standard included the original SHA-1 algorithm, with updated technical notation consistent with that describing the inner workings of the SHA-2 family.

In February 2004, a change notice was published for FIPS PUB 180-2, specifying an additional variant, SHA-224, defined to match the key length of two-key Triple DES. In October 2008,

the standard was updated in FIPS PUB 180-3, including SHA-224 from the change notice, but otherwise making no fundamental changes to the standard. The primary motivation for updating the standard was relocating security information about the hash algorithms and recommendations for their use to Special Publications 800-107 and 800-57. Detailed test data and example message digests were also removed from the standard, and provided as separate documents.

In January 2011, NIST published SP800-131A, which specified a move from the then-current minimum of 80-bit security (provided by SHA-1) allowable for federal government use until the end of 2013, to 112-bit security (provided by SHA-2) being both the minimum requirement (starting in 2014) and the recommended security level (starting from the publication date in 2011).

In March 2012, the standard was updated in FIPS PUB 180-4, adding the hash functions SHA-512/224 and SHA-512/256, and describing a method for generating initial values for truncated versions of SHA-512. Additionally, a restriction on padding the input data prior to hash calculation was removed, allowing hash data to be calculated simultaneously with content generation, such as a real-time video or audio feed. Padding the final data block must still occur prior to hash output.

In July 2012, NIST revised SP800-57, which provides guidance for cryptographic key management. The publication disallowed creation of digital signatures with a hash security lower than 112 bits after 2013. The previous revision from 2007 specified the cutoff to be the end of 2010. In August 2012, NIST revised SP800-107 in the same manner.

The NIST hash function competition selected a new hash function, SHA-3, in 2012. The SHA-3 algorithm is not derived from SHA-2.

#### D. Secure Hash Algorithm-3 (SHA-3)

In 2006, NIST held a hash function competition to create a new hash standard, namely SHA-3. SHA-3 was not made to replace SHA-2, because no major attacks had yet occurred on SHA-2. NIST made SHA-3 out of concern because MD5, SHA-0, SHA1 were successfully breached. That's why NIST is looking for an alternative hash algorithm that is very different from the previous algorithm, namely SHA-3. In 2012, Keccak became the winner in this competition. Then in 2014 published a draft of FIPS 202 concerning "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions". After that in 2015 SHA-3 was formalized as the new standard of hash function (Charles, 2015).

Unlike SHA-1, SHA-3 has a variety of output sizes. The types of SHA-3 outputs are SHA-3 224, SHA-3 256, SHA-3 384, SHA-3 512, SHAKE128 and SHAKE256. Due to these various types of output, SHA-3 is included in the sponge function type. In the journal entitled "Cryptographic sponge function", the sponge function provides a specific way to generalize hash functions with various output results.

SHA (Secure Hash Algorithm) is a one-way hash function created by NIST and used with DSS (Digital Signature Standard). SHA is defined by the NSA as a standard one-way hash function. SHA is based on the MD4 made by Ronald Linn Rivest. SHA-3 is a one-way hash function algorithm that uses

sponge construction using permutation functions which include Initialization Functions, Absorbing Functions, and Squeezing Functions that ensure data does not change during transmission. SHA-1 is currently the most widely used type of hash function. With advances in technology, the security level of SHA-1 is vulnerable to attacks by hackers so that the NSA (National Security Agency) developed the second version of SHA consisting of several hash functions, namely SHA-224, SHA-256, SHA-384, and SHA-512.

The SHA-3 algorithm is an advanced algorithm from the SHA-1 and SHA-2 algorithms. The SHA-3 algorithm was selected by NIST (National Institute of Standards and Technology) through an open competition to develop a new hash algorithm. The competition for the new SHA-3 algorithm was announced in 2007 and ended in October 2012. Several finalists for the new SHA-3 algorithm included: BLAKE, Grøstl, JH, Keccak, and Skein with Keccak as the winner and designated as the new SHA-3 algorithm. The Keccak algorithm was created and designed by Guido Breton, Joan Daemen, Michaël Peeters and Gilles Van Assche. The name Keccak comes from Kekak, a Balinese dance. Keccak differs from other SHA3 finalists in that it uses a sponge construction, where each input message block will be XORed with the previous block bitrate and state to be passed into the speed permutation function. (absorbing & squeezing phase). If other designs depend on compression function, Keccak uses non-compression function to absorb and then compress short digest.

The Keccak algorithm accepts three input parameters, namely bitrate ( $r$ ), capacity ( $c$ ), and diversity ( $d$ ). In general, the process of keccak is:

1. Process message input ( $P$ ), which is to apply padding to the message input.
2. Splitting the input message into  $P_0, P_1, p_2, \dots, P_i$ , where  $I$  = the number of multiples of the bitrate length for the length of the input message.
3. Absorbing all fractions of the input message.
4. Squeezing as much as  $j$ , where  $j$  = the multiple of output length  $r/w$  to meet the desired output length.
5. Output is concatenation of Squeezing output in a certain bitrate range. The state in keccak is a series of bits which is seen as a three-dimensional array of these bits.

The sponge function in keccak is based on sponge construction. The sponge construction is a simple iterative construction to construct a variable sponge function. The length of the input and the length of the output varies depending on the length of a fixed transformation or permutation operating in a number of bits.

In general, there are two phases in sponge construction, namely the phase

absorbing and squeezing phase.

1. The absorbing phase is the phase where the process is carried out on all fractions of the input input and XORed with the bitrate portion of the state then passed into the function  $f$ .

2. Squeezing phase, is the phase to get the output results. In this phase, a certain number of bits of the function  $f$  is concatenated so that the number of concatenation bits is the same as the desired number of concatenation bits.

#### E. Message Digest-2 (MD-2)

In 1989 Message Digest 2 was designed for 8-bit based computers. MD2 converts the message into Hexadecimal form as much as 32 bits, and compresses 128 message bits into blocks measuring 16 bytes after which check is added.

The MD2 algorithm was developed by Ron Rivest in 1989. This algorithm is optimized by using an 8-bit computer. MD2 is actually specified in RFC 1319. The MD2 algorithm generates a 128-bit hash value and accepts an input message of an unspecified length. The message that will be used as input for this hash function will be padded first.

Suppose we have a  $b$ -byte of messages as input, and we expect to be able to get a message digest of that message. In this case,  $b$  is an arbitrary integer that is positive, it can also be zero, and is of any magnitude.

Below will be explained the 4 stages of the process to generate a message digest for the MD2 algorithm.

##### 1. Enter the padding byte or message length

The message length is added by the congruent message length process  $0 \text{ mod } 16$ , so that the length will be expanded to a multiple of 16 bytes. The message length should be at least 1 to 16 bytes. at this stage the message results have a multiple of 16 bytes in length

##### 2. Enter the Checksum

Entering a check sum of 16 bytes and added from the results of the previous step, then followed by a 256 byte step and generated randomly

##### 3. Initialize the MD Buffer

To produce a message digest, an X 48 byte storage is used, and the value is initialized to zero.

##### 4. Process messages in 16-byte blocks

This process uses the generated numbers of 256 bytes. And the following is a  $p_i$  value of 256 bytes.

#### F. Message Digest-5 (MD-5)

MD5 (Message-Digest algorithm 5) is a widely used cryptographic hash function with a 128-bit hash value. On the Internet standard (RFC 1321), MD5 has been used in a variety of security applications, and MD5 is also commonly used to test the integrity of a file. MD5 was designed by Ronald Rivest in 1991 to replace the previous hash function, MD4.

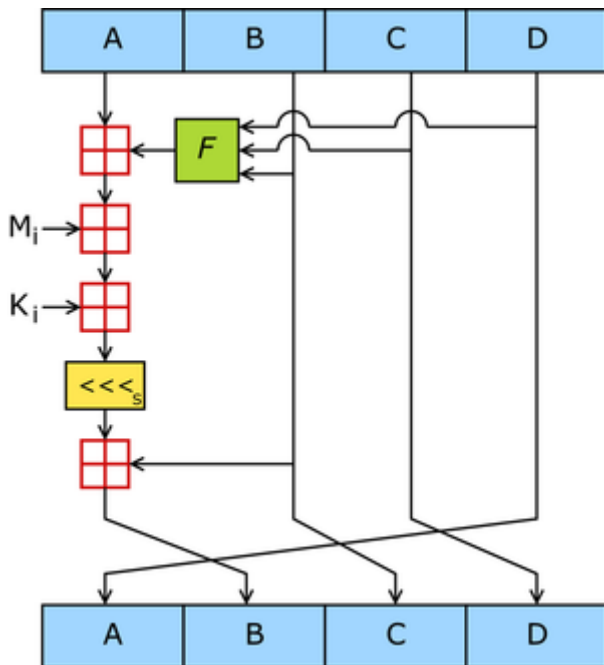


Figure 1 The overview of MD5 Algorithm

In MD5 algorithms, there are several steps that have to be done, such as:

#### 1. Append Padding Bits

Padding means adding extra bits to the original message. So in MD5 original message is padded such that its length in bits is congruent to 448 (mod 512). Padding is done such that the total bits are 64 less being a multiple of 512 bits length.

Padding is done even if the length of the original message is already congruent to 448 modulo 512. In padding bits, the only first bit is 1 and the rest of the bits are 0.

#### 2. Append Length

After padding, 64 bits are inserted at the end which is used to record the length of the original input. Modulo  $2^{64}$ . At this point, the resulting message has a length multiple of 512 bits.

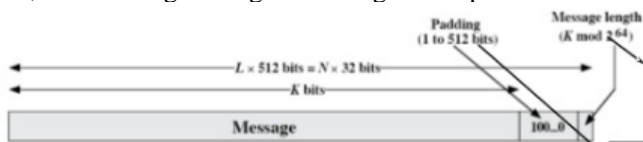


Figure 2 Append of Padding Bits into Original Message

#### 3. Initialize MD buffer

A four-word buffer (A, B, C, D) is used to compute the values for the message digest. Here A, B, C, D are 32-bit registers and are initialized in the following way

A = 01234567  
 B = 89ABCDEF  
 C = FEDCBA98  
 D = 76543210

#### 4. Processing message in 16-word block

MD5 uses the auxiliary functions which take the input as three 32-bit number and produces a 32-bit output. These functions use logical operators like OR, XOR, NOR.

| Name  | Notation   | $g(b,c,d)$                            |
|-------|------------|---------------------------------------|
| $f_F$ | $F(b,c,d)$ | $(b \wedge c) \vee (\sim b \wedge d)$ |
| $f_G$ | $G(b,c,d)$ | $(b \wedge d) \vee (c \wedge \sim d)$ |
| $f_H$ | $H(b,c,d)$ | $b \text{ XOR } c \text{ XOR } d$     |
| $f_I$ | $I(b,c,d)$ | $c \text{ XOR } (b \wedge \sim d)$    |

The content of four buffers are mixed with the input using this auxiliary buffer and 16 rounds are performed using 16 basic operations.

#### 5. Output

After all, rounds have performed the buffer A, B, C, D contains the MD5 output starting with lower bit A and ending with higher bit D.

Wherever Times is specified, Times Roman, or Times New Roman may be used. If neither is available on your word processor, please use the font closest in appearance to Times that you have access to. Please avoid using bit-mapped fonts if possible. True-Type 1 fonts are preferred. Type your main text in 10-point Times. Be sure your text is fully justified—that is, flush left and flush right. Please do not place any additional blank lines between paragraphs.

### III. ANALYSIS OF SHA-1 AND MD5 ALGORITHM

#### A. Advantages and Disadvantages of MD5

In terms of advantages, MD5 has several of them, such as can be used to verify file integrity, functions as a file change detection, store passwords, and is very sensitive to the slightest data changes.

Every algorithm has disadvantages and weaknesses, including MD5 also, such as vulnerable to collision attacks, preimage attack, and advanced cryptanalysis, and the process of changing the original data to MD5 takes a relatively long time (hardware resource).

Large figures and tables may span both columns. Place figure captions below the figures; place table titles above the tables. If your figure has two parts, include the labels "(a)" and "(b)" as part of the artwork. Please verify that the figures and tables you mention in the text actually exist. Please do not include captions as part of the figures. Do not put captions in "text boxes" linked to the figures. Do not put borders around the outside of your figures. Use the abbreviation "Fig." even at the beginning of a sentence. Do not abbreviate "Table." Tables are numbered with Roman numerals.

#### B. Advantages and Disadvantages of SHA-1

##### a. Advantages

SHA 1 is often used because it has several advantages, including:

Used for validating a password; The hash value of the password will be stored, then when the password is authenticated, the password entered by the user will be calculated the hash and if the hash is correct, the password is declared valid. However, to get the original password cannot be obtained from the hash that has been stored.

Challenge handshake authentication; To avoid sending a password in a "clear" condition, the client can send the hash value of a password over the internet to be validated by the

server without risking the original password being intercepted.

Large figures and tables may span both columns. Place figure captions below the figures; place table titles above the tables. If your figure has two parts, include the labels “(a)” and “(b)” as part of the artwork. Please verify that the figures and tables you mention in the text actually exist. Please do not include captions as part of the figures. Do not put captions in “text boxes” linked to the figures. Do not put borders around the outside of your figures. Use the abbreviation “Fig.” even at the beginning of a sentence. Do not abbreviate “Table.” Tables are numbered with Roman numerals.

Anti-tamper; to ensure that the data does not change during transmission. The recipient will calculate the hash value and match the hash that was sent, if the value is the same it means that the data sent has not changed.

Digital signatures; is done by encrypting the hash value of a document using a private key, resulting in a digital signature for the document. Someone else can check the document's authentication by decrypting the signature using a public key to get the original hash value and compare it with the hash value of the text.

### C. Comparison Between SHA-1 and MD5

Because SHA-1 and MD5 were developed or derived from MD4, they both have similarities with each other, both in strength and characteristics. Here are the differences:

First, the author analyse about the security against brute-force attacks. The most important point is that SHA-1 produces 32-bit diggest longer than MD5. With brute-force, SHA-1 is stronger than MD5.

Next, security against cryptanalysis. The weakness of MD5 is in the design so that it is easier to do cryptanalysis than SHA-1

Third, speed. Both algorithms work on modulo 232 so they both work well on 32 bit architectures. SHA-1 has more steps than MD5 (80 versus MD5 64) and must process 160 bit buffers compared to MD5 128 bit buffers, so SHA-1 runs slower than MD5 on the same hardware.

Fourth, simplicity. Both algorithms are simple to describe and easy to implement because they do not require large programs or large substitution tables.

## V. CONCLUSION

There are many hash functions that can be used out there. Several of them are SHA-1 and MD5. Both of them are improvement from MD4 algorithm, so they have pretty much similarities, except that MD5 uses 80 rounds compared to 4 rounds in SHA-1. Both of them are quite fast in term of speed but there has been collition found in SHA-1. So it can be said that MD5 is safer than SHA-1 overall.

## VII. ACKNOWLEDGMENT

First, the writer thanks God because with His mercy, the report on the assignment of Kriptografi can be done by the author and the writer completes it on time. The author also thanked Mr. Dr. Ir. Rinaldi Munir, MT. as the author's lecturer in the Kriptografi for all the guidance and teaching she has given

to the author

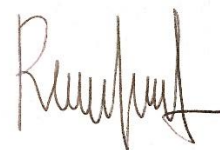
## REFERENCES

- [1] R. Munir, "Fungsi Hash," Teknik Informatika STEI - ITB, Bandung.
- [2] R. Munir, "Fungsi Hash MD5," Teknik Informatika STEI - ITB, Bandung.
- [3] R. Munir, "Fungsi Hash SHA," Teknik Informatika STEI - ITB, Bandung.
- [4] R. Munir, "Fungsi Hash SHA-3," Teknik Informatika STEI - ITB, Bandung.
- [5] Search Security, "What is Cryptography," [Online]. Available: <https://searchsecurity.techtarget.com/definition/cryptography#:~:text=Cryptography%20is%20a%20method%20of,%22%20stands%20for%20%22writing.%22>. [Accessed 17 Dec 2020].
- [6] Tutorials Point, "Cryptography Hash Functions," [Online]. Available: [https://www.tutorialspoint.com/cryptography/cryptography\\_hash\\_functions.htm](https://www.tutorialspoint.com/cryptography/cryptography_hash_functions.htm). [Accessed 18 Dec 2020].

## PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 21 Desember 2020



Muhammad Rizki Fonna  
13516001