Combination of MD5 and ElGamal in Verifying File Authenticity and Improving Data Security

[1] Muhammad Iqbal, [2] Andysah Putera Utama Siahaan, [3]Riska Putri Sundari
[1]Faculty of Science and Technology, Universitas Pembangunan Panca Budi, Medan, Indonesia
[2]Degree Student, Faculty of Science and Technology, Universitas Pembangunan Panca Budi, Medan, Indonesia

Abstract: *This study aims to maintain the authenticity of the data to assure the recipient that the data is free from modifications made by other parties, and if there is a modification to the data, then the recipient will know that the data is no longer maintained authenticity. The digital signature technique is used by using a combination of the MD5 algorithm as a hash function algorithm to generate the message digest, and ElGamal algorithm as a public key algorithm, with the combination of the two algorithms, will be generated the digital signature of each data that will be preserved. The ElGamal algorithm is used to encrypt message digest from the process of the MD5 algorithm to files where the digital signature is the result of encryption of message digest. The private key is used for the encryption process, while the public key is used to process the digital signature description when the file testing process is received. The results of the combination of the two algorithms are implemented in a web-based application that is built using the PHP programming language. Data used as input can be in the form of files of any format that has a maximum size of 100 MB. The results showed that a combination of the two algorithms provides digital signatures that have a high level of security and with reasonably fast processing time.*

Key Words: *MD5, ElGamal, encryption, decryption, hash function, digital signature.*

1. INTRODUCTION:

In many scenarios, the sender and recipient of the message need a guarantee that the message has not been modified during the transmission, intentionally or unintentionally. Although hiding a message in the form of encryption, it is still possible to change an encrypted message without having to understand it. However, if the message is digitally signed, any changes in the message after signing will cancel the signature. A hash function algorithm is needed to generate digital signatures. The hash function algorithm extracts the message digest from a file. Each file has a different message digest. This digest message is then encrypted with a public key algorithm using a private key. It is so that only the sender of the message can generate digital signatures from the file before sending it. The sender of the message will guarantee the recipient that the message is free from modification as long as it is transmitted. It is possible because the private key is very confidential which is only known by the sender of the message. While the recipient of the message will hold the public key or public key that is used to check the authenticity and compatibility of the message with the digital signature provided. If the digital message and signature match, then it is certain that the message does not change when transmitted. Based on the problems that have been described, in a digital signature generation at least a hash function algorithm is needed and a public key cryptographic algorithm that will be combined and used to generate digital signatures that are safe from a message. MD5 algorithm (Message-Digest-5) is one of the hash function algorithms that can be used in producing a digital signature of data while the ElGamal is one of the most reliable public key algorithms used for encryption.

2. THEORIES:

2.1 Digital Signature

A digital signature is a signature that is performed using an electronic device that functions the same as a manual signature but differs in its application. The digital signature in question is not a signature digitized using a scanner, but a cryptographic value that depends on the message and the sender of the message. With a digital signature, data integrity can be guaranteed, besides that, it is also used to prove the origin of the message and non-denial. A digital signature is a sign that is made to mark a data or message to ensure the authenticity of the document sent is usually a digital signature is a unique digit number based on the specific algorithm which is encrypted to ensure confidentiality. Based on its history, the use of digital signatures begins with the use of cryptographic techniques used to secure information that is to be transmitted/delivered to other people who have been used for hundreds of years. In a cryptography, a message is encrypted (encrypted) using a key. The result of this encryption is in the form of the ciphertext and then transmitted/submitted to the intended destination. The ciphertext is then opened/decrypted with a key to get the information that has been encrypted. There are two kinds of ways to do encryption, such as using symmetrical cryptography and symmetrical cryptography which is then known as public key cryptography [1].

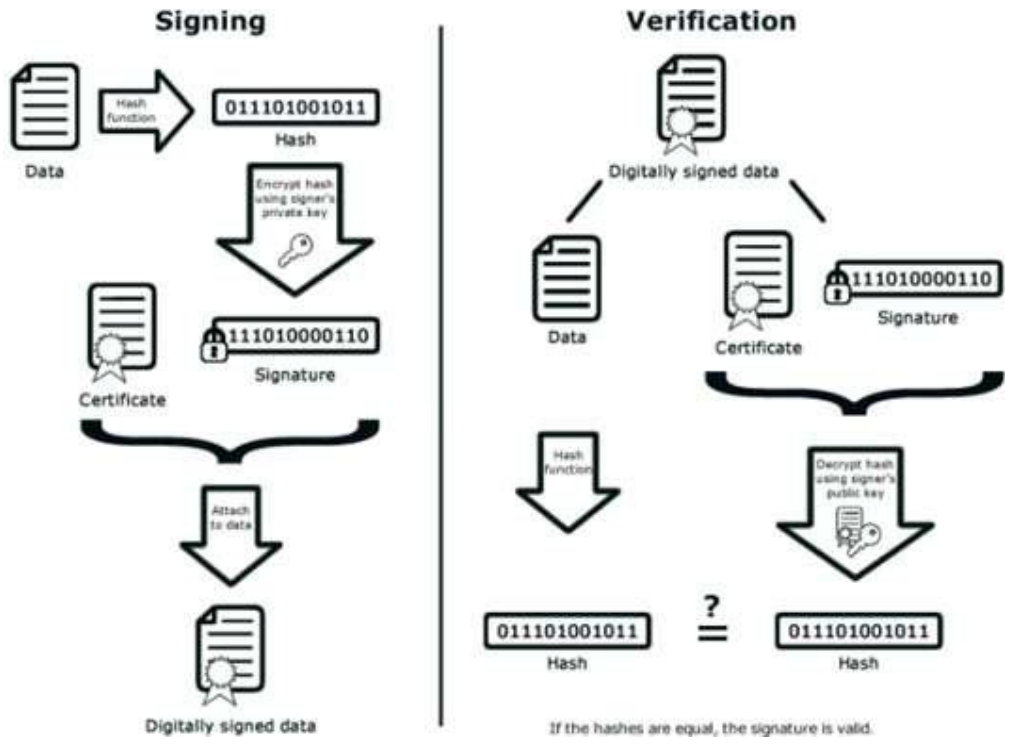

Figure 1. An overview of the concept of Digital Signature

Figure 1 describes the concept of digital signatures. There are two things discussed such as signing and verification. It is commonly used for marking documents from where or whom the document comes from that can be proven authenticity and validity using specific verification [2].

2.2 Hash function

The Hash function is a function that accepts input strings that are arbitrary in length and then transforms into fixed length output strings and are generally much smaller than the original string size. Next to this is a hash function scheme where the size of the input is arbitrary, but produces a fixed size output [3]–[5] In the hash function, there is the term one-way function which is a hash function that works in one direction. It means that the message that has been converted into a digest message cannot be returned to the original message (irreversible). The hash function is often called a one-way encryption function, or the message diggest is also called. The hash function is used to guarantee authentication services and the integrity of a message or file. A hash function h maps bits of any length to a string of a certain length eg n. With domain D and range R then: The hashing process is the process of mapping an input string into an output called. The output of the hash function is called the hash value or hash result [6].

$$h: D \rightarrow R \; and \mid D \mid > \mid R \mid$$

The main idea of hashes is that a hash value acts as a simple representation of data (also called message-diggest, imprint, digital fingerprint) from an input string, and can be used only if the hash value can be uniquely identified with the input string that. The h function is many-to-one, allowing for the same input pair with output: collision. In general, the hash function must have 2 (two) basic properties, such as:

1. Compression, the function h maps an input x with any length to output y = h (x) with a fixed length n;
2. Easy to calculate (Easy Computation), given h and an x input, y is easy to calculate.

The hash function is classified in two classes [7]:

1. MDCs (Manipulation Detection Codes).
2. Message Authentication Codes (MACs).

MDCs (Manipulation Detection Codes), or also referred to as codes detecting codes (MICs) which are keyless Hash functions [8]. The goal is (informal) to provide a representative image (called a hash value) in a message. In this class, the input of the hash function is only the message that will be sent and does not require a secret key input. MDCs are

widely used for data integrity services especially in digital-signature schema applications [9]. Class divisions of MDCs are:

1. One Way Hash Functions (OWHFs).
2. Collision resistant hash Functions (CRHFs).

The examples of hash algorithms for MDCs include MD2, MD4, MD5, RIPE-MD, Snefru, N-Hash, Secure-Hash-1 (SHA-1) which are OWHF while MASH-1 and MASH-2 entered the CRHF category. Message Authentication Codes (MACs) are hash functions for data authentication with symmetric techniques [10]. The MACs algorithm takes two inputs that are functionally different, a message and a secret key to produce a fixed size output. Practically it is difficult to produce the same MAC value without knowledge of the secret key. MACs also provide data integrity services. Examples of hash algorithms for MACs applications are HMAC, MD5-MAC, MAA, Block-based MACs: CBC-MAC, RIPE-MAC, GOST; Stream-based MACs: SEAL; CRC-based MAC, and others. In principle, MACs can be constructed from MDCs such as the MD5-MAC algorithm.

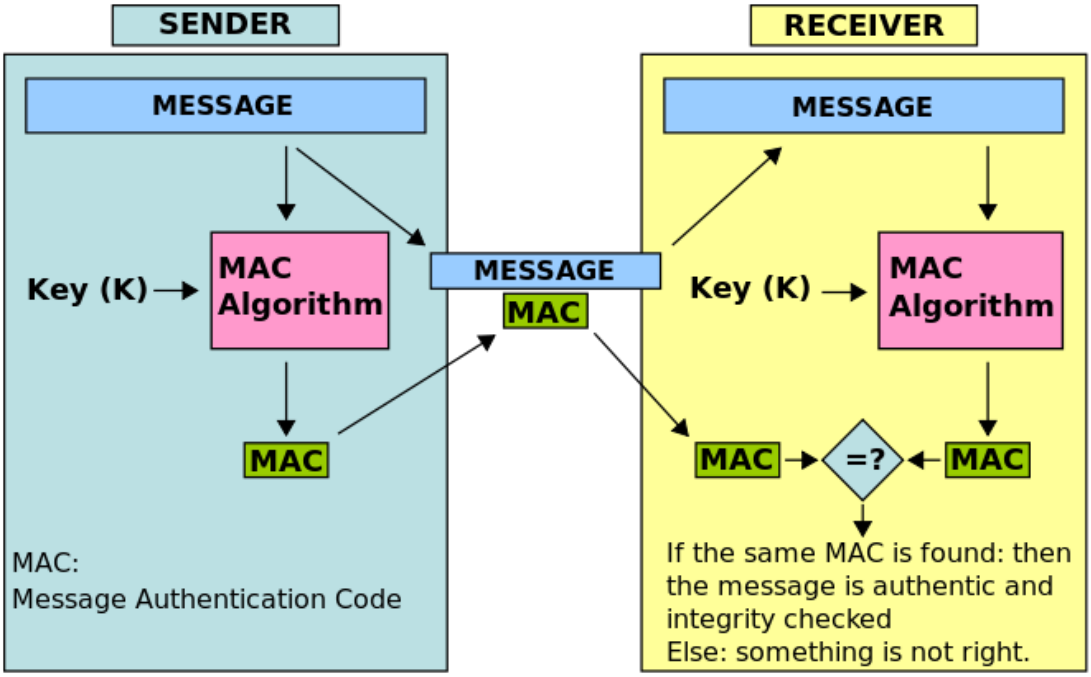

Figure 2. Message Authentication Code scheme

Figure 2 explains the MAC process. Wikipedia describes "*The sender runs it through a MAC algorithm to produce a MAC data tag. The message and the MAC tag are sent to a receiver. The receiver runs the message portion of the transmission through the same MAC algorithm using the same key, producing a second MAC data tag. The receiver then compares the first MAC tag received in the transmission to the second generated MAC tag. If they are identical, the receiver can safely assume that the message was not altered or tampered with during transmission*" [10].

2.3 MD5 Algorithm

The MD5 algorithm was developed by an MIT Professor named Ronald L. Rivest. Development of MD5 has been through 5 revisions, where first and second generation MDs are designed to help RSA algorithm in computing the signature of a secret message that will be sent and encrypted by RSA [11]. The third and fourth generation MDs are present because of competition from other hash algorithms called SNEFRU, which has a speed advantage in the computation process compared to MD2. When it was discovered that there was a security gap from SNEFRU in 1992, the same year found the weaknesses of MD4, which then Professor Rivest immediately patched the weakness and replaced it with the fifth generation Message digest; it is MD5. Of these five generations, the first and third generation MDs are unpublished algorithms. While the algorithm specifications MD2, MD4, and MD5 are found in RFC1319, RFC1320, and RFC1321 [12].The MD5 algorithm is an algorithm that uses a one-way hash function created by Ron Rivest. The algorithm is the development of the previous algorithms such as MD2 algorithm and the MD4 algorithm because cryptanalysts successfully attacked both of these algorithms. The cryptographic work of the MD5 algorithm is to receive input in the form of messages of any size and produce a diggest message that has a length of 128 bits. The following figure is the MD5 cryptographic algorithm process [13].

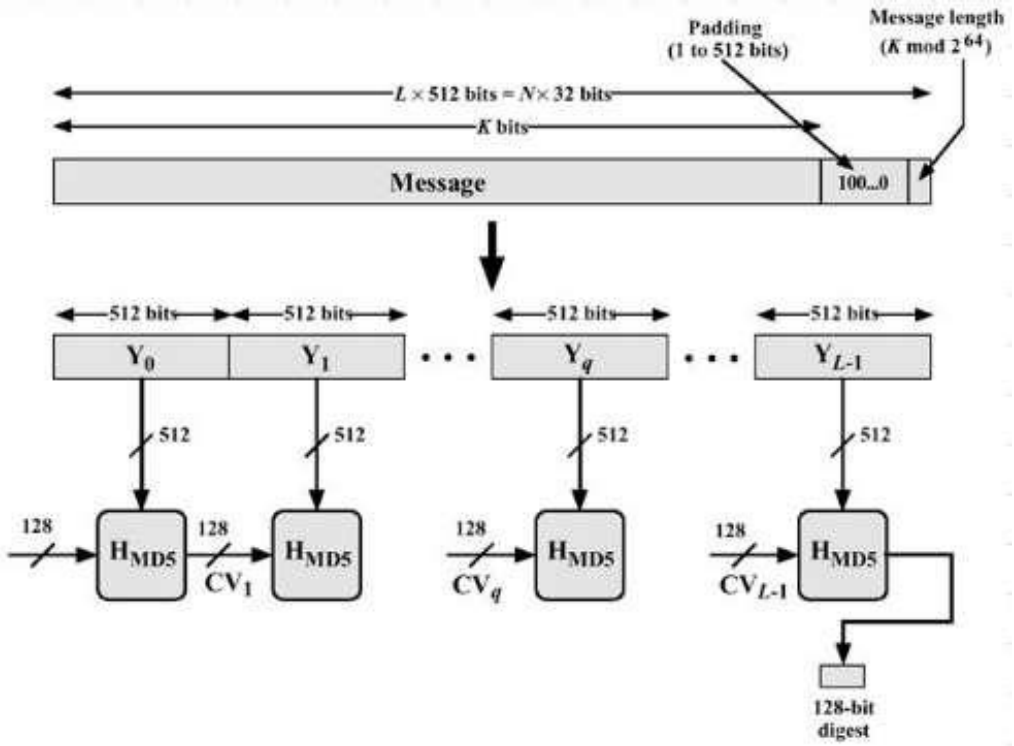

Figure 1. MD5 algorithm process

2.4 ElGamal algorithm

A scientist designed the ElGamal algorithm from Egypt named Taher Elgamal in 1984 based on the concept of the Diffie-Helman algorithm. This algorithm belongs to the category of public or asymmetric key cryptographic algorithms. This algorithm was originally used for digital signature processes, but then modified so that it can be used for encryption and decryption processes. The security of the ElGamal algorithm lies in the difficulty of calculating discrete logarithms in large modulo so that the attempt to solve the problem of logarithms becomes challenging to solve which is the advantage of this algorithm. Besides, this algorithm also has other advantages, such as that ordinary plain text will be encrypted into different text ciphers. However, even though the cipher text obtained varies, but the decryption process will get the same plain text. In the key formation process, it takes a prime number p, random numbers g and x, with the terms g <p and $1 \leq x \leq p-2$. The ElGamal algorithm's public key consists of pairs of 3 numbers (y, g, p) where:

$$y = g^x mod\ p$$

While the secret key is the number (x, p). The encryption process is done by calculating the values of a and b with the following equation.

$$a = g^k mod\ p$$
$$b = y^k m\ mod\ p$$

The "k" is a random number with the condition $1 \leq k \leq p-2$. The calculation results will get a ciphertext block from the "m" character in the block (a, b). The decryption process is carried out with the following equation.

$$m = b.c\ mod\ p$$

The "m" is plain text. The value of the variable c can be searched using the following equation.

$$c = a^{(p-1-x)}\ mod\ p$$

3. RESULT AND DISCUSSION:

This section explains the testing of MD5 and ElGamal algorithms in verifying and encrypting files. There are eight different types of files. Each file will experience MD5 and ElGamal testing. The test results of these files can be seen in the following table.

Table 1. File test results for both algorithms

No.	File	Key	Digital signature	Process Time
1	Data.sav 19 KB	*Private Key*: 41ad7,3c2e4,d2ba7 *Public Key*: 7b60,d2ba7	42663,6bf7b.7cf04,71bd0.6e58b,50bf8.62f3c,19cb2.5c329,12bc7.aca3e,1b4.3c876,2b079.6392d,7859b.32dda,36738.5ddd5,28736.c3cc9,c4122.93dd,9e628.810a,580ec.30390,8f763.6e9be,d1533.2877b,b4d76.	5,0351119041443
2	Data.xlsx 59 KB	*Private Key*: 14bb4,200a1,3576b *Public Key*: 352a9,3576b	25997,31032.23b69,2e878.121fe,2e8ef.f068,27b96.3160d,e7a3.350df,2a168.1e76c,8143.137aa,dece.14e79,84c4.2f185,a22.248e6,231ea.5e6d,254cb.d869,10921.13971,a89d.1f14c,3463a.673e,f78b.	5,6404099464417
3	Chapter I.pdf 151 KB	*Private Key*: 13997,b1043,ee6d9 *Public Key*: 2a26d,ee6d9	1f99d,a997f.178c8,153ad.9d85f,34001.d641d,7d6d0.b4f72,67851.b1b26,19512.764d2,227d1.dd64b,1f5a9.dda68,e0107.cbce1,54ad8.52aec,edf1.5f9fd,d5e2c.7607b,b3367.f7c7,14be1.818ff,59f86.15a23,6c370.	6,7597620487213
4	Chapter II.docx 428 KB	*Private Key*: 24337,2783b,49ced *Public Key*: 49a2e,49ced	3f631,29a6.2469a,101bb.1e9ce,21fad.1200d,41c8a.24665,dfa1.32200,201bd.11034,111da.36936,48bc1.14309,14560.1c86f,0.31069,47d9.45013,19e6d.217c4,46c5f.1f5ce,3b972.4263c,56e0.c45,3a368.	6,9104231834412
5	SMS.rar 3221 KB	*Private Key*: 23410,17b08,2538d *Public Key*: 9db0,2538d	23e10,1ceb8.1a732,1a66.e1fb,250f.16309,d935.238e4,35de.dd50,1b6b4.12202,d371.9ac7,dd2f.d919,1fca6.8435,5d06.86d6,8d6e.1c343,136.22f60,11ad9.1a763,9bcb.3c8e,b130.2164b,3b91.	7,2332480430603
6	Sing a Song.mp3 3939 KB	Private Key: 56f91,4163c,6536b Public Key: 1ae4f,6536b	2b8a4,287f6.1ebd9,608e.6286f,2c334.531ac,59f42.3daaf,47326.63ece,4c03d.21c73,27cd5.32a5f,1ebf3.486f0,49b3b.348ad,1133c.21970,238e2.3abf,2c058.495ce,2a19.3f6dd,2ae3a.5dcbd,48b0e.1f53f,1c6a8.	7,3264001369476
7	xampplite-win32-1.7.3.exe 29177 KB	*Private Key*: 5938,750b,1c5dd *Public Key*: 13f62,1c5dd	2c57,159c1.12ed3,148d2.15e35,bbb7.c61c,b64.4c8e,a33d.154dc,19145.6c2e,c8bf.1544b,2da3.12291,89d9.1b800,10d47.179e4,14dd1.ac03,1acb3.821e,138c8.11b9f,142fd.187da,da0c.393d,f445.	7,8895659446716
8	Dragon Ball Super – 82.mp4 99382 KB	Private Key: 6e4c3,306f1,baf4d Public Key: 6fff8,baf4d	7b3c9,19739.453fa,add07.12e4e,82c14.aad9a,9b94f.5a362,7433d.99b66,58ef.a7338,b0b1f.84c03,1882b.7dad7,98c4a.7c44,95bab.4237c,90b8e.559ce,b666.448e1,b28dc.6dffd,52b54.1213d,65c7.3202f,7a439.	8,5293769836426

The MD5 algorithm will provide the same message digest length for each file that is processed, which is 128 bits long or 32 hexadecimal characters. Therefore, there are 2^{128} or 3,402 x 10^{38} other possible files that have the same message digest. So, if the modifier (the person trying to modify the file) has a computer that can try 1 trillion (10^{12}) / second operation to find a pair that has the same message digest. Solution time: $2^{128}/10^{12}$ seconds or 10,790,283 trillion years to find the same data pair. The existence of the ElGamal algorithm, the modifier cannot arbitrarily modify the file and generate a digital signature. It is because digital signatures in this study are generated by encrypting message digest from files using private keys. For digital signature testing, a public key is needed, while a public key only has one private key pair, so the modifier cannot encrypt using any private key, it requires a private key with a partner that matches the public key so that the modification is unknown. To break the private key that is the value of y, g, and p, the time required depends on the length of the prime number for the key used. The longer the prime number used, the harder it is for the key to being solved. In the built system, prime numbers used only have six digits of prime numbers, such as from the range 100000 to 999999. These prime numbers are not too strong, but if the system uses a prime number of 25 characters, then there are 10^{25} possible pairs of y and g numbers can occur. So that if the modifier can experiment 1 trillion (10_{12}) /

second operation, then it will take $10^{25}/10^{12} = 10^{13}$ seconds or 31,7098 years. The total time needed to solve message digest and private key is 10,790,283 trillion x 31,7098 = 3,421,576,293,759 trillion years. Based on the security analysis that has been described, it can be concluded that the combination of MD5 and ElGamal algorithms will provide a high level of security and difficult to penetrate for the application of digital signatures.

4. CONCLUSION:

The digital signature of a file can be generated by first generating message digest from the file and then encrypting the message digest with the ElGamal algorithm using a private key. However, there is a note that in the private key digital signature and the public key in the ElGamal algorithm, the private key becomes the public key and the public key becomes the private key. File authenticity testing can be done by comparing message digest from files tested against decryption results from the included digital signatures. If the digest message is both different then the file has been modified, but if both message digests are the same then the file has not been modified, and authenticity is maintained. The digital signature generated from the combination of the MD5 algorithm and Elgamal algorithm has a very high level of security. Assuming that if the modifier has Experiment capability of 1 trillion operations per second, it takes 10,790,283 trillion years to solve the message digest from the file. The modifier must first solve the encryption of the required digital signature which if the prime number used is 25 digits then it takes 31,7098 years. So the total time needed to solve the digital signature is 3,421,576,293,759 trillion years.

REFERENCES:

1. Hariyanto dan A. P. U. Siahaan, “Intrusion Detection System in Network Forensic Analysis and,” *IOSR J. Comput. Eng.*, vol. 18, no. 6, hal. 115–121, 2016.
2. E. Dailami dan T. I. Rukmanto, “Gambaran Digital Signature,” *Belajar Logika*, 2011. [Daring]. Tersedia pada: https://belajarlogika.wordpress.com/2011/09/16/gambaran-digital-signature/.
3. A. P. U. Siahaan, “Rabin-Karp Elaboration in Comparing Pattern Based on Hash Data,” *Int. J. Secur. Its Appl.*, vol. 12, no. 2, hal. 59–66, Mar 2018.
4. 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. 2.27, hal. 17–21, 2018.
5. S. Ramadhani, Y. M. Saragih, R. Rahim, dan A. P. U. Siahaan, “Post-Genesis Digital Forensics Investigation,” *Int. J. Sci. Res. Sci. Technol.*, vol. 3, no. 6, hal. 164–166, 2017.
6. Zae, “Tutorial Pemrograman Kriptografi C++ dan Java,” *Belajar Kriptografi Sambil Ngopi*, 2015. [Daring]. Tersedia pada: http://ilmu-kriptografi.blogspot.com/2009/05/fungsi-hash.html.
7. M. Peyravian dan D. Coppersmith, “A structured symmetric-key block cipher,” *Comput. Secur.*, vol. 18, no. 2, hal. 134–147, Jan 1999.
8. R. Cramer, S. Fehr, dan C. Padró, “Algebraic manipulation detection codes,” *Sci. China Math.*, vol. 56, no. 7, hal. 1349–1358, Jul 2013.
9. K. Wang, Pei, L. Zou, A. Song, dan Z. He, “On the security of 3D Cat map based symmetric image encryption scheme,” *Phys. Lett. A*, vol. 343, no. 6, hal. 432–439, Agu 2005.
10. J. Katz dan A. Y. Lindell, “Aggregate Message Authentication Codes,” in *Topics in Cryptology – CT-RSA 2008*, Berlin, Heidelberg: Springer Berlin Heidelberg, hal. 155–169.
11. D. Rachmawati, J. T. Tarigan, dan A. B. C. Ginting, “A comparative study of Message Digest 5(MD5) and SHA256 algorithm,” *J. Phys. Conf. Ser.*, vol. 978, hal. 012116, Mar 2018.
12. Z. Wang dan L. Cao, “Implementation and Comparison of Two Hash Algorithms,” in *2013 International Conference on Computational and Information Sciences*, 2013, hal. 721–725.
13. A. P. U. Siahaan, “A Three-Layer Visual Hash Function Using Adler-32,” *Int. J. Comput. Sci. Softw. Eng.*, vol. 5, no. 7, hal. 142–147, 2016.