International Journal of Informatics and Information System ISSN 2579-7069
Vol. 4, No. 1, March 2021, pp. 01-12

Comparison of Security Signing Data Authentication Integrity in Combination of Digest And AES Message Algorithm

Rasna [1,*], Irjii Matdoan [2], Sitti Nur Alam [3]

[1,3] University of Yapis Papua, Indonesia
[2] University of Sains and Technology, Indonesia
[1] razna.irjii@gmail.com*; [2] irjiimatdoan12@gmail.com; [3] adzkadsar@gmail.com
* corresponding author

(Received January 8, 2021 Accepted February 19, 2021, Available online March 1, 2021)

Abstract

Online information systems with the Single Sign-On (SSO) model are currently widely used by many companies. Single Sign-On (SSO) is an independent authentication model. This system runs on the Hypertext Transfer Protocol (HTTP) protocol. Sending data or information without security is at risk of eavesdropping on information by the authorities. This study aims to compare the combination of Message-Digest and Advanced Encryption Standard (AES) algorithms to improve data security by modifying dynamic keys. The test results show that in each execution the user name and password with the MD5 algorithm are always the same while in the AES algorithm the results are always different so it is safe from replay attacks. So that the Advanced Encryption Standard (AES) algorithm can secure data through the Single-Sign-On authentication process with high-security accuracy.

Keywords: Message-digest, AES, SSO, Encryption, Security

1. Introduction

The rapid development of technology and information has a positive impact, namely the ease of sharing information or data through a computer network [1]. But at the same time, it also has a negative impact that is information or data can be accessed by parties who are not responsible for the crime. This also applies to the application of new technologies, the organization is faced with various opportunities and risks that can affect the performance of the organization[2]. One of the applications of new technology is that online information systems currently take an important role in companies. Online information systems are intended to improve service to each employee and improve company performance. Management of information systems becomes an important principle of the security of the system. This is because, the ease of accessing information, whether directly or indirectly, certainly has an impact on the emergence of risks and threats to the security and integrity of the data set. Threats that are expected to occur are unauthorized access to information or sources of information, such as duplication, alteration, or even destruction of information itself, thus bringing harm. For this reason, security management is needed that can protect or retain unauthorized access to maintain the data within a certain period time [3]. The system needs to implement security services such as authentication, encryption, access control, user management, and licensing [4]. Information security is one of the problems in ensuring data transmission over the web [5].

Information security is an effort to safeguard information and information systems from all possible threats to ensure and ensure business continuity, minimize business risk, and increase business opportunities. Information is an organizational asset that must be protected by security [6]. Data security has a major role in the development of a communication system, where more randomization in the secret keys increases the security as well as the complexity of the cryptography algorithms[7].

The login system is something that is found and is one of the important aspects of the world of the internet, which needs to be considered for its safety. The internet is built through a network of interconnected computers, with so many users connected to the network, data and information are very vulnerable from unauthorized outsiders [8]. This makes all data and information easily accessible to anyone. Not only can be accessed by people who are authorized or interested, but also by others who want to use it for personal gain by stealing data and information from the

computer system (hackers)[8]. For this reason, the system is demanded to be able to know the users or users who will use the system are those who have been given permission or who have an interest[9]. The user must identify himself to the system, and the system must ascertain whether the identification is authentic or not [8].

From the literature search results, there are currently several methods that can be used to improve the security of sending data using the HTTP protocol. These security methods include Hypertext Transfer Protocol Secure (HTTPS) [10], Secure Hypertext Protocol (SHTTP) [10], and Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) [11]. However, several other research results show there are still some weaknesses in using HTTPS [12]. Other literature that has improved data security by using MD5 [13][14][15], hardware to support MD5 speed and security level [16]. MD5 process by adding bits, adding message length values, initializing the Message Digest buffer, and processing messages in 512-bit blocks [17]. The data processing MD5 algorithm is quite fast because it only takes a data length of 128 bits [8].

In information systems that implement authentication using a password, each user logs in to the system by typing in a username and password, which is ideally known only by the system and the user concerned. The process of logging in is when the system is convinced that the user who is trying to access is entitled [18]. Web-based information systems usually store data about the username and password in a table in the database. Therefore, the system will check into the database whether the user name and password entered are correct or not [18].

From the background and problems raised above, a study was conducted to compare the use of the MD5 algorithm on Single Sign-On and the 128 bits Advanced Encryption Standard (AES) algorithm. The output of this research is expected considering the selection of algorithms in securing the integrity of data security, flexibility in various software and hardware.

2. Research Methods

2.1. Algorithm Message Digest

MD5 is developed from MD, MD2, MD3, and MD4 [16]. This algorithm utilizes a series of non-linear algorithms to perform circular operations, so the cracker cannot return the original data.

In cryptography, it is said that an algorithm such as an irrevocable algorithm can effectively prevent data leakage caused by reverse operations. Both theory and practice have because the use of the MD5 algorithm does not require payment of royalties, less time, and costs which makes it widely used in general non-confidential applications. Operates on a single block (of 512 bits for Whirlpool) but the communication remains unidirectional except for the last block of the

hash computation when the result is returned [19]. The processing logic is shown in Fig. 1

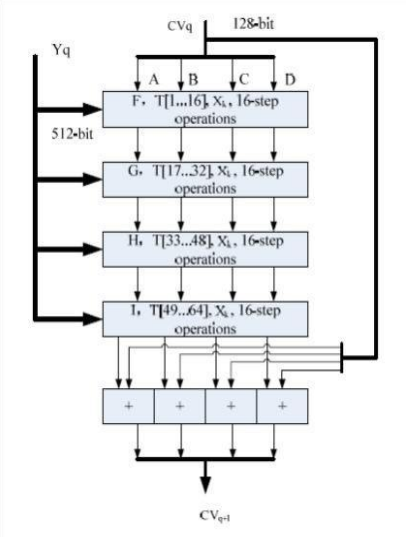

Fig 1. MD5 Processing Logic [16]

The 4-cycle process has the same structure but each has a different logic function. The functions used in each round are as follows:
F (x, y, z) = (x & y) | ((~ x) & z) (1)
G (x, y, z) = (x & z) | (y & (~ z)) (2)
H (x, y, z) = x ^ y ^ z (3)
I (x, y, z) = y ^ (x | (~ z)) (4)
The MD5 algorithm falls into two categories.
1. Hash operation on large data block and get hash value
2. Hash operation on many small data blocks and get the hash value of each small data block[16].

2.2. Algorithm Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) can store data and communications within an organization. AES uses the same key as the transmission isobilateral key as the receiver. AES uses 128, 192, and 256-bit cryptographic keys [20].

The AES algorithm has a complex internal process and structure that ensures it is very safe and has no weaknesses [21]. The Rijndael AES algorithm consists of variable block sizes which can also be 128, 192, or 256-bit. This Rijndael algorithm with key sizes of 128, 192, and 256-bit provides approx [21]. Rijndael can be used as an iterated hash function by using it as the round function. Here is one possible implementation. It is advised to use a block and key length both equal to 256 bits. The chaining variable goes into the "input" and the message block goes into the "Cipher Key". The new value of the chaining variable is given by the old value EXORed with the cipher output [22].

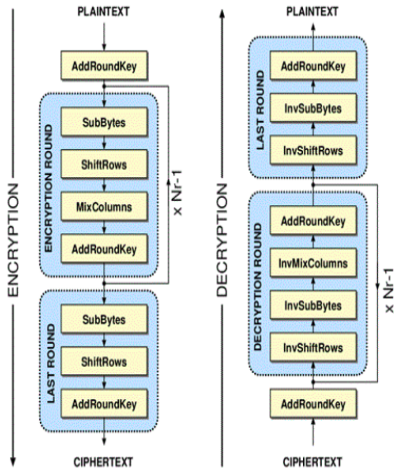

Fig 2. The overall structure of The AES algorithm
(Source: Http:// crypto.stackexchange.com/questions/8043/aes-addroundkey)

Some stages in AES are ByteSub, Shift Rows, MixColumn, and add Round key [20]. The first stage is SubBytes, which is the replacement of non-linear bytes, where each state byte is operated independently. H is reached via S-box. S-box is a substitute table that has been calculated previously with a value of 256 numbers (from 0 - 255) and the value of the matching results.

		x														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
	0	52	9	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	8	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	0	8c	bc	d3	0a	f7	e4	58	5	b8	b3	45	6
y	7	d0	2c	1e	8f	ca	3f	0f	2	c1	af	bd	3	1	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1a
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	7	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	4	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Fig 3. S-Box [20]

The second stage, Shift Rows: In the transformation phase shift Line 0 stays in position, when rows 1,2 and 3 shift one byte, two bytes, and three bytes respectively [23].

The fourth stage is the Round Key. Subkeys are generated from the primary key. Subkeys are the same size and inserted with concatenating each with related bytes in subkeys [24]. In the AES decryption process, the cipher transformation can be reversed and implemented in the opposite direction to produce an inverse cipher that is easily understood for the AES algorithm. The byte transforms used in inverse ciphers are InvShiftRows, InvSubBytes, InvMixColomns, and ArroundKey.

2.3. Research Tools and Materials

The hardware specifications used in the system implementation used in this study are as follows:

Table 1. Hardware Specifications

Hardware	**Spesifikasi**
Computer	• Processor Intel Core i7 2,5 GHz • Memory : 4 Gb • Microsoft Windows 64 Bit

While the software specifications used in implementing the system used in this study are as follows:

Table 2. Software Specifications

No	***Software***
1	Web Browser
2	Wamp Server
3	Notepad
4	Burp Suite
5	Wireshark
6	Visual Studio

3. Results and Discussion

3.1. Implementation of MD5 dan AES

The implementation of the MD5 algorithm is to accept input in the form of randomly generated messages and produce a message digest that has a length of 128 bits. AES is a symmetric block cipher where a single key is used for both encryption and decryption process. Te input and output for the AES algorithm each consist of sequences of 128 bits. The key used in this algorithm consists of 128, 192, or 256 bits. AES operates on 8-bit bytes [25].

AES is a symmetric blocks cipher with key sizes 128, 192, or 256 bits and blocks size of 128 bits. It has 10, 12, and 14 rounds which depend on the key size. The proposed design is based on the AES-128 Encryption. Each 128-data bits block along with the 128-bit cipher key are processed through a 4 x 4 state matrix and key matrix respectively. At the start of the algorithm, the state matrix is initialized with the original plain text while the key matrix is initialized with the user input key.

The following block diagram of AES Algorithm:

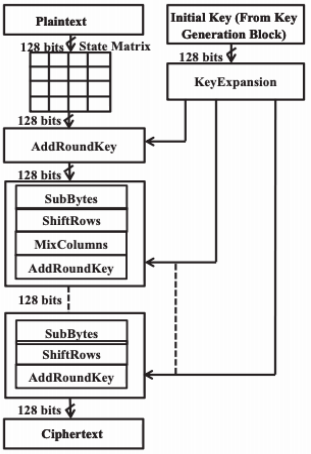

Fig 4. Implementation of AES Algorithm [26]

The input data block is XOR-ed with the first 128 bits of the passkey to generate the status (intermediate cipher result)[27]. For encryption, each round consists of the following four steps: 1) substitute bytes, 2) shift rows, 3) mix columns, and 4) add round key. For decryption, each round consists of the following four steps: 1) inverse shift rows, 2) inverse substitute bytes, 3) add round key, and 4) inverse mix columns. The last step consists of XOR-ing the output of the previous three steps with four words from the key schedule; the outcome of the last round is either the encrypted or decrypted block. In the encryption and decryption process of AES, the State array is modified at each round by a round function that defines four different byte-oriented transformations [28]. Each round of the encryption process requires a series of steps to alter the state of the array [29]. Implementation of AES Algorithm is as follows :

a. All the 16 byte input messages are arranged in a four-of-four byte matrix called state matrix.

b. Key bytes are arranged into a matrix with four rows and 4, 6 or 8 columns as the length of key of 128,192 or 256 bit respectively.

c. Byte Substitution Layer: this is the first layer of each round for encryption of text.

d. Shift Row Transformation Layer: In this layer cyclically shift the second row by one position left, third row by two position left and forth row by three position left and first row has no change. A Shift Row is arranging elements of a state key matrix which performs a circular shift each row. The circular shift length is different for each row. The first row is never moved over. Second row moves one first element to the right at the last element. Third row moves two first elements to the right at the last element and the last row moves three first elements [30].

e. Mix Column Transformation: This is a linear transformation which mixes each column of the state matrix obtained after shift row transformation and all the arithmetic involving the coefficients is done in Galois Field (GF(28)). Key Generation Using GA-The process of generating the key from the Genetic [31]. The MixColumn step is a linear transformation which mixes each column of the state matrix. Each byte is replaced by a value dependent on all 4 bytes in the column and is performed by the following multiplication [32]. The decryption process will be the opposite of the encryption round where the different functions will be using their inverses: ShiftRow to Inverse ShiftRow, SubBytes to Inverse SubBytes, and MixColumns to Inverse MixColumns [33].

f. Key Addition Layer: In this layer the state byte matrix obtained after the mix column transformation layer are XORed with the sub keys of the previous round. Which also consist of 16 bytes [28]. A simple bitwise XOR operation between each byte in the state matrix with its corresponding byte in the key matrix. The key matrix corresponds to the same round of the state matrix [34]. The final round consists of only three transformations ignoring MixColumns [35]. The Decryption method is the reverse of encryption and it consists of four transformations [36]:

   1. *Inverse Substitution Bytes*
      The 16 byte plain-text substitutes the corresponding value from substitution table S-box [37].

Table 3. Inverse S-Box [35]

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Inverse Substitution Bytes is the inverse of the substitution byte transformation. This is performed through inverse S-box [38] [39].

2. *Inverse Shift Rows*

In shiftrows transformation, the bytes in last 3 rows will be shifted cyclically over number of bytes present.

a) The first row will remain same.
b) The second row will get shifted to the left by one position.
c) The third row will get shifted to the left by two positions.
d) The fourth row will be shifted to the left by three positions.

3. *Inverse Mix Columns*

MixColumns transformation performs by transforming each column of four bytes. It takes input as one column which is of 4 bytes and output as completely different 4 bytes by transforming the original column. The resultant matrix is same as the size of plain-text. MixColumn transformation will not be carried in the last round.

4. *Add Round Key*

The 16 bytes which is produced from MixColumns is equal to 128 bits which is XORed with the round key of 128 bits. The above process has been repeated until final round to produce the corresponding cipher text [40].

Different from AES, MD5 many situations where a potentially long message needs to be processed and/or compared quickly [41]. 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 [42]. MD5 consists of 64 operations, grouped into four rounds of 16 operations [43].

The main MD5 process consist of five steps that is used to convert plain text into cipher text which are as follows:

Step 1: Append padding bits The message is padded so that its length is congruent to 448, modulo 512. The message is extended so that it is just 64-bit shy of being a multiple of 512 bits long. So, “1” bit is appended to the message and then 0 is appended so that the length is congruent to 448.

Step 2: Append length A 64-bit representation of length of message before padding bits were added is appended to the result of the previous step.

Step 3: Initialize MD buffer A four-word buffer is used to compute the message digest where each of the 32-bit register is initialized in hexadecimal, low-order bytes.

Step 4: Process message in 16-word bits The four auxiliary function is then processed with various steps to produce the desired output.

Step 5: Output The message digest is produced as an output. The plain text is converted into cipher text or hashed form [44].

Following is the basic implementation of the MD5 algorithm:

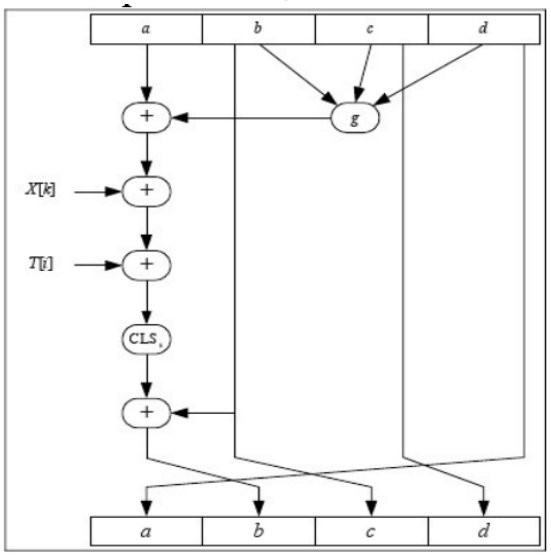

Figure 5. Implementation of AES Algorithm

The basic operations of MD5 shown in the picture above can be written in the following equation:

$$a \leftarrow b + CLs(a + g(b, c, d) + X[k] + T[i]$$

Where :
a, b, c, d = four 32-bit buffer variables (containing buffer values A, B, C, D)
g = one of the functions F, G, H, I
CLSs = circular left shift of s bits
X[k] = the 32nd k-32-bit group of the 512th q-message block. Value k = 0 to 15.
T[i] = Table element (32 bits)
+ = modulo addition operation 232
The steps in implementing the MD5 method are done by using 2 PHP files that function as input form files and database connections. Initial value of the hash function is replaced by a non-standard value, which is the result of the attack [45]. The database used for this implementation is MySQL, arguing that MD5 cryptography is an integrated part of MySQL.

By using syntax:

$encripted_password=md5 {$password}

Then the password on the database will be encrypted by itself MD5 in a similar way, and the result hash value is compared with the hash value in the database for that particular user [46].

The SSO authentication in this study uses the MD5 algorithm to validate login user accounts. Users can connect with SSO through validating user login sessions. The process goes through stages on the NuSOAP web service by checking the value on id_user.

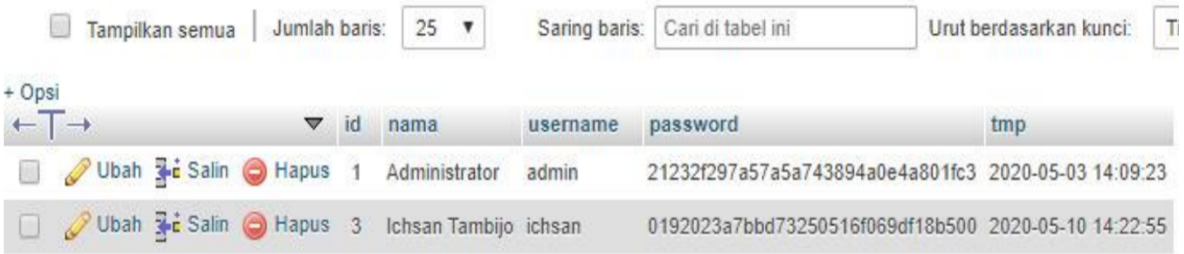

Fig 6. Display encrypted password on the database.

3.2. Testing

Testing in this study was conducted on hash variables and user login accounts. The test was carried out to obtain the results of the plaintext from the user's login encryption on SSO and MD5 from the table. Stages of testing consist of two, namely testing the sample username and password stored in two separate files.

Matches on the username and password in the results of this test were found by looking for the match string and Http response variables of each username and password. Testing on the AES encryption side with a dynamic key generator with the username and password received in the form of ciphertext. The process with the application of a dynamic key generator means that the AES key used will always change for each decryption encryption process based on changes in time value. AES is secure against the brute-force attack[47]. In the AES encryption process using four basic transformations with sub bytes, shift rows, mix columns, and add round key sequences. Whereas the decryption process uses the inverse of all basic transformations in the AES algorithm except add round key in the order of the transformation of in shift rows, in sub bytes, add round key, and in mix columns. In-text data, the encryption process begins by converting text into ASCII code in hexadecimal numbers formed into 4x4 byte matrices. Then some basic information is performed, such as sub bytes, shift rows, mix columns, and add round key. However, when carrying out data transformation, the data that is processed in each form is binary data from the hexadecimal matrix. AES 128-bit cryptography has a keyspace of 2128 which is a very large value and is considered safe to use avoid brute force attacks[48]. The following is a comparison test table for encryption with MD5 and AES Table 4 :

Table 4. Comparison test table for encryption with MD5 and AES

No	User Name	Password	MD5 (Username)	MD5 (Password)
1	vicario	vicario	3123cc9d0ae528ab0040cbb57ceaf225	3123cc9d0ae528ab0040cbb57ceaf225
2	Abdul	Abdul	ef095ba5e89f362a19c0ba2dd921ba10	ef095ba5e89f362a19c0ba2dd921ba10
3	Ichsan	Ichsan	0192023a7bbd732505161069df18b500	0192023a7bbd732505161069df18b500
4	Ichsan	Ichsan	0192023a7bbd732505161069df18b500	0192023a7bbd732505161069df18b500
5	Abdul	Abdul	ef095ba5e89f362a19c0ba2dd921ba10	ef095ba5e89f362a19c0ba2dd921ba10
6	user	user	ee11cbb1905e40b07aa0ca060c23eea1	ee11cbb1905e40b07aa0ca060c23eea1
7	vicario	vicario	3123cc9d0ae528ab0040cbb57ceaf225	3123cc9d0ae528ab0040cbb57ceaf225
8	User	User	ee11cbb1905e40b07aa0ca060c23eea1	ee11cbb1905e40b07aa0ca060c23eea1
9	Abdul	Abdul	ef095ba5e89f362a19c0ba2dd921ba10	ef095ba5e89f362a19c0ba2dd921ba10
10	Ichsan	Ichsan	0192023a7bbd732505161069df18b500	0192023a7bbd732505161069df18b500
11	User	User	ee11cbb1905e40b07aa0ca060c23eea1	ee11cbb1905e40b07aa0ca060c23eea1
12	vicario	vicario	3123cc9d0ae528ab0040cbb57ceaf225	3123cc9d0ae528ab0040cbb57ceaf225
13	User	User	ee11cbb1905e40b07aa0ca060c23eea1	ee11cbb1905e40b07aa0ca060c23eea1
14	Abdul	Abdul	ef095ba5e89f362a19c0ba2dd921ba10	ef095ba5e89f362a19c0ba2dd921ba10
15	Ichsan	Ichsan	0192023a7bbd732505161069df18b500	0192023a7bbd732505161069df18b500
16	vicari	Vicari	3123cc9d0ae528ab0040cbb57ceaf225	3123cc9d0ae528ab0040cbb57ceaf225
17	user	User	ee11cbb1905e40b07aa0ca060c23eea1	ee11cbb1905e40b07aa0ca060c23eea1
18	Abdul	Abdul	ef095ba5e89f362a19c0ba2dd921ba10	ef095ba5e89f362a19c0ba2dd921ba10
19	Ichsan	Ichsan	0192023a7bbd732505161069df18b500	0192023a7bbd732505161069df18b500
20	user	User	ee11cbb1905e40b07aa0ca060c23eea1	ee11cbb1905e40b07aa0ca060c23eea1
No	**User Name**	**Password**	**AES (Username)**	**AES (Password)**
1	vicario	vicario	9c006023e5007aa0cae112dde322e440	0b07060aa023e5161a2df1515019e4b0
2	Abdul	Abdul	cc9d0921ba0133e23f9011bcd821b10a	02037bdbee1223eeaa0ccbb1060cf334
3	Ichsan	Ichsan	ec9162a1aa0c2a19ba2dceafb500a2300	ee3109739dd4e723408761980ba5ef25
4	Ichsan	Ichsan	a012098e7c23e732550161069df18b00	ee320ae1060ccb0905dce690915e4f20
5	Abdul	Abdul	ba0c921dba10362a19ee292ee1211b00	dce289f210ab8b0011fdca0dd999ba17
6	user	user	0e017aa3ae25ceaf57cee2e382ba00aa	Adbc00110b07ceaea0ca11ccb40b000d
7	vicario	vicario	ae25cc9dee2e28ab0040cca06cea5ba5	ba5ecc095be528ab004057ce7c893200
8	User	User	98e7a0c9905e40b073e73a060cdf18a1	0ca11210a905e40b07aa0fdca0cdca00
9	Abdul	Abdul	aa3a5ba5e89f362a19c0b3e73921ba10	0dca5ba5e89f362a1900ba2dd921ba10
10	Ichsan	Ichsan	0192e2e37bbd732505161069df18b500	0192023a710a932505163921df186cea
11	User	User	0cdfcbb1905e40b0721baa060ce73a05	0c99cba3a55e40b07aa0ca03e733ee01
12	vicario	vicario	bb19cc9e40b528a a0600cbb57ceaf225	af22cc9d0ae528ab0040cbb57cea1cbb
13	User	User	9e40cbb1905e40b07aa0ca060c23ee0a	0b52cbb1905e40b07aa0ca060c23ee
14	Abdul	Abdul	2e375b57ce7f362a1910fa060921ca06	e3755b7ce79f362a19c07cea0610fa00
15	Ichsan	Ichsan	dfcb02b57cbd73250016121caf1860c2	b302023a7bb5e40b 05161069df610f 00
16	vicario	vicario	a0cacc9d0ae528ab004023a57ceaa7bb	095bcc9d905e28ab0023ee b57cea1cbb
17	user	User	0c23cb023a5e40b07ca0ca060ca060a1	d732cbb1905e40b07aa0ca03a7b30000

18	Abdul	Abdul	ba2d5ba5e89f362a19c0b00a06021aa70	a03a5aa5e89f362a19f0aa2dd92a7b30
19	Ichsan	Ichsan	0a06023a7bbd73350362a0a06018021a	019202dd9bbd732505a2dd69de1825a0
20	user	User	cc9dcbb1905e40b14ca0aa060c0a0670	9bbdc3a7b051e010b00ca0a2dd0c11ff

The test results above show that in each execution the user name and password with the MD5 algorithm are always the same, while in the AES algorithm the results are always different so it is safe from replay attacks. The dynamic key is generated on the AES algorithm using a function of time. Key can be generated at random based on the value of the time when the sender logs in to the system [49]. Cryptographic hash functions like MD5 do not have a sound mathematical security definition, but instead rely on the following "intuitive" notions of security: for a hash function h with domain D and range R, we require the following three properties [50].

AES has a varying number of rounds depending on key size [51]. The results of message encryption on both MD5 and AES algorithms increase in messages. For AES, the timing of memory accesses to look-up tables is strongly correlated with secret key data. Several implementation recommendations seek to reduce or eliminate this correlation. If possible, the embedder should avoid look-up tables altogether and use the logical implementations of AES instead. Alternatively, lookup tables can be stored in registers to eliminate memory accesses and associated timing. AES implementations using a smaller set or multiple copies of tables are also available which changes the access statistics, making timing more difficult to predict [52]. implementing AES in a way that is impervious to this attack, let alone developing an efficient generic countermeasure, appears non-trivial [53]. Based on the text files used and the experimental result it was concluded that the AES algorithm consumes the least encryption and RSA consumes the encryption time [54]. AES, Advanced Encryption Standard, is a symmetric key encryption standard which is widely used to secure data where data confidentiality is an important and critical issue. Symmetric key (AES) has high efficiency that it is suitable for encrypting a relatively long plaintext [7].

4. Conclusion

This study uses a comparison of the combination of Message-Digest and Advanced Encryption Standard (AES) algorithms to improve data security by modifying dynamic keys. The test results show that in each execution the user name and password with the MD5 algorithm are always the same while in the AES algorithm the results are always different so it is safe from replay attacks. So that the Advanced Encryption Standard (AES) algorithm can secure data through the Single-Sign-On authentication process with high-security accuracy. AES has different advantages such as security, flexibility, and ease of implementation [33]. Advanced Encryption Standard (AES) algorithm has become the optimum choice for various security services in numerous applications [55]. MD5 are commonly used for encrypting plaintext passwords into strings that theoretically cannot be deciphered by hackers due to their one-way encryption feature. However, with time, attacks became possible through the use of dictionary tables and rainbow tables [46].

References

[1] S. Sulastri and R. D. M. Putri, "Implementasi Enkripsi Data Secure Hash Algorithm (SHA-256) dan Message Digest Algorithm (MD5) pada Proses Pengamanan Kata Sandi Sistem Penjadwalan Karyawan," *J. Tek. Elektro*, vol. 10, no. 2, pp. 70–74, 2018.

[2] W. S. Prabowo, . W., N. A Setiawan, M. H. Muslim, and Y. S. Utama, "Manajemen Risiko Infrastruktur Cloud Pemerintah Menggunakan Nist Framework Studi Kasus Lembaga Ilmu Pengetahuan Indonesia (LIPI)," *J. Penelit. Pos dan Inform.*, vol. 7, no. 1, p. 17, 2017.

[3] Sumarno, I. Gunawan, H. S. Tambunan, and E. Irawan, "Analisis Kinerja Kombinasi Algoritma Message-Digest Algortihm 5 (Md5), Rivest Shamir Adleman (Rsa) Dan Rivest Cipher 4 (Rc4) Pada Keamanan E-Dokumen," *JUSIKOM PRIMA (Junal Sist. Inf. Ilmu Komput. Prima)*, vol. 2, no. 1, pp. 41–48, 2018.

[4] N. Rjaibi, L. B. A. Rabai, and A. Mili, "The MFC cybersecurity model extension and diagnostic toward a depth measurement: E-learning systems case study," *Achiev. Enterp. Agil. through Innov. Softw. Dev.*, no. January, pp. 179–198, 2015.

[5] S. Tayal, N. Gupta, P. Gupta, D. Goyal, and M. Goyal, "A Review paper on Network Security and Cryptography," *Adv. Comput. Sci. Technol.*, vol. 10, no. 5, pp. 763–770, 2017.

[6] R. Fauzi, "Implementasi Awal Sistem Manajemen Keamanan Informasi pada UKM Menggunakan Kontrol ISO/IEC 27002," *JTERA (Jurnal Teknol. Rekayasa)*, vol. 3, no. 2, p. 145, 2018.

[7] N. Mathur and R. Bansode, "AES Based Text Encryption Using 12 Rounds with Dynamic Key Selection," *Procedia Comput. Sci.*, vol. 79, pp. 1036–1043, 2016.

[8] M. S. Ramadhan and F. Ariyani, "Peningkatan Keamanan Login Website Dengan Implementasi One Time Password Menggunakan Algoritma Sha1 Dan Md5 Berbasis Mobile," *Skanika*, vol. 1, no. 2, pp. 689–696, 2018.

[9] S. Nagaraj, G. S. V. P. Raju, and V. Srinadth, "Data encryption and authetication using public key approach," *Procedia Comput. Sci.*, vol. 48, no. C, pp. 126–132, 2015.

[10] A. S. E. Rescola, "The Secure Hypertext Transfer Protocol. The Internet Engineering Task Force (IETF), California, USA.," no. 3, pp. 1–8, 1999.

[11] L. Fortunati, A. M. Manganelli, F. Cavallo, and F. Honsell, "You need to show that you are not a robot," *New Media Soc.*, vol. 21, no. 8, pp. 1859–1876, 2019.

[12] Z. Musliyana, M. Dwipayana, A. Helinda, and Z. Maizi, "Improvement of Data Exchange Security on HTTP using Client-side Encryption," *J. Phys. Conf. Ser.*, vol. 1019, no. 1, 2018.

[13] M. D. A. Chawdhury and A. H. M. A. Habib, "Security enhancement of MD5 hashed passwords by using the unused bits of TCP header," *Proc. 11th Int. Conf. Comput. Inf. Technol. ICCIT 2008*, vol. 5, no. Iccit, pp. 714–717, 2008.

[14] C. Lu and G. M. Xu, "Research and implementation of file encryption and decryption," *Adv. Intell. Soft Comput.*, vol. 141 AISC, pp. 165–170, 2012.

[15] F. A. Sagar, "Cryptographic Hashing Functions - MD5," no. September, pp. 1–9, 2016.

[16] P. Walia and V. Thapar, "Implementation of new modified MD5-512 bit algorithm for cryptography," *Int. J. Innov. Res. Adv. Eng.*, vol. 1, no. 6, pp. 2349–2163, 2014.

[17] T. Prasetyo and A. Hikmawan, "Analisis Perbandingan Dan Implementasi Sistem Keamanan Data Menggunakan Metode Enkripsi RC4 SHA Dan MD5," *Infotech J.*, vol. 2, no. 1, p. 236705, 2016.

[18] U. Rahardja and S. Valent, "Global Password Untuk Kemudahan," *Seminar*, vol. 2010, no. Snati, 2010.

[19] M. Wolf and T. Gendrullis, "Design, Implementation, and evaluation of a vehicular hardware security module," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7259 LNCS, pp. 302–318, 2012.

[20] I. K. Nti, E. Gymfi, and O. Nyarko, "Implementation of Advanced Encryption Standard Algorithm with Key Length of 256 Bits for Preventing Data Loss in an Organization," *Int. J. Adv. Technol.*, vol. 08, no. 02, pp. 1–5, 2017.

[21] A. Philip, "Information Security Reading Room The Legal System and Ethics _____ tu ho ll r igh," 2020.

[22] V. R. Joan Daemen, *AES Proposal : Rijndael*. 2018.

[23] H. Trang and N. Van Loi, "An efficient FPGA implementation of the advanced encryption standard algorithm," *2012 IEEE RIVF Int. Conf. Comput. Commun. Technol. Res. Innov. Vis. Futur. RIVF 2012*, vol. 2, no. 07, pp. 413–417, 2012.

[24] R. S. K. V Viswanadha, "An efficient FPGA implementation of the AES Algorithm With Reduced Latency," vol. 1, no. 10, 2013.

[25] R. Saha, G. Geetha, G. Kumar, and T. H. Kim, "RK-AES: An Improved Version of AES Using a New Key Generation Process with Random Keys," *Secur. Commun. Networks*, vol. 2018, 2018.

[26] H. Zodpe and A. Sapkal, "An efficient AES implementation using FPGA with enhanced security features," *J. King Saud Univ. - Eng. Sci.*, vol. 32, no. 2, pp. 115–122, 2020.

[27] A. Uskov, A. Byerly, and C. Heinemann, "Advanced encryption standard analysis with multimedia data on Intel® AES-NI architecture," *Int. J. Comput. Sci. Appl.*, vol. 13, no. 2, pp. 89–105, 2016.

[28] A. Kumar and R. R. Tewari, "Expansion of Round Key Generations in Advanced Encryption Standard for Secure Communication," *Int. J. Comput. Intell. Res.*, vol. 13, no. 7, pp. 1679–1698, 2017.

[29] B. Padmavathi and S. R. Kumari, "A Survey on Performance Analysis of DES, AES and RSA Algorithm along with LSB Substitution Technique," *Int. J. Sci. Res.*, vol. 2, no. 4, pp. 170–174, 2013.

[30] R. Riyaldhi, Rojali, and A. Kurniawan, "Improvement of Advanced Encryption Standard Algorithm with Shift Row and S.Box Modification Mapping in Mix Column," *Procedia Comput. Sci.*, vol. 116, pp. 401–407, 2017.

[31] A. Pandey and U. kumar Lilhore, "An Improved AES Cryptosystem Based Genetic Method on S-Box, With, 256 Key Sizes and 14-Rounds," *Int. J. Adv. Eng. Res. Sci.*, vol. 4, no. 3, pp. 166–171, 2017.

[32] A. S. Nampalliwar, "Implementation of AES Algorithm," *IOSR J. Eng.*, vol. 4, no. 6, pp. 01–05, 2014.

[33] E. M. De Los Reyes, A. M. Sison, and R. P. Medina, "Modified AES cipher round and key schedule," *Indones. J. Electr. Eng. Informatics*, vol. 7, no. 1, pp. 28–35, 2019.

[34] S. M. Soliman, B. Magdy, and M. A. Abd El Ghany, "Efficient implementation of the AES algorithm for security applications," *Int. Syst. Chip Conf.*, vol. 0, pp. 206–210, 2016.

[35] K. Renuka Devi, N. Suba Rani, and A. Noble Mary Juliet, "An Image Encryption and Decryption And Comparison With Text - AES Algorithm," *Int. J. Sci. Technol. Res.*, vol. 8, no. 7, pp. 668–673, 2019.

[36] G. Chaitanaya, B. Keerthi, A. Saleem, A. T. Rao, and K. T. P. S. Kumar, "An Image Encryption and Decryption using Chaos Algorithm," *IOSR J. Electron. Commun. Eng. Ver. II*, vol. 10, no. 2, pp. 2278–2834, 2015.

[37] S. Z. M. Naziri and N. Idris, "The memory-less method of generating multiplicative inverse values for S-box in AES algorithm," *2008 Int. Conf. Electron. Des. ICED 2008*, 2008.

[38] G. L. Guo, Q. Qian, and R. Zhang, "Different implementations of AES cryptographic algorithm," *Proc. - 2015 IEEE 17th Int. Conf. High Perform. Comput. Commun. 2015 IEEE 7th Int. Symp. Cybersp. Saf. Secur. 2015 IEEE 12th Int. Conf. Embed. Softw. Syst. H*, pp. 1848–1853, 2015.

[39] A. Msolli, A. Helali, and H. Maaref, "Image encryption with the AES algorithm in wireless sensor network," *2nd Int. Conf. Adv. Technol. Signal Image Process. ATSIP 2016*, pp. 41–45, 2016.

[40] L. M. Dinca and G. Hancke, "User-centric key entropy Study of biometric key derivation subject to spoofing attacks," *Entropy*, vol. 19, no. 2, 2017.

[41] E. Sediyono, K. I. Santoso, and Suhartono, "Secure login by using One-time Password authentication based on MD5 Hash encrypted SMS," *Proc. 2013 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI 2013*, pp. 1604–1608, 2013.

[42] S. Mishra, S. Mishra, and N. Kumar, "Hashing Algorithm : MD5," vol. 1, no. 9, pp. 931–933, 2013.

[43] C. G. Thomas and R. T. Jose, "A Comparative Study on Different Hashing Algorithms," *Int. J. Innov. Res. Comput. Commun. Eng. (An ISO Certif. Organ.*, vol. 3297, no. 7, pp. 170–175, 2015.

[44] P. Gupta and S. Kumar, "A Comparative Analysis of SHA and MD5 Algorithm," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 3, pp. 4492–4495, 2014.

[45] X. Wang and H. Yu, "How to break MD5 and other hash functions," *Lect. Notes Comput. Sci.*, vol. 3494, pp. 19–35, 2005.

[46] M. C. Ah Kioon, Z. S. Wang, and S. Deb Das, "Security analysis of MD5 algorithm in password storage," *Appl. Mech. Mater.*, vol. 347–350, pp. 2706–2711, 2013.

[47] Y. Zhang, X. Li, and W. Hou, "A fast image encryption scheme based on AES," *2017 2nd Int. Conf. Image, Vis. Comput. ICIVC 2017*, vol. 256, no. 104 Xi, pp. 624–628, 2017.

[48] A. R. Tulloh, Y. Permanasari, and E. Harahap, "Kriptografi Advanced Encryption Standard (AES) Untuk Penyandian File Dokumen," *J. Mat. UNISBA*, vol. 2, no. 1, pp. 118–125, 2016.

[49] F. J. D'souza and D. Panchal, "Advanced encryption standard (AES) security enhancement using hybrid approach," *Proceeding - IEEE Int. Conf. Comput. Commun. Autom. ICCCA 2017*, vol. 2017-Janua, pp. 647–652, 2017.

[50] J. Black, M. Cochran, and T. Highland, "A study of the MD5 attacks: Insights and improvements," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4047 LNCS, pp. 262–277, 2006.

[51] Z. Hercigonja, "Comparative Analysis of Cryptographic Algorithms," *Int. J. Digit. Technol. Econ.*, vol. 1, no. 2, pp. 127–134, 2016.

[52] A. Kaminsky, M. Kurdziel, and S. Radziszowski, "An overview of cryptanalysis research for the advanced encryption standard," *Proc. - IEEE Mil. Commun. Conf. MILCOM*, pp. 1310–1316, 2010.

[53] E. Tromer, D. A. Osvik, and A. Shamir, "Efficient cache attacks on AES, and countermeasures," *J. Cryptol.*, vol. 23, no. 1, pp. 37–71, 2010.

[54] C. Andersson and P. Runeson, "A replicated quantitative analysis of fault distributions in complex software systems," *IEEE Trans. Softw. Eng.*, vol. 33, no. 5, pp. 273–286, 2007.

[55] A. Fathy, I. F. Tarrad, H. F. A. Hamed, and A. I. Awad, "Advanced Encryption Standard Algorithm: Issues and Implementation Aspects," *Commun. Comput. Inf. Sci.*, vol. 322, pp. 516–523, 2012.