E-Surveillance System Security Using RSA-AES Algorithm (Rivest Shamir Adleman - Advanced Encryption Standard)

Hajra Rasmita Ngemba[a,1,*] , Aimar Anand [b,2] , Syaiful Hendra [c,3] , Anita Ahmad Kasim [c,4] , Elyana Aulia Chandra [b,5]

[a] Information Systems Study Program, Faculty of Engineering, Tadulako University, Jl. Soekarno Hatta No.KM. 9, Palu 94118, Indonesia
[b] Study Program S1 Informatics Engineering Faculty of Engineering, Tadulako University, Jl. Soekarno Hatta No.KM. 9, Palu 94118, Indonesia
[c] Informatics Engineering Study Program, University of Tadulako Faculty of Engineering, Jl. Soekarno Hatta No.KM. 9, Palu 94118, Indonesia
[1] hajra.rasmita@gmail.com*;[2] aimaranand02@gmail.com;[3] syaiful.hendra.garuda@gmail.com; [4] nita.kasim@gmail.com; [5] elyanaauliachandraa@gmail.com
* hajra.rasmita@gmail.com

ARTICLE INFO

Article history
Received
Revised
Accepted

Keywords
Encryption
RSA
AES
Decrypt
Cryptography

ABSTRACT

Introduction: The Regional Inspectorate of Central Sulawesi Province has the task of supervising and fostering the implementation of government such as conducting audits, evaluations, reviews, and monitoring. The Regional Inspectorate of Central Sulawesi Province built a system for fostering Regional Apparatus Organizations called the E- E-Monitoring system. However, the files uploaded by the Regional Device Organization have not been encrypted in this system, so this study aims to use the RSA-AES algorithm in the E-Pengawasan system to increase system security.

Method: The process that occurs in the RSA-AES algorithm is that the RSA key is used to encrypt the AES key and the AES key is used to encrypt and decrypt the file.

Results and Discussion: The test results used are the black box testing method and also usability testing. Black box testing is used to test whether the functions on the system are running smoothly or not and usability testing is used to test efficiency in encryption and decryption based on file size.

Conclusion: Based on the tests carried out, it was found that the amount of time used to encrypt files with a file size of 1MB - 50MB was 0.182 - 0.393 seconds. The test results for decryption were found with a file size of 1MB - 50MB, namely 0.050 - 0.148 seconds.

1. Introduction

APIP (Government Internal Supervisory Apparatus) is an institution in charge of overseeing government administration. This institution was formed to ensure that government administration runs with the expected goals [1].

The Provincial Inspectorate has the task of assisting the Governor in fostering and supervising the implementation of government which is the authority of the Region and tasks by Regional Apparatus. The Regional Inspectorate has duties and functions of authority such as the formulation of technical policies in the field of supervision and supervision facilities, and internal performance supervision through audits, evaluations, reviews, and monitoring [2].

The Regional Inspectorate of Central Sulawesi Province built a Regional Apparatus Organization (OPD) coaching system with the name E-Supervision system. So that Regional Apparatus Organizations can conduct administrative and financial checks on the Regional Inspectorate of Central

Sulawesi Province by uploading data from each related OPD. However, this is very risky because the uploaded data is a state document and is confidential, so it is necessary to encrypt the file.

Previous research has discussed securing files using the cryptographic algorithm "AES-128" in various different case studies and also using the cryptographic method, namely AES-128 [3], [4]. The difference from previous research is that this research uses a combination of RSA and AES algorithms which are expected to increase the security and integrity of the uploaded file.

This research aims to secure files uploaded by Regional Apparatus Organizations so that they can help the Regional Inspectorate of Central Sulawesi Province in strengthening data security, thus minimizing data leaks that occur.

2. Methods

3.1 Cryptography

Cryptography is a mathematical engineering science that deals with information security, namely confidentiality, data integrity, and also authentication [5]. Cryptography is increasingly developing with problems that are often faced so the term cryptography is used to characterize secret activities in sending messages. Encryption is a term for scrambling messages and decryption is a term for tidying up messages [6]. Cryptography has 2 types, namely symmetric and asymmetric. Symmetric keys are used in the encryption process as well as decryption. The asymmetric key has 2 keys, namely the public key and the private key where the public key is used for encryption while the private key is used for decryption [7].

3.2 Encryption and Decryption

Encryption is the process of securing information by making information unreadable. Encryption can also be defined as converting plaintext into ciphertext [8]. Decryption in the security world is the process of converting ciphertext into plaintext so it can be defined that decryption is the opposite of encryption [8]. The following is a diagram depicting the data encryption process and the data decryption process:

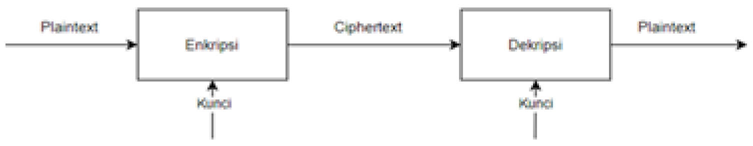

Fig. 1. Encryption and Decryption Process

3.3 AES Algorithm

AES is an algorithm used to secure data. Data security is done by randomizing messages and this AES algorithm is a data security process owned by cryptographic algorithms. AES has many steps in randomizing data, one of which is using permutation and substitution with the S-box table [9]. AES is an algorithm that uses blocks in the process with a block size of 4×4. AES encrypts a 16-byte block by consuming approximately the same amount of time even though the input to the block varies so that the best case is O(1). O(n) is the worst case for AES if the message is more than 16 bytes so the message needs to be split into n fractions [10].

3.4 AES Algorithm Process

AES has several processes, namely key expansion, where the given key is expanded to fulfill the substitution of each round key. At the beginning of encryption, 16-byte of plaintext is converted into a 4×4 matrix called state. Then AES encrypts performing 4 transformations namely AddRoundKey, SubBytes, ShiftRows, and MixColumns. Here are the operations in AES:

- *AddRoundKey* is XOR between the *state* and the *cipher key.*
- *There are n* revolutions - 1 time where the revolutions depend on the key length. The process that occurs in *n* turns - 1 is:
  1. *SubBytes* is the substitution of *bytes* with the AES S-Box. The S-Box is a *lookup* table for

byte substitution.

2. *ShiftRows* is shifting the *state rows* to the left as much as the location of the row, if the *state row is* at index 0, then the row is not shifted.
3. *MixColumns* randomizes the data in each *state* column with the *mixColumns array*.
4. *AddRoundKey*.

- Then the last round of the process is:
  1. *SubBytes*.
  2. *ShiftRows*
  3. *AddRoundKey*

The result of the loop is 1 encrypted state that is 16 bytes of plaintext. If the plaintext is more than 16 bytes then re-do the loop above. For decryption the functions in the AES encryption process are inverse:

- *AddRoundKey*.
- N rounds - 1 time as in encryption where the rounds depend on the key length. The process is:
  1. *InvShiftRows* is the same as the *ShiftRows* encryption function but in the opposite direction, *InvShiftRows* shifts right.
  2. *InverseSubBytes* is the same as the *SubBytes* function but the substitution operation that *InverseSubBytes* performs with the *inverse* S-Box of AES.
  3. *AddRoundKey*.
  4. *InverseMixColumns* randomizes the data in each *state* column with the *invMixColumns array*.
- In the last round of the process viz:
  1. *InvShiftRows*.
  2. *InvSubBytes*.
  3. *AddRoundKey*.

The following is an image of the encryption and decryption of the AES algorithm:

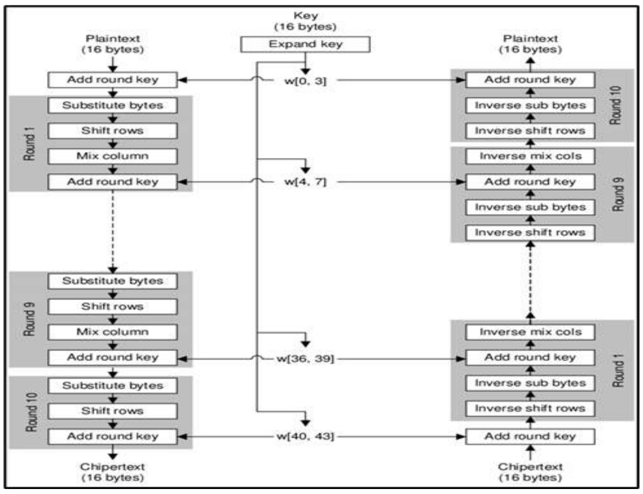

Fig. 2. AES Encryption and Decryption Process[11].

Here is the AES S-box lookup table:

Table 1. AES S-box table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

The following is an example of a 128-bit AES algorithm, where the plaintext contains 16 characters as follows:

- Plaintext: INFORMATIKAUNTAD
- Key: UNIVTADULAKO'19'

Plaintext will be converted into a 4×4 state, each cell contains 1-byte or 2-hex where 1 character is 1-byte, then:

Table 2. Plaintext in state form

49	4E	46	4F
52	4D	41	54
49	4B	41	55
4E	54	41	44

The key will change to:

Table 3. Key in array form

55	4E	49	56
54	41	44	55
4C	41	4B	4F
27	31	39	27

First, the key is expanded first, for example, k[0] contains the first four elements of the key i.e. k[0] = [55, 4E, 49, 56], k[1] contains the next four elements, and so on. Will be looped for example i = 4 where i < 44 then i plus 1, do it:

- Store k[i-1] into a variable e.g.: temp = k[i-1]
- If i mod 4 = 0 then:
  1. Shift temp one byte to the left, e.g. temp becomes [31, 27, 39, 27]

2. Make a substitution with S-Box, e.g. [31,27,39,27] becomes [C7,CC,12,CC]
3. XOR the above result with the i/4th round constant, e.g. [C7,CC,12,CC]⊕[RC[i/4],0,0,0]=[C6,CC,12,CC]. RC is a round constant. RC has an initial value such as RC=[01,02,04,08,10,20,40,80,1B,36], if more RC is needed, then RC[i/4]=2·RC[i/4-1] where the dot represents multiplication in GF(28) (Galois Field).
4. Save the substitution result in a variable e.g. temp

- XOR k[i-4] with variable temp

From the loop above, the key expansion form becomes:
Round 0 key: [55, 4E, 49, 56, 54, 41, 44, 55, 4C, 41, 4B, 4F, 27, 31, 39, 27]
1st round key: [93, 5C, 85, 9A, C7, 1D, C1, CF, 8B, 5C, 8A, 80, AC, 6D, B3, A7]
Round 2 key: [AD, 31, D9, 0B, 6A, 2C, 18, C4, E1, 70, 92, 44, 4D, 1D, 21, E3]
Round 3 keys: [0D, CC, C8, E8, 67, E0, D0, 2C, 86, 90, 42, 68, CB, 8D, 63, 8B]
4th round key: [58, 37, F5, F7, 3F, D7, 25, DB, B9, 47, 67, B3, 72, CA, 04, 38]
Round 5 keys: [3C, C5, F2, B7, 03, 12, D7, 6C, BA, 55, B0, DF, C8, 9F, B4, E7]
6th round key: [C7, 48, 66, 5F, C4, 5A, B1, 33, 7E, 0F, 01, EC, B6, 90, B5, 0B]
7th round key: [E7, 9D, 4D, 11, 23, C7, FC, 22, 5D, C8, FD, CE, EB, 58, 48, C5]
8th round key: [0D, CF, EB, F8, 2E, 08, 17, DA, 73, C0, EA, 14, 98, 98, A2, D1]
Round 9 keys: [50, F5, D5, BE, 7E, FD, C2, 64, 0D, 3D, 28, 70, 95, A5, 8A, A1]
Round 10 keys: [60, 8B, E7, 94, 1E, 76, 25, F0, 13, 4B, 0D, 80, 86, EE, 87, 21]
After the key has been expanded, then the plaintext that is converted into a 4×4 state will be encrypted in the following way:

- *AddRoundKey* XORs the 0th round key with *state*

Table 4. AddRoundKey Function Example

49	4E	46	4F		55	4E	49	56		1C	00	0F	19
52	4D	41	54	?	54	41	44	55	=	06	0C	05	01
49	4B	41	55		4C	41	4B	4F		05	0A	0A	1A
4E	54	41	44		27	31	39	27		69	65	78	63

- Then the state enters the round loop for 10 rounds, namely:

1. *SubBytes*, substituting with S-Box then becomes

Table 5. Example of *SubBytes* Function

9C	63	76	D4
6F	FE	6B	7C
6D	67	67	A2
F9	4D	BC	FB

2. *ShiftRows shifts the state row. The state here is in the form of a transposition*

Table 6. *ShiftRows* Function Example

9C	63	76	D4		9C	FE	67	FB
6F	FE	6B	7C	>	6F	67	BC	D4
6D	67	67	A2		6D	4D	76	7C
F9	4D	BC	FB		F9	63	6B	A2

3. *MixColumns* randomizes the data with a custom matrix.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 0 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 9C \\ FE \\ 67 \\ FB \end{bmatrix} = \begin{bmatrix} A6 \\ 29 \\ BA \\ CB \end{bmatrix}$$

$$(02 \cdot 9C) \oplus (03 \cdot FE) \oplus (01 \cdot 67) \oplus (01 \cdot FB) = A6$$
$$(01 \cdot 9C) \oplus (01 \cdot FE) \oplus (03 \cdot 67) \oplus (01 \cdot FB) = 29$$
$$(01 \cdot 9C) \oplus (01 \cdot FE) \oplus (02 \cdot 67) \oplus (03 \cdot FB) = BA$$
$$(03 \cdot 9C) \oplus (01 \cdot FE) \oplus (01 \cdot 67) \oplus (02 \cdot FB) = CB$$

Where points represent multiplication in GF(28). Then for the whole table, it becomes:

Table 7. Results of *MixColumns* Function

A6	29	BA	CB
1F	AA	0C	D9
0B	17	4E	7E
85	20	B1	47

4. *AddRoundKey*, where the XOR key is the expanded key. The key is used according to the round, since this is the first round, the key used is the 1st round key.

Table 8. XOR Result with Expansion Key

A6	29	BA	CB		93	5C	85	9A		35	75	3F	51
1F	AA	0C	D9	?	C7	1D	C1	CF	=	D8	B7	CD	16
0B	17	4E	7E		8B	5C	8A	80		80	4B	C4	FE
85	20	B1	47		AC	6D	B3	A7		29	4D	02	E0

By looping above, the plaintext changes each round as follows:
Round - 1: [35, 75, 3F, 51, D8, B7, CD, 16, 80, 4B, C4, FE, 29, 4D, 02, E0]
Round 2: [87, 2B, E6, C6, C0, 78, 4C, 1A, 6C, B8, 9F, 10, A6, FE, AE, 5A]
Round 3: [99, 70, 17, D1, EC, 01, 12, 02, DC, 86, 53, F2, 68, E1, A1, B0]
Round 4: [11, EA, 94, 9A, 78, F9, 89, 98, 3A, 56, 7B, C4, 4B, E4, 19, 66]
Round 5: [81, 3E, FE, F4, C4, 08, BA, BD, 7E, 27, D7, 83, 9C, D7, F1, 31]
Round 6: [46, F1, 14, E0, AD, 82, E2, 1F, 50, 4C, 1F, A3, C4, DA, 05, F7]
Round - 7: [CE, D2, 27, FC, E3, 5C, AE, 1D, 38, E0, 4D, 0D, B9, A4, C0, CC]
Round - 8: [76, A5, 2A, 41, 3E, A5, 43, C9, CE, BE, D0, CB, C3, 5F, F0, CF]
Round 9: [D0, DB, 8E, 8F, E7, 04, 43, 96, 72, DA, C1, 7D, C6, B6, B7, F1]
Round - 10: [10, 79, 9F, 35, 8A, 21, 8C, 83, 53, 05, 14, 10, 32, 57, 9D, DE]

Table 9. Encryption Results from *Plaintext* to *Ciphertext*

49	4E	46	4F		10	79	9F	35
52	4D	41	54	>	8A	21	8C	83
49	4B	41	55		53	05	14	10
4E	54	41	44		32	57	9D	DE

If converted back to text then the result becomes "™óX¢È50QA %yÝ" which was previously "INFORMATIKAUNTAD".

3.5 RSA Algorithm Process

The RSA algorithm is a commonly used asymmetric algorithm. It multiplies two large prime numbers to create a key. The public key is used to encrypt and the private key is used to encrypt [12]. The security of the RSA algorithm lies in factoring large prime numbers which is difficult. This factoring is done to obtain the private key [13].

3.6 RSA Algorithm Process

The RSA algorithm has several processes in encrypting. First, key generation. This key generation is done by selecting two prime numbers *p* and *q* where the two numbers are not the same.

Then the value of$n = p \times q$ where n *is* the modulus of the public key and private key. Then calculate the value of$T(n) = (p-1).(q-1)$ where$T(n)$ is a modulus to find e and d where the 2 variables are exponents of the public key and private key. Then find the value of$e = FPB(e, T(n))$ where e and$T(n)$ are coprime and e is a natural number. Then look for the value of d with$k.e \; mod \; T(n) = 1$ where k *is a* natural number, and if fulfilled then$k = d$. Then after getting the 2 keys, the keys are stored where the public key contains the variables e and n while the private key contains all these variables. Then after both private keys and public keys are stored, the keys can be used for data encryption with the formula$c = p^e \; mod \; n$ where c *is* the ciphertext and then decrypt the data with the formula$p = c^d \; mod \; n$ where p *is* the plaintext.

The following is an example of the RSA algorithm in generating keys and encrypting data in the form of text:

- *Plaintext*: "UNTAD"
- Variable p: 5
- Variable q: 11

After determining the plaintext and prime number variables, then calculate the variables n,$T(n)$, e, and d:

- $n = p \times q = 5 \times 11 = 55$
- $\Theta(n) = (p-1) \times (q-1) = 4 \times 10 = 40$
- $e = FPB(e, T(n)) = 7$ because 7 coprime with 40
- $k \,.\, e \; mod \; T(n) = 1$ if k is 23, then23$\,.\,9 \; mod \; 8 = 1$. Since it satisfies, then $k = d$ *which* is 23.

Then, to encrypt, 2 variables are used, namely the e variable and the n variable where these two variables are public keys. To encrypt the above plaintext as follows:

- Convert the characters into decimals so that they can be entered into the formula, namely U = 37, N = 30, T = 36, A = 17, and D = 20.
- Then input each character into the formula$c = p^e \; mod \; n$ where p is each decimal:
  1. U = 377 mod 55 = 38
  2. N = 307 mod 55 = 35
  3. T = 367 mod 55 = 31
  4. A = 177 mod 55 = 8
  5. D = 207 mod 55 = 15

  After encryption, the character shape becomes UNTAD > VSO8.
- To decrypt it using the formula p= c^d mod n where c is the ciphertext in decimal form:
  1. V = 3823 mod 55 = 37
  2. S = 3523 mod 55 = 35
  3. O = 3123 mod 55 = 31
  4. 8 = 823 mod 55 = 8
  5. ? = 1523 mod 55 = 15

  After decryption, the character shape becomes: VSO8> UNTAD

3. Results and Discussion

3.1 Testing Results

The tests carried out are black box testing and usability testing. Black box testing aims to find out whether the functions contained in the system run well or not. The following are the results of *black box* testing:

- Submission Period Management Testing
  This table is a test that tests the function of verifying files, where the examiner must successfully open the encrypted file and successfully provide valid or invalid information from the file.

Table 10. File Verification Testing Table

What to do	What to Expect	Observation	Results
Change the description of a valid file	The examiner can determine the status of the file by changing the valid description	The examiner successfully determines the status of the file uploaded by the OPD.	Good
Change the invalid file description	The examiner can change the invalid file description	The examiner changes the description of the invalid file and the OPD must upload the file again	Good

- File Upload Testing
  This table is a test that tests the function of uploading files, where the Regional Apparatus Organization must succeed in uploading files and also fail if the time limit expires. File uploads also use an encryption process to secure data from the OPD, so that only users who have access can open the file.

Table 11. File Upload Testing Table

What to do	What to Expect	Observation	Results
OPD does not upload files until the deadline ends	OPD cannot upload	OPD did not successfully upload when the deadline expired	Good
OPD uploads files according to the type of audit	OPD can upload files according to the type of audit	OPD successfully uploads files before the deadline ends	Good
OPD wants to see the file after uploading	OPD can open and view uploaded files	OPD successfully opens the uploaded file	Good
OPD uploads invalid files	OPD can replace uploaded files with valid files	OPD successfully replaces invalid files with valid ones	Good

Usability testing is used to measure encryption and decryption processing tables based on *file* size. The following is a table of encryption processing time:

Table 12. Encryption Processing Time Table

No.	File Size	Encryption Time (seconds)
1	1 MB	0.182
2	5 MB	0.205
3	10 MB	0.234
4	20 MB	0.271
5	50 MB	0.393

In the table above, it can be seen that the encryption time increases as the size of the *file* increases. Here is a graphical image of the encryption processing timetable:

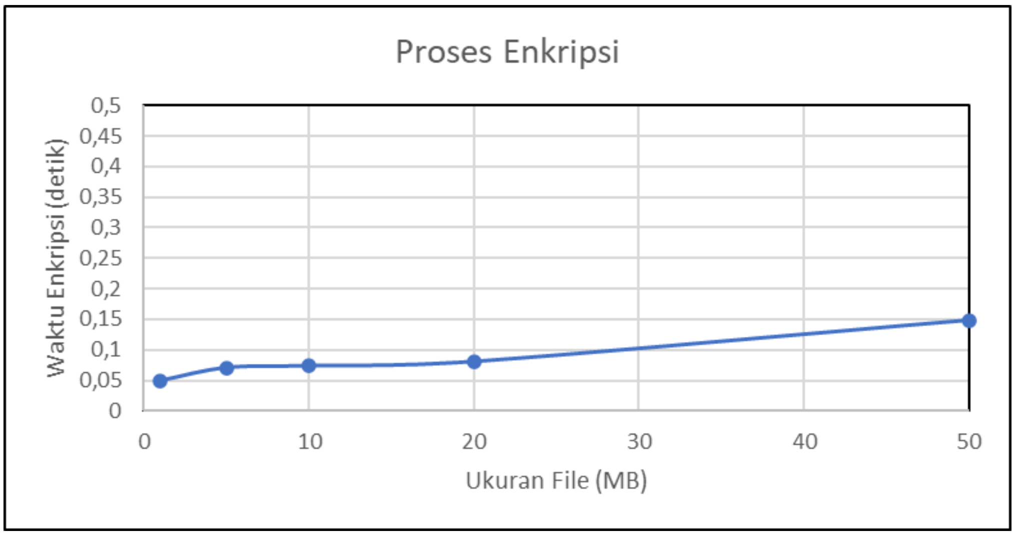

Fig. 3. Graph of Encryption Processing Time

From the figure above, it can be seen that the *file* size affects the encryption time linearly. Where the *file* size of 1MB takes 0.182 seconds, the 5MB file takes 0.205 seconds, and the 50MB file takes 0.393 seconds. The following is a table of decryption processing time:

Table 13. Decryption Processing TimeTable

No.	File Size	Decryption Time (seconds)
1	1 MB	0.050
2	5 MB	0.071
3	10 MB	0.074
4	20 MB	0.081
5	50 MB	0.148

From the table above, it can be seen that the decryption time also increases as the size of the *file* increases, but the time required for decryption is lower than encryption. The following is a graphic image of the decryption processing timetable:

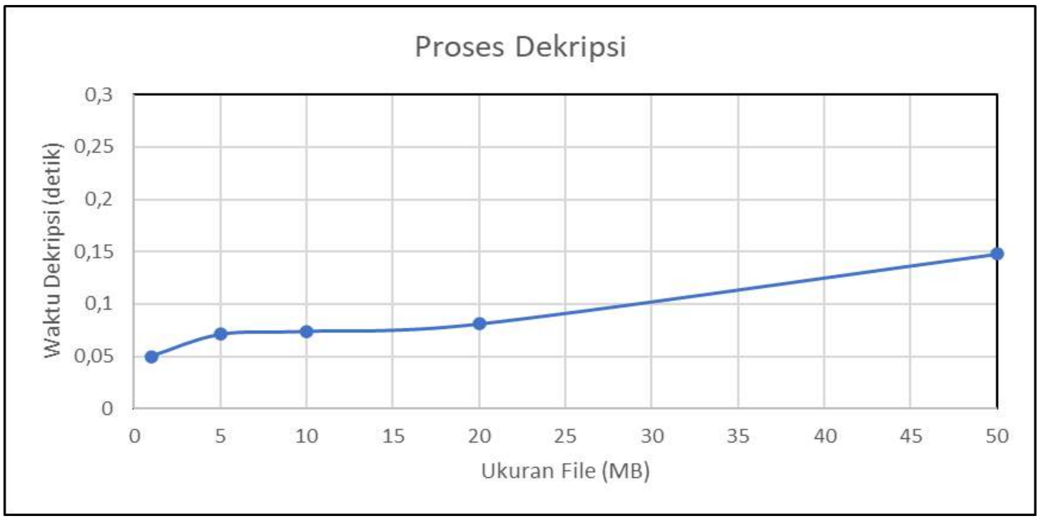

Fig. 4. Decryption Processing Time Chart

From the figure above, it can be seen that the file size affects the encryption time linearly. Where the file size of 1MB takes as much as 0.050 seconds, the 5MB file takes as much as 0.072 seconds, and the 50MB file takes as much as 0.148 seconds.

3.2 Discussion

Based on the results of the tests that have been carried out, the system can run well and can secure files uploaded to the E-Supervision system from OPD (Regional Apparatus Organization) related to each Examiner, namely Irban (Assistant Inspector). Inspectors can also open files that have been encrypted because they have access rights to open them. View files can also be done for anyone who has access rights such as Examiners, OPD, and Admin, so that each function can run properly and correctly. This system is web-based with the PHP programming language and uses the Laravel framework. At the beginning of system operation, every user who has any access rights is required to log in first so that the system can validate the user and grant access rights to the user.

Next, users enter their respective main pages. Admins can add, change, and delete users, both OPD and examiners. The examiner can determine the audit submission period for the relevant OPD and can also manage the files that have been uploaded by the OPD by providing valid or invalid status. OPDs can upload files according to the provisions of the Examiner before the deadline expires. When the OPD uploads the file, it is encrypted first and then stored, while the decryption process is carried out when the OPD, Examiner, or Admin wants to see the file. The file Decryption Encryption process is carried out using the AES algorithm while the key from AES is encrypted using the RSA algorithm key. Previous research that has been done on the encryption of the Land Book Borrowing Information System also successfully encrypts [14]. Encryption and decryption using RSA-AES can be seen using a few milliseconds of time based on the size of the file uploaded by the Regional Apparatus Organization. Decryption using RSA-AES uses less time than encryption. The RSA-AES process begins by decrypting the AES key using the RSA key and then the AES key is used to encrypt as well as decrypt existing files. It can be seen that the encryption and decryption time is affected by the file size and the time grows linearly. Research that measures time complexity such as testing Bawaslu's virtual cooperative using the RC4 algorithm which is based on the Big-O calculation [15].

4. Conclusion

Based on the test results and implementation of securing the E-Supervision system using the RSA-AES algorithm, it can be concluded that the RSA-AES algorithm can be used together to secure different things. RSA secures the key from AES while AES secures the uploaded file. Based on the results of testing and implementation of E-surveillance system security using the RSA-AES algorithm, it can be concluded that 1) The security system built aims to secure files that have been uploaded to the E-Supervision system so that the file is safe from data theft 2) The RSA-AES algorithm can be used together to secure different things. For RSA to secure the key from AES while AES to secure the uploaded file 3) This research uses data from the Inspectorate of Central Sulawesi Province where the data is a lot of sensitive data.

References

[1] I.D.I.Y, "Duties, Functions and Roles of APIP according to Permendagri," Jul. 29, 2020. http://inspektorat.jogjaprov.go.id/about/ (accessed Nov. 03, 2022).

[2] Inspectorate of Banten Province, "Main Tasks and Functions of the Provincial Inspectorate," 2019. https://inspektorat.bantenprov.go.id/tugas-pokok-fungsi (accessed Nov. 03, 2022).

[3] D. Widyawan and Imelda, "FILE SECURITY USING KRIPTOGRAPHY WITH WEB-BASED AES-128 METHOD AT THE NATIONAL COMMITTEE FOR TRANSPORTATION SAFETY," SKANIKA, vol. 4, 2021.

[4] I. A. R. Simbolon, I. Gunawan, I. O. Kirana, R. Dewi, and S. Solikhun, "Application of 128-Bit AES Algorithm in Securing Population Data at the Dukcapil Office of Pematangsiantar City," Journal of Computer System and Informatics (JoSYC), vol. 1, no. 2, 2020.

[5] R. Munir, Cryptography, 2nd ed. Bandung: Informatics, 2019.

[6] H. Mukhtar, Cryptography for Data Security. Deepublish, 2018.

[7] T. H. Saputro, N. Hidayati, and E. I. H. Ujianto, "SURVEY ON ASMETRICAL CRYPTOGRAPHY ALGORITHMS," Polinema Informatics Journal, 2020.

[8] T. Bin Tahir, M. A. Hadi Sirad, and M. Rais, "Encrypt and Decrypt Information System with AES Algorithm Using Laravel Framework," Patria Artha Technological Journal, vol. 4, no. 1, Apr. 2020, doi: 10.33857/patj.v4i1.326.

[9] L. B. Handoko and C. Umam, "TLS-based Email Attachment File Security Using AES and LSB Algorithms," 2022.

[10] P. Pranav, S. Dutta, and S. Chakraborty, "Empirical and Statistical Comparison of Intermediate Steps of AES-128 and RSA in Terms of Time Consumption," Res Sq, 2021, doi: 10.21203/rs.3.rs-189456/v1.

[11] D. Wijaya and Ratnadewi, "Implementation and performance analysis of AES-128 cryptography method in an NFC-based communication system," 2017. [Online]. Available: https://www.researchgate.net/publication/318528672

[12] R. M. Ridwan, H. E. Victor, A. Setiawan, and K. Kunci, "Application of Data Security and Inserted in Images with RSA Algorithm and Modified LSB Based on Android," 2019.

[13] I. Gunawan, "COMBINATION OF CAESAR CIPHER ALGORITHMA AND RSA ALGORITHMA FOR SECURITY OF DOCUMENT FILES AND TEXT MESSAGES," 2018.

[14] Ngemba, H.R., Ulhaq, M.N.D., Hendra, S., Azhar, R., Alamsyah & Laila Rahma. 2024. IMPLEMENTATION OF THE RC4 ALGORITHM ON THE VIRTUAL COOPERATIVE INFORMATION SYSTEM BAWASLU VIRTUAL BAWASLU CENTRAL SULAWESI PROVINCE. *PROSISKO* 11 (1).

[15] Rasmita Ngemba, H., Hendra, S., Gusti Ngurah Agung Kade Dwi Arsana, I. & Information Technology, J. (n.d.). Implementation of MD5 and SHA-256 Data Encryption in the Land Book Lending Information System. *August*.