Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

15th October 2025. Vol. 5. No. 1; e-ISSN: 2808-4519

Implementation of Super Encryption Using Affine Cipher, Playfair Cipher, and RSA on Image Files

Muhamad Arif[1]*, Achmad Fauzi[2], Hermansyah Sembiring[3]

[1,2,3] *Informatics Engineering, STMIK KAPUTAMA*
Jl. Veterans No. 4A-9A, Binjai, North Sumatra, IndonesiaCorresponding
mhdarif146@gmail.com[1], fauzyrivai88@gmail.com[2], hermansyahsembiring240165@gmail.com[3]*

Abstract

This research aims to enhance the security of image files by implementing a super-encryption technique that integrates three cryptographic algorithms from both classical and modern domains: Affine Cipher, Playfair Cipher, and RSA. Each algorithm provides a distinct layer of encryption applied sequentially—starting with byte-value transformation using the Affine Cipher, followed by byte-pair substitution through the Playfair Cipher, and concluding with public-key RSA encryption. The proposed approach was evaluated on image files while ensuring both integrity and byte-level equivalence between the original and decrypted files. The implementation was developed as a desktop application in Visual Basic .NET, featuring separate modules for encryption and decryption, along with structured displays of results and process logs. Experimental results indicate that this super-encryption method successfully preserves file integrity and significantly increases cryptographic complexity without altering file size. System security is substantially improved, as the combined algorithms make the encrypted data highly resistant to analysis without complete knowledge of the underlying structure and encryption keys. This approach offers a viable alternative for securing sensitive image files, such as identity documents and medical records.

Keywords: *Affine Cipher, Image Files, Playfair Cipher, RSA, Super Encryption*

1. Introduction

In today's highly interconnected digital era, data security has become a strategic issue that cannot be overlooked. Image files, as a medium for storing visual information, are now widely used across various sectors—ranging from biometric identification and healthcare services to cyber security systems. However, the growing prevalence of digital threats, such as theft, manipulation, and data breaches, indicates that conventional protection mechanisms are no longer sufficient to guarantee complete data security..

One of the main causes of weak data protection lies in the use of encryption methods that remain simple and static, making their patterns easily recognizable by attackers, especially with the aid of artificial intelligence (AI). Therefore, a more complex, adaptive, and layered encryption approach—known as super encryption—is required. This technique combines multiple cryptographic algorithms to reinforce data protection structures, expand the key space, and significantly complicate analysis efforts by unauthorized parties..

Previous studies have demonstrated the effectiveness of algorithm combinations, such as RSA and Diffie-Hellman, in securing image files [1]. Building on this rationale, the present study proposes a combination of three distinct algorithms: Affine Cipher, Playfair Cipher, and RSA. The Affine Cipher is employed to transform the initial byte values, the Playfair Cipher is used to shuffle byte pairs, and RSA adds an additional layer of security through public-key cryptography. This approach is implemented as a desktop application developed in Visual Basic .NET and evaluated on image files with various common extensions, including BMP, PNG, and JPEG.

With the application of this super-encryption technique, the system is expected to maintain the confidentiality, authenticity, and integrity of image files while enhancing resistance to cryptanalysis attacks. This research aims to make a tangible contribution to the development of more robust and adaptive image file security systems, capable of addressing the evolving challenges of today's digital security landscape.

2. Theoretical Foundation

2.1 Cryptography

Cryptography is one of the most common methods used to protect the confidentiality of data in information systems. In general, cryptography is the study of techniques for transforming original data (plaintext) into an unreadable form (ciphertext) that cannot be understood by unauthorized parties without the use of a specific key [2].

2.2 Image

An image is a two-dimensional visual representation of an object composed of individual pixels. Each pixel stores information about color and intensity, forming the overall picture. In its digital form, an image is represented as binary data that can be processed by a computer.

To maintain confidentiality during transmission, digital images are often encrypted using cryptographic algorithms to prevent unauthorized access and modification [3].

2.2.1 Types of Image Formats

Commonly used image formats include JPG, PNG, and BMP. JPG (Joint Photographic Experts Group) is a popular format for storing images with high compression while maintaining good visual quality, although it does not support transparency. PNG (Portable Network Graphics) offers superior transparency handling and lossless compression, making it suitable for digital graphics and web display. Meanwhile, BMP (Bitmap) stores images in full pixel representation without compression, resulting in high visual quality but large file sizes, and is often used in Windows systems for technical purposes that require high precision[4].

2.3 Affine Cipher

The Affine Cipher is one of the classical cryptographic algorithms categorized as a substitution cipher, in which each character in the plaintext is transformed into another character through a mathematical operation. Unlike the Caesar Cipher, which uses only a single key in the form of a fixed shift, the Affine Cipher employs two keys, namely a and b, in its encryption formula:

$$\boldsymbol{E(x) = (a.x + b) mod\ m} \quad \textbf{(1)}$$

Explanation:

E(x) = encryption result (ciphertext)
x = numerical value of the character
a = multiplicative key (must be coprime with m)
b = additive key
m = the total number of characters in the alphabet (typically 26 for letters, or 256 for image files)

The decryption process uses the following formula:

$$\boldsymbol{D(y) = a - 1\ .(y - b) mod\ m} \quad \textbf{(2)}$$

Explanation:

D(y) = decryption result (plaintext)
y= numerical value of the cipher text character
a -1 = modular inverse of a (the modular inverse satisfies the condition $a \times a^{-1} \equiv 1 \pmod{256}$)

The Affine Cipher is effective for understanding the fundamental concepts of cryptography and can provide additional security when combined with other methods in a super-encryption system[5]. Previous research has shown that applying the Affine Cipher as the initial stage in a layered encryption system can produce a complex initial cipher text and support multi-layered security in the transmission of digital data[6] .

2.4 Playfair Cipher

The Playfair Cipher algorithm is a form of classical cryptography that encrypts data in pairs of characters (digraphs). In this study, the algorithm is modified to be applicable for encrypting digital image data based on byte values (0–255). The key matrix is expanded to a 16×16 size to represent all possible byte values. This matrix is constructed by arranging the values from the numerical key after removing duplicates, followed by the numbers 0 through 255 without repetition, organized sequentially from left to right and from top to bottom.

The encryption process begins by dividing the image data into pairs of bytes. If the number of bytes is odd or there is an identical byte pair, a byte with the value 0 is inserted as padding. Each byte pair is then encrypted according to specific rules: if both bytes are located in the same row of the matrix, each byte is shifted one position to the right; if they are in the same column, each is shifted downward; and if they are in different rows and columns, each byte swaps columns to form a rectangle. The decryption process reverses these encryption rules, shifting to the left for byte pairs in the same row, shifting upward for those in the same column, and swapping columns—similar to the encryption process—for bytes in different positions. With this approach, the modified Playfair cipher algorithm can be effectively applied for the encryption and decryption of image data, ensuring both security and consistency of byte structure [7]. Previous studies have shown that the Playfair Cipher can be adapted to support non-text characters or values, such as ASCII data and digital image bytes, by enlarging the matrix beyond its original size and adjusting the rules for processing data pairs[8] .

2.5 RSA (Rivest-Shamir-Adleman)

RSA (Rivest–Shamir–Adleman) is an asymmetric cryptographic algorithm developed in 1977 and is still widely used today to secure digital data. Unlike symmetric algorithms, which use a single key for both encryption and decryption, RSA relies on two distinct keys: a public key for encryption and a private key for decryption. The security of this algorithm lies in the computational difficulty of factoring large numbers into their prime factors[9].

1. Key Generation

Select two large prime numbers: p and q.
Compute the modulus n as:

$n = p \times q$

Calculate Euler's totient (φ) of n:

$\varphi(n) = (p-1)(q-1)$

Choose an integer e such that:

$1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$

(e serves as the public key)
Compute d as the modular inverse of e with respect to φ (n):
d ≡ e^1 mod φ (n)
(d serves as the private key)

2. Encryption
To encrypt a plaintext message M (which must be represented as an integer), use the public key (e, n):

$$C = M^e \bmod n \quad (3)$$

Explanation:
C = ciphertext produced by the encryption process
M = plaintext to be encrypted

3. Decryption
To convert the ciphertext back into plaintext, use the private key (d, n):

$$M = C^d \bmod n \quad (4)$$

Explanation:
M = the original message that has been decrypted from C

In this study, RSA is employed as the final stage in the super-encryption process following the Affine Cipher and Playfair Cipher. The role of RSA in this system is to lock the previous encryption result into a large numerical value that can only be decrypted using the corresponding private key. This strategy provides an additional layer of security in the transmission of digital data, particularly image files. Previous research has shown that RSA is highly effective when used as part of a hybrid system, as it can significantly enhance the security of digital data. Although that study combined RSA with the Blum Blum Shub (BBS) algorithm, the fundamental principle remains the same—RSA is well-suited as the final layer in a multi-layered encryption system, as implemented in this research for securing image files[10]. Thus, RSA plays an important role in ensuring the integrity and confidentiality of data at the final stage of the implemented super-encryption system.

2.6 Super Encryption Technique
Super encryption is a layered approach in cryptography that combines more than one algorithm sequentially to enhance the strength and complexity of data protection. In this system, the encryption process is carried out in three main stages: first, the Affine Cipher algorithm is used to encode data at the byte level as an initial form of substitution; second, the result is re-encrypted using the Playfair Cipher algorithm, which transforms the data into pairs of values, thereby strengthening both diffusion and confusion aspects; and finally, the data that has passed through these two layers is secured using the RSA algorithm, which is based on public-key cryptography, making it resistant to classical cryptanalysis attacks. The combination of these three algorithms produces an encryption system that is far more complex and difficult to break, especially when the keys are unknown.

3. Analysis and Design
3.1 Research Method
This study employs an experimental method with a systematic and empirical approach to examine the effectiveness of combining the Affine Cipher, Playfair Cipher, and RSA cryptographic algorithms in the super-encryption process for image files. The purpose of this method is to comprehensively evaluate the security level, integrity, and accuracy of the encryption and decryption processes. The research steps include collecting data in the form of image files (PNG, JPG, and BMP formats), designing a super-encryption algorithm consisting of three sequential encryption stages (Affine → Playfair → RSA), and implementing the program using the Visual Basic .NET programming language. The testing process is carried out by comparing the decrypted file with the original file on a bit-by-bit basis using the Binary Viewer application to ensure that no data alterations occur. The evaluation also covers process speed and algorithm complexity.

3.2 System Design
The flowchart is designed to illustrate the process flow of encrypting and decrypting image files using the Affine Cipher, Playfair Cipher, and RSA algorithms. This diagram facilitates a step-by-step and structured understanding and implementation of the system.

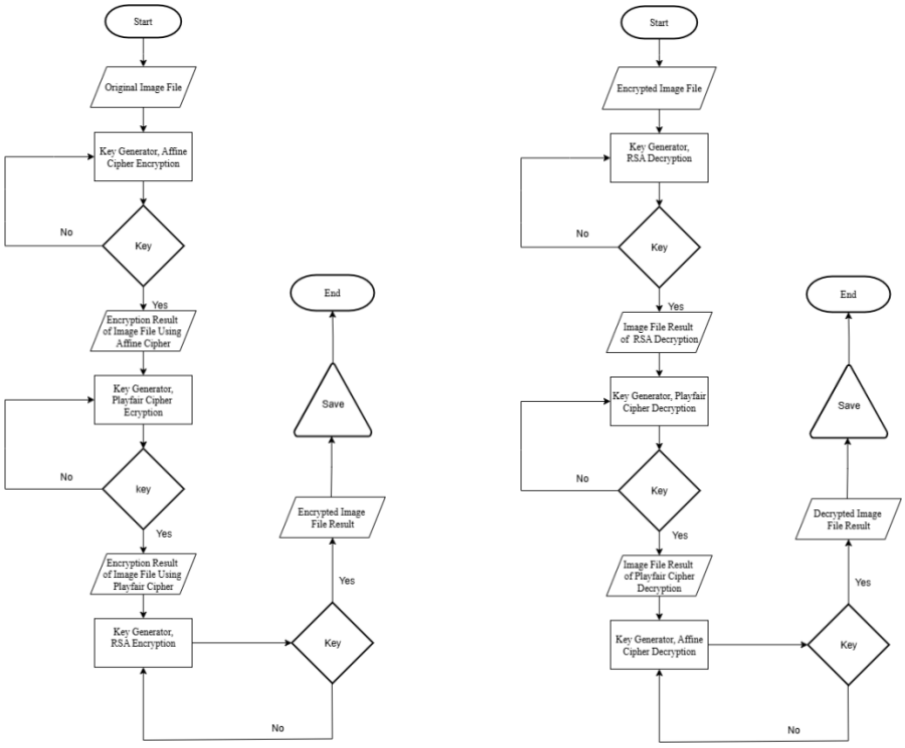


Fig. 1: Flowchart of Encryption and Decryption Process

3.3 Algorithm Analysis

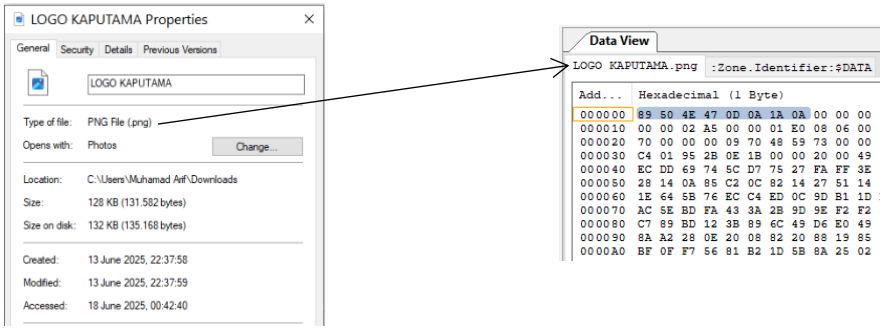


Fig. 2: The original image file was converted using Binary Viewer

At this stage, an analysis is conducted on the super-encryption process by combining the Affine Cipher, Playfair Cipher, and RSA algorithms for securing digital image files. The test image used is in PNG format with the file name “LOGO KAPUTAMA.PNG” and a size of 128 KB. Prior to the encryption process, the image file is first converted into hexadecimal form using the Binary Viewer software. From this image, several hexadecimal values are sampled and used as the plain file for the super-encryption process employing the Affine Cipher, Playfair Cipher, and RSA algorithms. These values are then converted into decimal form and displayed

Table. 1: Hexadecimal to Decimal Conversion

Hexadecimal	Decimal
89	137
50	80
4E	78
47	71
0D	13
0A	10
1A	26
0A	10

3.3.1 Affine Cipher Encryption Calculation

The plain file from Table 1 is 137 80 78 71 13 10 26 10.
It is encrypted using the Affine Cipher with the following keys:
a = 5 (since the value of a is relatively prime)
b = 8 (as the shift value, or random offset)
m = 256 (since the alphabet used consists of 256 characters)
The encryption of the plain file is calculated using the formula:
E = a x + b (mod m)

For the image file encryption process using the Affine Cipher algorithm, the calculation is as follows.
Plain file 137 80 78 71 13 10 26 10
E (137)= (5× 137+ 8) mod 256 = 693 mod 256 = 181
E (80) = (5 x 80 + 8) mod 256 = 408 mod 256 = 152
E (78) = (5 x 78 + 8) mod 256 = 398 mod 256 = 142
E (71) = (5 x 71 + 8) mod 256 = 363 mod 256 = 107
E (13) = (5 x 13 + 8) mod 256 = 73 mod 256 = 73
E (10) = (5 x 10 + 8) mod 256 = 58 mod 256 = 58
E (26) = (5 x 26 + 8) mod 256 = 138 mod 256 = 138
E (10) = (5 x 10 + 8) mod 256 = 58 mod 256 = 58
The resulting cipher file is **181 152 142 107 73 58 138 58**

3.3.2 Playfair Cipher Encryption Calculation

The plain file resulting from the Affine Cipher encryption is 181 152 142 107 73 58 138 58 These values are then paired into byte pairs as follows (181 152) (142 107) (73 58) (138 58). The numerical key used to construct the Playfair matrix is as follows 5, 8, 9,12,13.

Table. 2 : Playfair Cipher 16×16 Encryption Process

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	5	8	9	12	13	0	1	2	3	4	6	7	10	11	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
10	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
11	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
12	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
13	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
14	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
15	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Encryption Process:
Pair (181, 152):
Position 181 → row 11, column 5
Position 152 → row 9, column 8
Rectangle rule → Replace with: 184 and 149

Pair (142, 107):
Position 142 → row 8, column 14
Position 107 → row 6, column 11
Rectangle rule → Replace with: 139 and 110

Pair (73, 58):
Position 73 → row 4, column 9
Position 58 → row 3, column 10
Rectangle rule → Replace with: 74 and 57

Pair (138, 58):
Position 138 → row 8, column 10
Position 58 → row 3, column 10
Same column→ Replace with the numbers below 154 and 74
The Playfair Cipher encryption result is **184, 149, 139, 110, 74, 57, 154, 74**

3.3.3 RSA Encryption Calculation

In the final stage of the super-encryption process, the RSA algorithm is applied to secure the output produced by the Playfair Cipher. RSA is an asymmetric cryptographic algorithm that uses two distinct keys: a public key for the encryption process and a private key for

the decryption process. The security of RSA relies on the complexity of factoring large prime numbers, which makes it widely used in modern security systems.

A. RSA Key Generation
Before performing the encryption, the RSA key generation process is carried out, consisting of the following steps:
1. Selecting Two Prime Numbers
p = 61
q = 53
2. Calculating n and φ (n)
$n = p \times q = 61 \times 53 = 3233$
$\varphi(n) = (p - 1) \times (q - 1) = 16 \times 10 = 3120$
3. Determining the Public Key (e) Select a value of e such that
$1 < e < \varphi(n)$
GCD (e, φ (n)) = 1 (relatively prime)
Thus, we choose:
e = 17, because gcd (17,3120)= 1
4. Calculating the Private Key (d) Find a value of d that satisfies
$d \equiv e^{-1} \bmod \varphi(n)$
$d \equiv 17^{-1} \bmod 3120 = 2753$
Because:
$(7 \times 2753) \bmod 3120 = 1$
Conclusion:
Public Key: (e =17, n = 3233)
Private Key: (d = 2753, n = 3233)
The plain file resulting from the Playfair Cipher encryption is 184, 149, 139, 110, 74, 57, 154, 74.
c[0] = m^e mod n = 184^17 mod 3233= 3112
c[1] = m^e mod n = 149^17 mod 3233=918
c[2] = m^e mod n = 139^17 mod 3233= 2860
c[3] = m^e mod n = 110^17 mod 3233= 2235
c[4] = m^e mod n = 74^17 mod 3233= 1877
c[5] = m^e mod n = 57^17 mod 3233= 1175
c[6] = m^e mod n = 154^17 mod 3233= 1260
c[7] = m^e mod n = 74^17 mod 3233= 1877
The RSA encryption result is **3112 918 2860 2235 1877 1175 1260 1877**

3.3.4 RSA Decryption Calculation
The RSA decryption stage is performed to convert the cipher file from the previous encryption process back into its plain file form. This process uses the keys d = 2753 N = 3233, with the following formula

$$\mathbf{P = c\ ^d \bmod n}$$

Using the cipher file from the previous encryption, namely:
3112 918 2860 2235 1877 1175 1260 1877
p[0] = c ^d mod n = 3112^2753 mod 3233= 184
p[1] = c ^d mod n = 918^2753 mod 3233= 149
p[2] = c ^d mod n = 2860^2753 mod 3233= 139
p[3] = c ^d mod n = 2235^2753 mod 3233= 110
p[4] = c ^d mod n = 1877^2753 mod 3233= 74
p[5] = c ^d mod n = 1175^2753 mod 3233= 57
p[6] = c ^d mod n = 1260^2753 mod 3233= 154
p[7] = c^d mod n =1877^2753 mod 3233= 74

Thus, the decryption result is **184, 149, 139, 110, 74, 57, 154, 74**

3.3.5 Playfair Cipher Decryption Calculation
This stage aims to revert the cipher file produced by the Playfair Cipher encryption back to its original values prior to that process. Decryption is carried out using the inverse Playfair Cipher rules with the same key matrix, which is based on the random numbers 5, 8, 9, 12, and 13. A 16×16 matrix containing the numbers 0–255 is constructed from this key and reused for the decryption process. The cipher file to be decrypted is the result of the previous RSA decryption stage, namely: 184, 149, 139, 110, 74, 57, 154, 74.

Table. 3: Playfair Decryption Process

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	5	8	9	12	13	0	1	2	3	4	6	7	10	11	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
10	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
11	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
12	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
13	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
14	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
15	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Decryption Process:
Pair (184, 149)
Position→ 184 row 11, column 5
Position→ 149 row 9, column 8
Rectangle rule → Replace with 181 and 152

Pair (139, 110)
Position→ 139 row 9, column 9
Position→ 110 row 8, column 12
Rectangle rule → Replace with 142 and 107

Pair (74, 57)
Position→ 74 row 6, column 2
Position→ 57 row 5, column 5
Rectangle rule → Replace with 73 and 58

Pair (154, 74)
Position→ 154 row 9, column 5
Position→ 74 row 6, column 2
Rectangle rule → Replace with 138 and 58
The decryption result of the Playfair Cipher is 181 152 142 107 73 58 138 58

3.3.6 Affine Cipher Decryption Calculation

The final step of the super-encryption process is decryption using the Affine Cipher method. The purpose of this stage is to recover the image data in its original form, prior to undergoing the layered encryption sequence. The decryption is carried out using the following parameters.

a = 5
b = 8
m = 256 (since the image file is processed in bytes)
a^{-1} = 205 (the modular inverse of 5 modulo 256)

Decryption Process for the cipher file 181, 152, 142, 107, 73, 58, 138, 58
Each value is calculated using the formula:

$$P = 205 \times (C - 8) \bmod 256$$

Calculations:
P = 205 x (181-8) mod 256= 205 x 173 mod 256 = 137
P = 205 x (152-8) mod 256= 205 x 144 mod 256 = 80
P = 205 x (142 -8) mod 256= 205 x 134 mod 256 = 78
P = 205 x (107 -8) mod 256= 205 x 99 mod 256 = 71
P = 205 x (73 -8) mod 256= 205 x 65 mod 256 = 13
P = 205 x (58 -8) mod 256= 205 x 50 mod 256 = 10
P = 205 x (138 -8) mod 256= 205 x 130mod 256 = 26
P = 205 x (58-8) mod 256= 205 x 50 mod 256 = 10
Thus, the decryption result is **137 80 78 71 13 10 26 10**

Table. 4 : Decimal to Hexadecimal Conversion

Decimal	Hexadecimal
137	89
80	50
78	4E
71	47
13	0D
10	0A
26	1A
10	0A

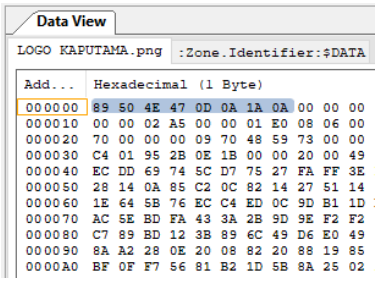


Fig. 3: Binary Viewer Result

The decryption result of the Affine Cipher is the original plain file that had previously undergone the layered encryption process. The successful recovery of this initial data serves as evidence that the implementation of super encryption using the combined Affine Cipher, Playfair Cipher, and RSA algorithms operates accurately, consistently, and reliably in securing image files.

4. Implementation and Discussion

4.1 Testing

The encryption process implementation begins from the Main Menu Form, which provides three main options: Encrypt, Decrypt, and Exit. The user selects the Encrypt option to open the Encryption Form, then chooses an image file (PNG, JPG, or BMP) using the Browse button. Once the file is loaded, its path is displayed as an indication that the file has been successfully read.
The first stage is encryption using the Affine Cipher, where the user enters the key values a and b. After the Affine Encrypt button is pressed, the system processes all image bytes and displays the result in the log.

The next step is Playfair Cipher encryption, with input in the form of numerical keys that will generate the substitution matrix. The Playfair Encrypt button processes the data from the previous stage and displays the second encryption result.
The final stage is RSA encryption, which begins with automatic key generation via the Generate Key button. Once the key values are generated, the user enters them into the RSA input fields and clicks the RSA Encrypt button to proceed. The final encryption result is displayed in the log. After all three stages are completed, the user can save the super-encrypted file as a new file with the .enc extension, indicating that the file has been secured through three sequential layers of algorithm

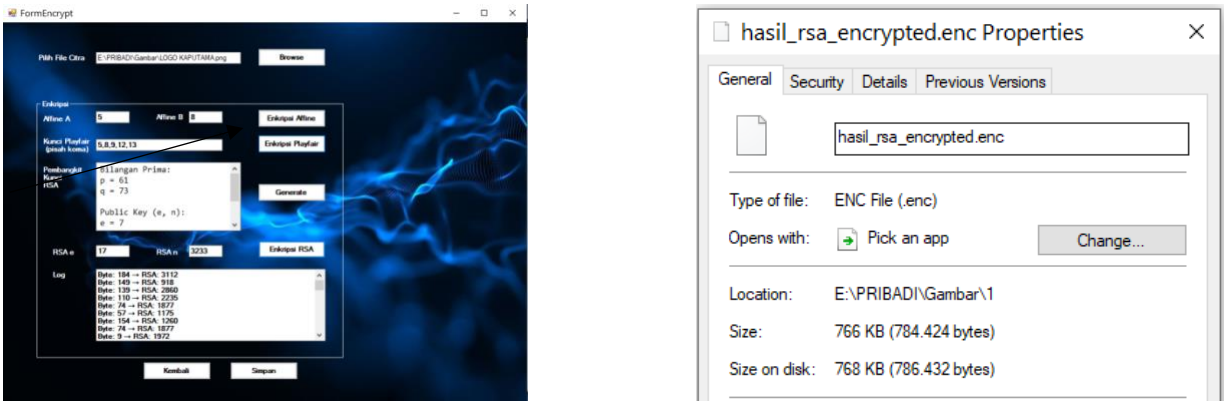


Fig. 4: Encryption Testing Results and Encrypted File Output

The decryption process is carried out through the Decryption Form, which has a structure and flow that is the reverse of the encryption process. The user loads the super-encrypted file by clicking the Browse button, then enters the same keys used during the encryption process. The first step is RSA decryption, where the user enters the corresponding private key to reverse the RSA encryption process. After clicking the RSA Decrypt button, the output in the form of Playfair-encrypted data is displayed in the log.

The second step is Playfair Cipher decryption. The system processes the RSA output using the same numerical key matrix as in the encryption stage. By clicking the Playfair Decrypt button, the data is reverted to the form produced by the Affine Cipher stage.
The final step is Affine Cipher decryption, in which the user re-enters the previously used values of a and b. Clicking the Affine Decrypt button executes the final process to restore all image file bytes to their original form.
Once all processes are complete, the user can save the decryption result as an image file (.png). Testing has confirmed that the decrypted file is bit-by-bit identical to the original image file, both in size and content, with no data loss

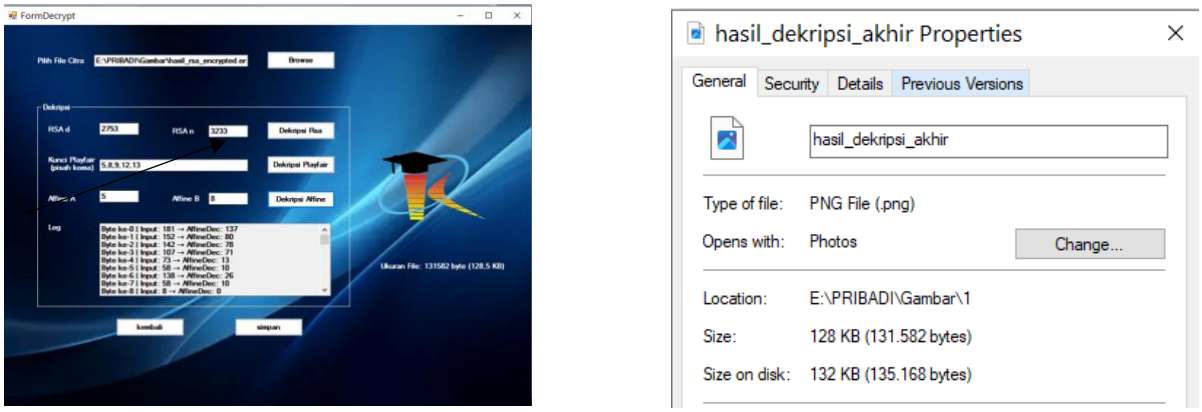


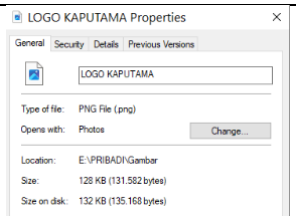
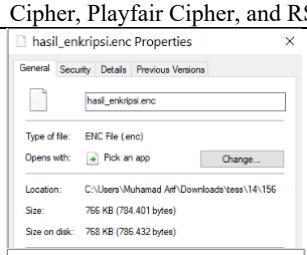
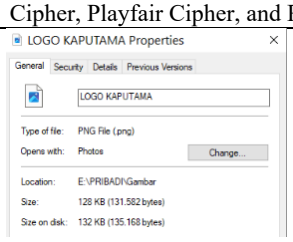
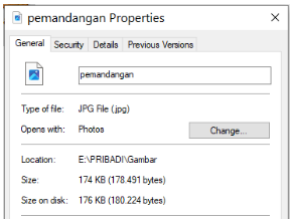
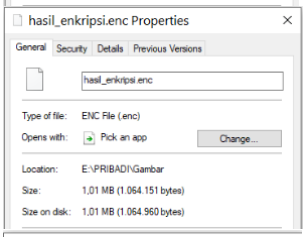
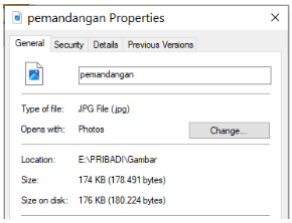
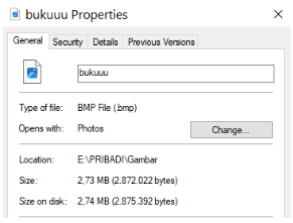
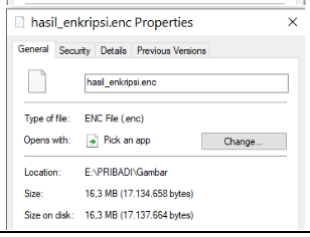
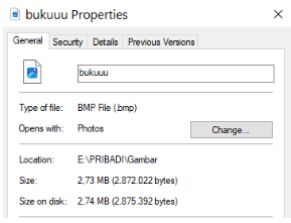
Fig. 5: Decryption Testing Results and Decrypted File Output

4.2 Test Result

Testing was conducted using several image files with .png, .jpg, and .bmp extensions to ensure that the encryption and decryption processes function properly and produce files identical to the originals. The results showed that both the encryption and decryption processes successfully preserved the file size and structure without any byte changes that could cause corruption to the image files.

Comprehensive testing was performed, starting from file selection, step-by-step encryption using the Affine Cipher, Playfair Cipher, and RSA algorithms, and followed by decryption in reverse order. Verification was carried out through visual inspection of the decrypted images, file size comparison, and byte-by-byte examination using tools such as Binary Viewer.

Table. 5 : Encryption and Decryption Testing Results

No	Original Image File	Encrypted File Results of Affine Cipher, Playfair Cipher, and RSA	Decrypted File Results of Affine Cipher, Playfair Cipher, and RSA	Result
1				Valid
2				Valid
3				Valid

5. Conclusion

This study successfully implemented a super-encryption method by sequentially combining the Affine Cipher, Playfair Cipher, and RSA algorithms to enhance the security of image files. Each algorithm contributes uniquely to strengthening the encryption layers: the Affine Cipher performs the initial transformation of image bytes, the Playfair Cipher reinforces data structure through byte-pair substitution, and RSA provides security based on public-key cryptography. The desktop application developed using Visual Basic .NET is capable of executing the encryption and decryption processes with consistent results, as evidenced by the identical bytes between the original image file and its decrypted version. Furthermore, the file size remains unchanged, ensuring the integrity of the image structure. The test results demonstrate that this super-encryption approach significantly increases system complexity and reduces the likelihood of successful cryptographic analysis without complete knowledge of the entire sequence of algorithms and keys used. Therefore, this method can serve as an alternative solution for securing sensitive image files, such as identity document images and medical images.

References

[1] R. Prastya, A. M. H. Pardede, and A. Fauzi, "Teknik Pembangkit Kunci Algoritma RSA Menggunakan Algoritma Diffie Hellman pada Keamanan Citra," *KAKIFIKOM (Kumpulan Artik. Karya Ilm. Fak. Ilmu Komputer)*, vol. 04, no. 01, pp. 16–22, 2022, doi: 10.54367/kakifikom.v4i1.1872.

[2] C. Repi, J. Titaley, and E. Ketaren, "Implementasi Kriptografi Dalam Pengamanan Data Gambar Menggunakan Algoritma Rsa," *J. TIMES*, vol. 13, no. 1, pp. 93–99, 2024, doi: 10.51351/jtm.13.1.2024750.

[3] Imam Riadi, Abdul Fadlil, and Fahmi Auliya Tsani, "Pengamanan Citra Digital Berbasis Kriptografi Menggunakan Algoritma Vigenere Cipher," *JISKA (Jurnal Inform. Sunan Kalijaga)*, vol. 7, no. 1, pp. 33–45, 2022, doi: 10.14421/jiska.2022.7.1.33-45.

[4] H. Hendri, "Kompresi Citra dari Format BMP ke Format PNG," *J. TIMES*, vol. 3, no. 1, pp. 27–31, 2014, doi: 10.51351/jtm.3.1.201412.

[5] M. A. Abdussyukur, M. Toga, J. Sinaga, A. T. Zy, T. Informatika, and U. P. Bangsa, "ANALISIS METODE AFFINE CIPHER DENGAN KEYSTREAM ACAK UNTUK," vol. 9, no. 2, pp. 2231–2236, 2025.

[6] R. G. SINAMBELA and A. Fauzi, "Development of Hybrid Encryption Method Using Affine Cipher, Vigenere Cipher, and Elgamal Algorithm To Secure Text Messages in Data Communication System," *J. Artif. Intell. Eng. Appl.*, vol. 2, no. 2, pp. 30–40, 2023, doi: 10.59934/jaiea.v2i2.154.

[7] I. Artikel, "ANALISIS TEKNIK PLAYFAIR DAN SHIFT CIPHER SEBAGAI METODE KRIPTOGRAFI KLASIK UNTUK KEAMANAN DATA," pp. 13–19, 2025.

[8] A. Japardi, E. Louis Frasetyo, A. Sahertian, C. Umam, and G. A. Trisnapradika, "Playfair Cipher untuk Kriptografi ASCII menggunakan Matriks 9x10," *J. Komputasi dan Pengemb. Apl.*, vol. 1, no. 1, pp. 40–49, 2025.

[9] A. R. Mido and E. I. H. Ujianto, "Analisis Pengaruh Citra Terhadap Kombinasi Kriptografi RSA dan STEGANOGRAFi LSB," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 2, p. 279, 2022, doi: 10.25126/jtiik.2022914852.

[10] T. B. Surbakti, A. Fauzi, and H. Khair, "Rivest Shamir Adleman (RSA) Hybrid Algorithm System and the deep Blum Blum Shub (BBS) Algorithm Securing E-Absence Database Files," *Indones. J. Educ. Comput. Sci.*, vol. 1, no. 2, pp. 53–61, 2023, doi: 10.60076/indotech.v1i2.59.