Keamanan Informasi Citra Menggunakan RIJNDAEL dan AES 256 pada Steganografi

Deni Suprihadi[1], Rosa Nugraha[2], Andi Sutandi[3]
[1,2] Program Studi Teknik Informatika, Universitas Kebangsaan Republik Indonesia (UKRI), Bandung, Indonesia
[3] FKIP, Universitas Suryakancana, Kab. Cianjur, Indonesia
Email: [1]deni.suprihadi99@gmail.com, [2]rosasaepulnugraha@gmail.com, [3]andisutandi@gmail.com

Abstrak - Keamanan informasi digital menjadi tantangan utama seiring meningkatnya ancaman penyadapan dan pencurian data pada era komunikasi modern. Penelitian ini bertujuan meningkatkan kerahasiaan pesan dengan menggabungkan kriptografi Rijndael dan AES 256 serta steganografi Least Significant Bit (LSB) yang dipadukan dengan algoritma Linear Congruential Generator (LCG) untuk pengacakan posisi bit. Metode yang digunakan adalah eksperimen berbasis pengembangan perangkat lunak dengan model Waterfall, meliputi tahap analisis kebutuhan, perancangan sistem menggunakan UML, implementasi, dan pengujian kualitas citra. Sistem yang dikembangkan mampu mengenkripsi pesan, mendistribusikan bit secara acak, serta menyisipkan data dalam citra digital. Hasil pengujian menunjukkan nilai PSNR di atas 40 dB dan MSE yang rendah, sehingga kualitas citra stego tetap baik tanpa distorsi visual. Kesimpulannya, kombinasi teknik kriptografi dan steganografi ini terbukti efektif meningkatkan keamanan pesan rahasia dalam citra digital tanpa menurunkan kualitas visual, serta layak diterapkan pada komunikasi rahasia berskala luas.

Kata kunci: Kriptografi, Rijndael, *Advanced Encryption Standard 256*, Steganografi, Keamanan Informasi.

Abstract - Digital information security has become a major challenge due to the increasing threats of eavesdropping and data theft in modern communication. This research aims to enhance message confidentiality by combining Rijndael and AES-256 cryptography with Least Significant Bit (LSB) steganography, supported by the Linear Congruential Generator (LCG) algorithm to randomize bit positions. The study employs an experimental software development approach using the Waterfall model, consisting of requirements analysis, system design with UML, implementation, and image quality testing. The developed system can encrypt messages, randomly distribute the bits, and embed data within digital images. Experimental results show PSNR values above 40 dB and low MSE, ensuring the stego image maintains good visual quality without noticeable distortion. In conclusion, this combination of cryptography and steganography effectively improves the security of secret messages in digital images while preserving visual quality, making it suitable for secure large-scale communication.

***Keywords*: *Cryptography*, Rijndael, *Advanced Encryption Standard 256*, *Steganography*, *Information Security*.**

I. PENDAHULUAN

Keamanan informasi menjadi isu penting dalam era digital saat ini. Ancaman terhadap data sensitif seperti penyadapan pesan dan pencurian data meningkat seiring dengan perkembangan teknologi komunikasi. Kriptografi ilmu yang mempelajari cara mengubah informasi ke dalam bentuk yang tidak dapat dipahami oleh siapa pun kecuali penerima yang dituju. Proses ini melibatkan pengubahan plaintext menjadi ciphertext menggunakan algoritma enkripsi dan sebuah kunci [1]. Steganografi adalah seni dan ilmu untuk menyembunyikan informasi dalam sebuah media penutup sedemikian rupa sehingga keberadaan data yang disembunyikan tidak terdeteksi. Berbeda dengan kriptografi yang menyamarkan isi pesan, steganografi menyembunyikan keberadaan pesan itu sendiri [2]

LSB merupkan teknik yang paling umum dan paling sederhana digunakan dalam steganografi citra, karena menyebabkan distorsi visual yang minimal [1]. Namun, pesan yang disisipkan dalam posisi tetap dapat menimbulkan pola mencurigakan. Untuk mengatasi hal ini, Steganografi LSB menyembunyikan data rahasia pada bit paling tidak signifikan dari nilai piksel, sementara enkripsi AES menambahkan lapisan keamanan kedua, sehingga tetap menjaga kerahasiaan meskipun citra dianalisis [3]. Teknik LSB mengganti bit terakhir dari citra digital dengan bit pesan rahasia sehingga perbedaan visual hampir tidak terlihat oleh mata manusia [4]. Penelitian ini mengintegrasikan kriptografi Rijndael dan AES 256 untuk mengenkripsi pesan sebelum disisipkan dalam citra digital menggunakan teknik steganografi LSB[5]. Metode ini bertujuan untuk meningkatkan keamanan pesan tanpa mengurangi kualitas visual citra. [6]

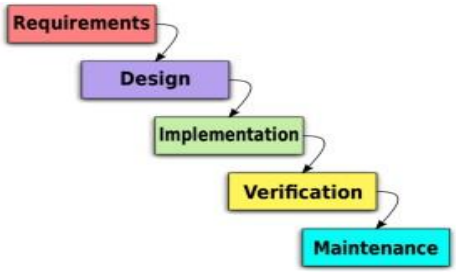
Beberapa penelitian sebelumnya telah membahas implementasi system keamanan pesan digital algoritma steganografi dan RSA [7] modifikasi LSB pada gambar hiding steganografi tanpa menggunakan enkripsi Rijndael. [8] menggunakan Caesar Cipher dan LSB dalam mengekripsi file penting bukan berupa gambar [9] . melindungi data dan informasi menggunakan metode PVD [10] mengembangkan steganografi LSB dan LCG untuk menghasilkan jumlah acak posisi pixel.[11]

Penelitian ini menggunakan pendekatan Waterfall dalam pengembangan sistem, yang mencakup tahap analisis kebutuhan, perancangan sistem, implementasi, pengujian, [12]. Selain itu, pendekatan ini juga dapat menjadi model bagi implementasi dalam kasus penyadapan data dalam bentuk pesan menunjukkan bahwa keamanan informasi menjadi tantangan dalam era digital saat ini yang

mengakibatkan kerugian bagi individu dan organisasi yang terkena dampak. Implementasi kombinasi kriptografi Rijndael, Kriptografi AES 256 [13], Pseudo Random Number Generator dan steganografi Least Significant Bit pada citra untuk menghasilkan citra stego karena dengan menggabungkan pesan dengan nilai-nilai acak yang dihasilkan oleh PRNG, pola ini dapat dihindari sehingga menjadikan pesan yang tersembunyi lebih sulit dideteksi dan menganalisis bagaimana perbandingan dan perbedaan kualitas citra stego dengan citra penampung jika dilihat dari MSE dan PSNR.

II. METODE PENELITIAN

Penelitian ini menggunakan metode eksperimen dengan pendekatan pengembangan perangkat lunak berbasis Software Development Life Cycle (SDLC) model waterfall [12]. Metode eksperimen dipilih karena penelitian ini bertujuan untuk menguji dan mengevaluasi implementasi sistem pengamanan data menggunakan kriptografi, algoritma pengacakan posisi, dan steganografi pada citra digital.



Gambar 1. SDLC model waterfall

Model tersebut melibatkan pembuatan prototipe sistem yang mampu mengenkripsi pesan, mengacak posisi bit dalam citra, serta menyisipkan data menggunakan steganografi LSB. Sistem diuji secara menyeluruh untuk memastikan keamanannya melalui pengukuran kualitas citra menggunakan metrik MSE dan PSNR. Model Waterfall pada gambar.1 dilaksanakan dalam 2 bagian yaitu analisa perancangan dan implementasi.

A. Analisa Kebutuhan Sistem

Analisa Kebutuhan sistem yang dikembangkan dalam penelitian ini bertujuan untuk mengintegrasikan kriptografi, pengacakan posisi bit, steganografi dan pengukuran kualitas citra dalam sebuah aplikasi desktop. Berikut adalah rincian setiap komponen utama:

a. Modul Enkripsi

Untuk Modul Enkripsi menggunakan algoritma Rijndael dan AES 256 untuk mengenkripsi pesan teks yang akan disisipkan. Rijndael: Algoritma ini dipilih karena kemampuannya untuk menangani ukuran blok fleksibel dan tingkat keamanan tinggi. Pada penelitian ini, panjang kunci 128- bit digunakan. AES 256: Merupakan versi Rijndael dengan panjang kunci tetap sebesar 256-bit, memberikan keamanan tambahan terhadap serangan brute force. Proses enkripsi meliputi substitusi byte, pergeseran baris (shift rows), pencampuran kolom (mix columns), dan penambahan kunci putaran (add round key).

b. Modul Pengacakan Posisi Bit

LCG adalah salah satu generator bilangan acak semu tertua dan paling sederhana. Algoritma ini menggunakan relasi rekursif [1]. LCG memastikan bahwa posisi penyisipan pesan tersebar secara acak, sehingga sulit ditebak. Persamaan LCG:

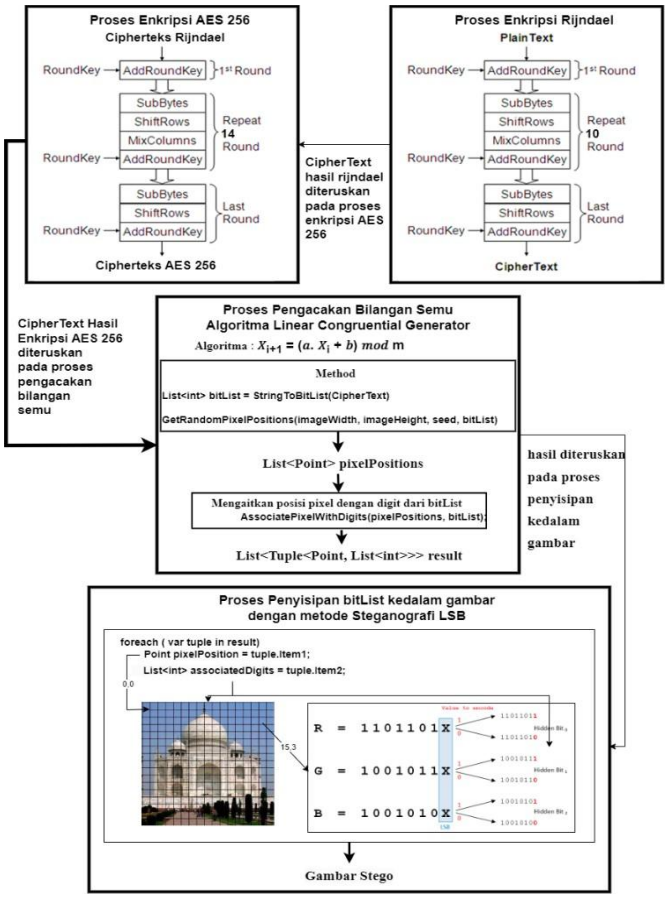
$$X_{n+1} = (aX_n + c) \bmod m \quad (1)$$

Di mana:

X_{n+1} : Nilai bilangan acak berikutnya
X_n : Nilai bilangan acak saat ini
a, c, m: Konstanta untuk memastikan distribusi acak yang baik

c. Modul Steganografi LSB

Teknik LSB menyisipkan bit pesan terenkripsi ke dalam bit paling tidak signifikan dari piksel citra penampung. Penyisipan dilakukan pada komponen RGB dari piksel citra, sehingga menghasilkan citra stego yang tampak identik dengan citra asli bagi mata manusia. Proses ini didukung oleh pengacakan posisi bit, sehingga pola yang mencurigakan dapat dihindari.



Gambar 2. Arsitektur Sistem Enkripsi Pesan

Gambar 2. Menunjukkan arsitektur sistem secara keseluruhan, mencakup alur enkripsi pesan, pengacakan posisi, hingga penyisipan pesan dalam citra

d. Pengukuran Kualitas Citra

Pengukuran kualitas citra dilakukan untuk memastikan bahwa citra stego tidak mengalami penurunan kualitas yang signifikan setelah proses penyisipan pesan. Dua metrik utama yang digunakan adalah MSE dan PSNR.

1) Mean Squared Error (MSE)

MSE digunakan untuk mengukur perbedaan rata-rata antara citra asli dan citra stego. Nilai MSE yang kecil menunjukkan bahwa perbedaan antara citra asli dan citra stego sangat kecil.

$$MSE = (1 / M.N). \Sigma (y=1 \text{ sampai } M). \Sigma (x=1 \text{ sampai } N). [l (x, y) - l'(x, y)] ^2 \quad (2)$$

Di mana:
M = Panjang cari citra (ukuran piksel)
N = Lebar dari citra (ukuran piksel)
(x, y) = koordinat masing-masing piksel
l = nilai bit pada citra asli
l' = nilai bit pada citra stego

2) *Peak Signal-to-Noise Ratio* (PSNR)

PSNR, atau Perbandingan Sinyal ke Derau, digunakan untuk menghitung perbandingan antara nilai maksimum sinyal yang diukur dengan jumlah derau yang berdampak pada sinyal. Pengukuran PSNR, yang dihitung dengan satuan decibel (db), memungkinkan untuk mengetahui perbandingan kualitas citra sebelum dan sesudah disisipi pesan. Hasil percobaan menunjukkan bahwa algoritma yang diusulkan secara efektif melindungi data dengan nilai PSNR yang tinggi dan MSE yang rendah [14]

$$PSNR = 10 \log_{10} (255^2 / MSE) \quad (3)$$

Dimana :
255 adalah nilai intensitas maksimum pada piksel (untuk citra 8-bit)

3) Interpretasi Nilai PSNR dan MSE

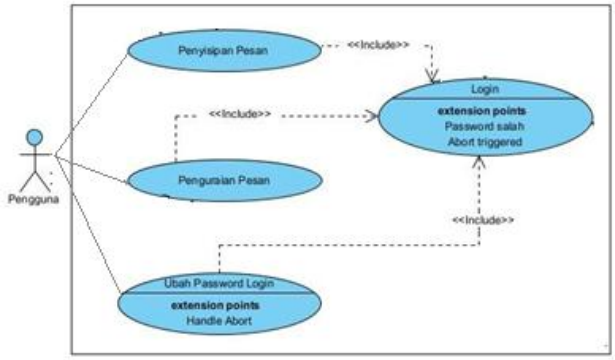
PSNR > 40 dB: Kualitas citra sangat baik, perbedaan visual tidak dapat dideteksi oleh mata manusia. MSE mendekati nol: Menunjukkan bahwa perubahan piksel sangat kecil, hampir tidak ada perbedaan antara citra asli dan citra stego.

B. Perancangan Sistem

Penggunaan *Unified Modelling Language* (UML) bertujuan untuk memodelkan proses bisnis, arsitektur system dan alat untuk analisis, desain dan implementasi [15]. UML bahasa pemodelan visual serbaguna yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan artefak dari suatu sistem perangkat lunak[16] UML bahasa pemodelan grafis yang digunakan untuk merancang dan mendokumentasikan sistem perangkat lunak. UML sering digunakan oleh para pengembang perangkat lunak untuk memvisualisasikan, mendokumentasikan, dan memahami rancangan sistem perangkat lunak yang akan dibangun [17]

1) Use Case Diagram

Untuk memberikan Gambaran fungsi-fungsi utama dari system *Use Case Diagram* menjelaskan bagaimana pengguna yang akan menyisipkan pesan dalam sistem dan melakukan penyisipan. mengurai pesan dan pengaturan akses, seperti terlihat pada Gambar 3.

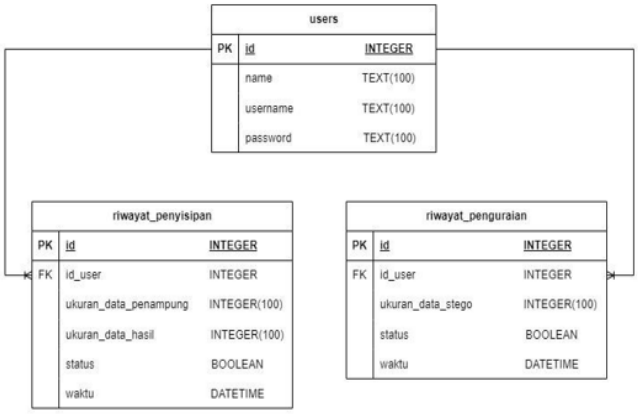


Gambar 3. Diagram Interaksi Sistem

Penggunna harus melakukan login sebelum masuk kedalam sistem utama untuk melakukan aktivitas penyisipan pesan kedalam gambar, sebaliknya jika ingin mengetahui pesan didalam gambar, maka perlu melakukan proses penguraian pesan didalam gambar.

2) Relationship Diagram

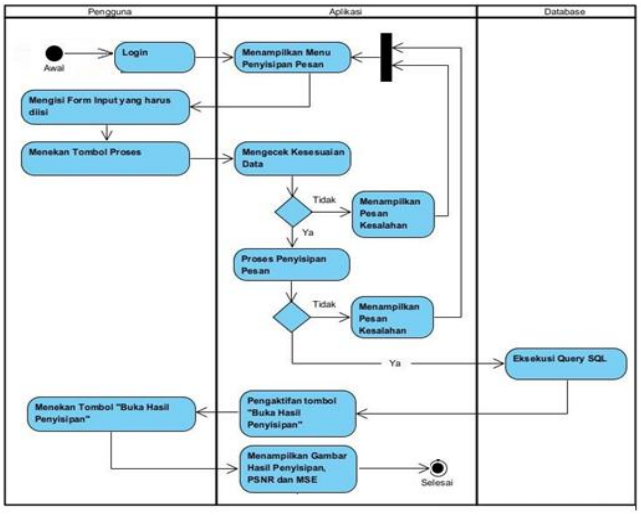
Agar Sistem Keamanan Informasi pada pesan ini berjalan dengan baik, maka dibutuhkan Tabel Relationship untuk memetakan data pengguna, data penyisipan pesan dan data penguraian pesan sehinga saling terhubung. Diagram ini menunjukan data pengguna yang memiliki akses masuk kedalam sistem, selanjutnya melakukan proses dan penyimpanan data penyisipan kedalama table Riwayat penyisipan, begitu pula dalam proses penguraian pesan akan disimpan pada table riwayat penguraian. Visualisasi tersebut seperti pada Gambar 4.



Gambar 4. Diagram Hubungan antar Tabel Data

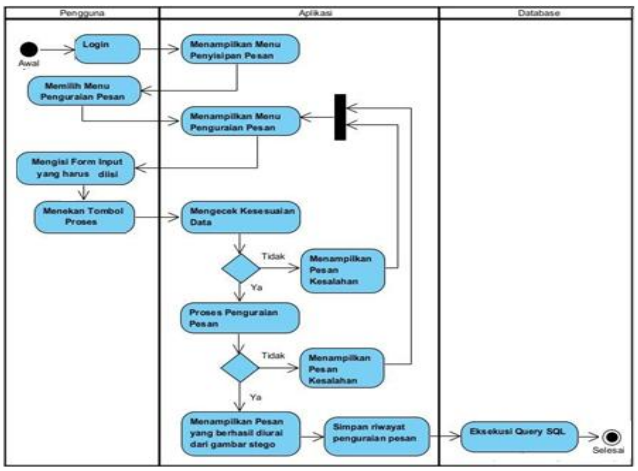
3) Diagram Activity

Bagian ini memberikan Gambaran alur aktivitas sekuensial dari setiap proses bisnis dari *Use Case* Diagram, sehingga mendukung fungsionalitas sistem, mulai dari pengguna melakukan login, penyisipan dan penguraian pesan pada gambar, seperti tervisualisasikan pada gambar 5 dan gambar 6.



Gambar 5. Aktivitas Penyisipan Pesan

Pengguna memulai login dengan memasukan user dan password, jika sudah sesuai maka pengguna berada dalam halaman utama, kemudian memilih menu penyisipan pesan dan memulai langkah-langkah penyisipan pesan pada gambar. Setelah ketentuan gambar terpenuhi dan teks yang akan disisipkan sebagai pesan sudah diinput, maka proses penyisipan dapat dieksekusi. Hasil penyisipan di simpan pada table riwayat penyisipan, untuk kebutuhan history.



Gambar 6. Aktivitas Pengurai Pesan

Pengguna memulai login dengan memasukan user dan password, jika sudah sesuai maka pengguna berada dalam halaman utama, kemudian memilih menu pengurai pesan dan memulai langkah-langkah pengurai pesan pada gambar. Setelah ketentuan penguraian terpenuhi, maka proses penguraian pesan dapat dieksekusi, sehinnga hasil penguraian menghasilkan teks yang disisipkan pada pesan gambar, kemudia di simpan pada table riwayat penguraian pesan, untuk kebutuhan history.

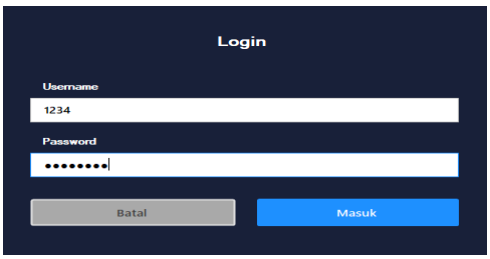
III. HASIL DAN PEMBAHASAN

Setelah melakukan Analisa dan perancangan yang mengacu pada model waterfall, maka selanjutnya berfokus pada Implementasi dan Pengujian (Verifikasi).

A. Implementasi Sistem

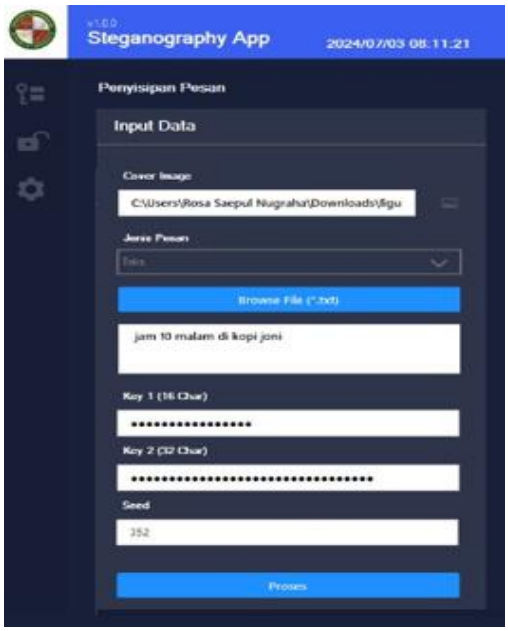
1) Antar Muka

Bagian antar muka login digunakan pengguna pada saaat akan masuk kedalam sistem keamana informasi pesan, hal ini diperlukan untuk memastikan hak akses pengguna secara tepat melalui pemeriksaan user dan password, seperti terlihat pada gambar 7.



Gambar 7. Antar Muka Muka Login

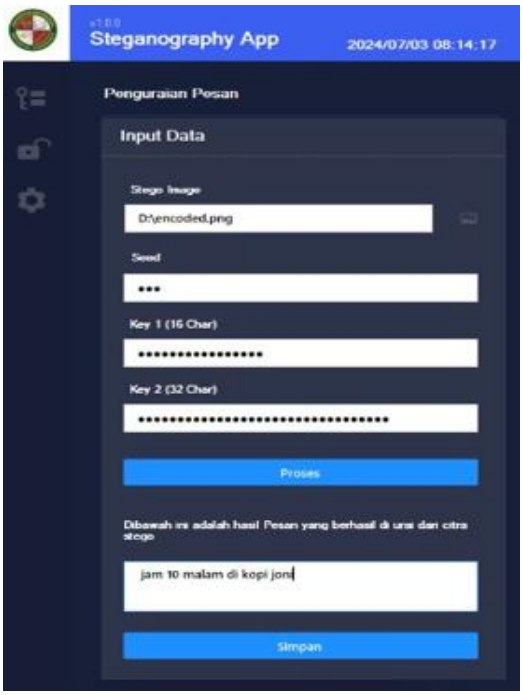
Bagian antar muka penyisipan pesan digunakan pengguna pada saaat akan memasukan teks pada pesan gambar ke dalam sistem keamana informasi pesan.



Gambar 8. Antar Muka Penyisipan Teks pada Pesan Gambar

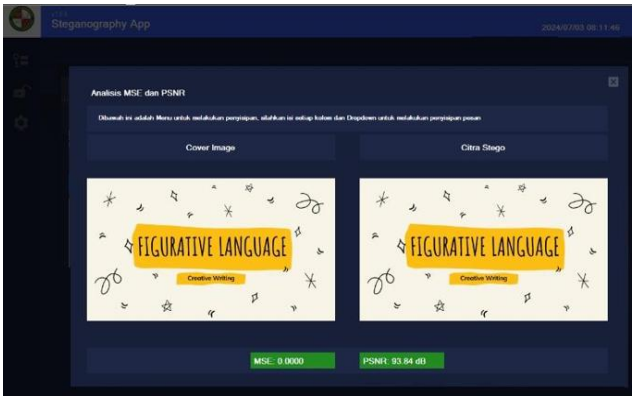
Pada gambar 8, pengguna dapat memilih gambar cover, tipe pesan apakah file .txt atau inputan pesan yang dapat di input pada kolom input pesan rahasia. Selain itu yang harus diisi adalah Key 1 yang memiliki Panjang 16 char dan Key 2 yang memiliki Panjang 32 char. Setelah terisi maka untuk melakukan penyisipan, pengguna dapat menekan tombol “proses”, setelah itu maka akan muncul

"seed" dan popup pesan apakah akan menyimpan gambar yang telah disisipi pesan rahasia.



Gambar 9. Antar Muka Pengurai Teks pada Pesan Gambar

Pada gambar 9, penguraian pesan ini, pengguna dapat melakukan penguraian pesan rahasia dari citra yang telah disisipi pesan rahasia. Pada menu ini pengguna dapat memilih citra stego, Key 1 yang memiliki Panjang 16 char dan Key 2 yang memiliki Panjang 32 char dan seed. Setelah terisi maka untuk melakukan penguraian, pengguna dapat menekan tombol "proses", setelah itu maka akan muncul pesan rahasia pada kolom pesan



Gambar 10. Antar Muka Analisis MSE dan PSNR

Pada Gambar 10, menjelaskan antar muka yang berfungsi sebagai analisis MSE dan PSNR, pengguna dapat melihat hasil dari gambar cover yang telah disisipi pesan dan gambar gambar sebelum dan sesudah disisipi pesan. Nilai PSNR diukur dalam desibel (dB), dan PSNR berbanding terbalik dengan MSE. Nilai PSNR yang besar menunjukkan nilai MSE yang kecil, sedangkan nilai MSE yang kecil menunjukkan perbedaan yang besar (degradasi) antara dua gambar. PSNR yang dapat diterima atau ditoleransi adalah jika lebih dari 30.

2) Pengkodean

Setelah melalui tahap perancangan, maka kriptografi Rijndael di implementasi dalam bentuk Coding sebagai berikut :

```
public byte[] Encrypt(byte[] plaintext, byte[] key)
{
    key = PadKey(key);
    KeyExpansion(key);

    int paddedLength = ((plaintext.Length + 15) / 16) * 16;
    byte[] paddedPlaintext = new byte[paddedLength];
    Array.Copy(plaintext, paddedPlaintext, plaintext.Length);

    for (int i = plaintext.Length; i < paddedLength; i++)
    {
        paddedPlaintext[i] = (byte)(paddedLength - plaintext.Length);
    }

    byte[] ciphertext = new byte[paddedLength];

    for (int block = 0; block < paddedPlaintext.Length / 16; block++)
    {
        byte[] blockPlaintext = new byte[16];
        Array.Copy(paddedPlaintext, block * 16, blockPlaintext, 0, 16);

        byte[] blockCiphertext = EncryptBlock(blockPlaintext);

        Array.Copy(blockCiphertext, 0, ciphertext, block * 16, 16);
    }

    return ciphertext;
}
```

Gambar 11. Kode Enkripsi Penyisipan Pesan

```
public byte[] Decrypt(byte[] ciphertext, byte[] key)
{
    key = PadKey(key);
    KeyExpansion(key);

    byte[] decrypted = new byte[ciphertext.Length];

    for (int block = 0; block < ciphertext.Length / 16; block++)
    {
        byte[] blockCiphertext = new byte[16];
        Array.Copy(ciphertext, block * 16, blockCiphertext, 0, 16);

        byte[] blockDecrypted = DecryptBlock(blockCiphertext);

        Array.Copy(blockDecrypted, 0, decrypted, block * 16, 16);
    }

    int paddingLength = decrypted[decrypted.Length - 1];
    byte[] unpaddedDecrypted = new byte[decrypted.Length - paddingLength];
    Array.Copy(decrypted, unpaddedDecrypted, decrypted.Length - paddingLength);

    return unpaddedDecrypted;
}
```

Gambar 12. Kode Dekripsi Penyisipan Pesan

Untuk menangani proses enkripsi dan dekripsi AES 256 dibuat Class, sebagai berikut.

```
public class AES256
{

    private static readonly byte[] SBox = new byte[256]
    {
    0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30, 0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76,
    0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,
    0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,
    0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,
    0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,
    0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,
    0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,
    0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,
    0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,
    0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,
    0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79,
    0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE, 0x08,
    0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,
    0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E,
    0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28, 0xDF,
    0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41, 0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB, 0x16
    };

    private static readonly byte[] InvSBox = new byte[256]
    {
    0x52, 0x09, 0x6A, 0xD5, 0x30, 0x36, 0xA5, 0x38, 0xBF, 0x40, 0xA3, 0x9E, 0x81, 0xF3, 0xD7, 0xFB,
    0x7C, 0xE3, 0x39, 0x82, 0x9B, 0x2F, 0xFF, 0x87, 0x34, 0x8E, 0x43, 0x44, 0xC4, 0xDE, 0xE9, 0xCB,
    0x54, 0x7B, 0x94, 0x32, 0xA6, 0xC2, 0x23, 0x3D, 0xEE, 0x4C, 0x95, 0x0B, 0x42, 0xFA, 0xC3, 0x4E,
    0x08, 0x2E, 0xA1, 0x66, 0x28, 0xD9, 0x24, 0xB2, 0x76, 0x5B, 0xA2, 0x49, 0x6D, 0x8B, 0xD1, 0x25,
    0x72, 0xF8, 0xF6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xD4, 0xA4, 0x5C, 0xCC, 0x5D, 0x65, 0xB6, 0x92,
    0x6C, 0x70, 0x48, 0x50, 0xFD, 0xED, 0xB9, 0xDA, 0x5E, 0x15, 0x46, 0x57, 0xA7, 0x8D, 0x9D, 0x84,
    0x90, 0xD8, 0xAB, 0x00, 0x8C, 0xBC, 0xD3, 0x0A, 0xF7, 0xE4, 0x58, 0x05, 0xB8, 0xB3, 0x45, 0x06,
    0xD0, 0x2C, 0x1E, 0x8F, 0xCA, 0x3F, 0x0F, 0x02, 0xC1, 0xAF, 0xBD, 0x03, 0x01, 0x13, 0x8A, 0x6B,
    0x3A, 0x91, 0x11, 0x41, 0x4F, 0x67, 0xDC, 0xEA, 0x97, 0xF2, 0xCF, 0xCE, 0xF0, 0xB4, 0xE6, 0x73,
    0x96, 0xAC, 0x74, 0x22, 0xE7, 0xAD, 0x35, 0x85, 0xE2, 0xF9, 0x37, 0xE8, 0x1C, 0x75, 0xDF, 0x6E,
    0x47, 0xF1, 0x1A, 0x71, 0x1D, 0x29, 0xC5, 0x89, 0x6F, 0xB7, 0x62, 0x0E, 0xAA, 0x18, 0xBE, 0x1B,
    0xFC, 0x56, 0x3E, 0x4B, 0xC6, 0xD2, 0x79, 0x20, 0x9A, 0xDB, 0xC0, 0xFE, 0x78, 0xCD, 0x5A, 0xF4,
    0x1F, 0xDD, 0xA8, 0x33, 0x88, 0x07, 0xC7, 0x31, 0xB1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xEC, 0x5F,
    0x60, 0x51, 0x7F, 0xA9, 0x19, 0xB5, 0x4A, 0x0D, 0x2D, 0xE5, 0x7A, 0x9F, 0x93, 0xC9, 0x9C, 0xEF,
    0xA0, 0xE0, 0x3B, 0x4D, 0xAE, 0x2A, 0xF5, 0xB0, 0xC8, 0xEB, 0xBB, 0x3C, 0x83, 0x53, 0x99, 0x61,
    0x17, 0x2B, 0x04, 0x7E, 0xBA, 0x77, 0xD6, 0x26, 0xE1, 0x69, 0x14, 0x63, 0x55, 0x21, 0x0C, 0x7D
    };
```

Gambar 13. Kode Class AES 256

B. Pengujian Sistem

Hasil implementasi sistem pengamanan informasi pada citra digital menunjukkan bahwa kombinasi kriptografi Rijndael, AES 256, algoritma LCG, dan steganografi LSB dapat meningkatkan keamanan data tanpa mengurangi kualitas visual citra stego. Berikut adalah hasil pengujian:

1) Pengujian Kode

Pengujian sistem keamanan informasi pesan ini dilakukan dengan langkah seperti tabel berikut:

Tabel 1. Langkah Pengujian Kode

No	*Skenario*	*Hasil yang diharapkan*	*Hasil Pengujian*
1	Melakukan konversi pesan ke bentuk hexadecimal	Hasil berupa hexadecimal	Berhasil
2	Melakukan Enkripsi pesan rahasia dengan algoritma Rijndael 128	Enkripsi pesan dengan algoritma Rijndael 128	Berhasil
3	Encoding cipherteks Rijndael ke base64	Ubah ke bentuk base64	Berhasil
4	Melakukan Enkripsi cipherteks hasil enkripsi algoritma Rijndael dengan Algoritma AES 256.	Enkripsi algoritma AES 256	Berhasil
5	Encoding cipherteks AES 256 ke base64	Ubah ke bentuk base64	Berhasil
6	Fungsi Pengacakan Posisi Pixel dengan Pseudo Random Number Generator	Mendapatkan list posisi random	Berhasil
7	Asosiasi pixel dan 3 digit bit dengan method AssociatePixelWithDigits	Mengasosiasi posisi pixel dan 3 digit list bit	Berhasil
8	Penyisipan Steganografi LSB	Penyisipan pesan kedalam citra	Berhasil
9	Fungsi Pengambilan posisi pixel dengan fungsi GetRandomPixelPositionsDecode	Pengambilan posisi pixel	Berhasil
10	Mengonversi setiap grup 8 bit menjadi byte dan kemudian menjadi karakter base64 dengan method BitListToString	Ubah ke base64	Berhasil
11	Decrypt AES 256	Decrypt AES 256	Berhasil
12	Decrypt Rijndael 128	Decrypt Rijndael 128	Berhasil

2) Keamanan Informasi

Metode pengacakan posisi bit menggunakan algoritma LCG berhasil mendistribusikan bit pesan secara acak ke seluruh piksel citra. Hal ini membuat pola bit sulit ditebak, sehingga meningkatkan keamanan data terhadap analisis visual dan serangan statistik.

Kombinasi dengan algoritma AES 256 memberikan perlindungan tambahan karena pesan telah dienkripsi sebelum disisipkan ke dalam citra. Dengan demikian, meskipun posisi bit ditemukan, pesan tidak dapat didekripsi tanpa kunci.

Tabel 2. Citra Hasil Pengujian Kode

No	*Jenis Citra*	*Gambar*
1	Citra Asli	
2	Cintra Hasil Penyisipan	

Tabel 2 menunjukkan citra asli dan citra stego untuk sampel pertama, yang menunjukkan kualitas visual yang sangat mirip.

3) Kualitas Citra Stego

Pengujian kualitas citra menggunakan parameter MSE dan PSNR menunjukkan bahwa perubahan pada citra akibat penyisipan pesan sangat kecil dan tidak terdeteksi oleh mata manusia

Tabel 3. Hasil Pengujian Kualitas Citra

NO	*Citra*	*PSNR(db)*	*MSE*

1		69.6205 dB	0.0070962963
2		80.1756 dB	0.0006244792
3		78.4933 dB	0.0009199211

Nilai PSNR > 40 dB menunjukkan bahwa kualitas visual citra stego sangat baik, sementara nilai MSE yang rendah menunjukkan perbedaan rata-rata kuadrat antara citra asli dan citra stego hampir tidak ada.

4) Efisiensi Sistem

Implementasi sistem menunjukkan waktu eksekusi yang cepat untuk proses enkripsi, pengacakan posisi, dan penyisipan data. Hal ini membuat sistem cocok untuk penggunaan pada skala besar atau kebutuhan real-time.

5) Perbandingan dengan metode Sebelumnya

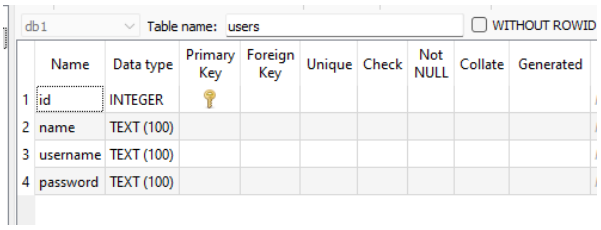
Dibandingkan dengan metode steganografi LSB tradisional tanpa pengacakan, metode ini menunjukkan peningkatan yang signifikan dalam keamanan. Teknik kriptografi AES 256 yang diterapkan juga membuat pesan hampir tidak mungkin didekripsi tanpa kunci enkripsi.

Hasil penelitian membuktikan bahwa kombinasi teknik yang digunakan tidak hanya meningkatkan keamanan data tetapi juga mempertahankan kualitas visual citra stego. Sistem ini ideal untuk aplikasi komunikasi rahasia berbasis citra digital.

C. Basis Data

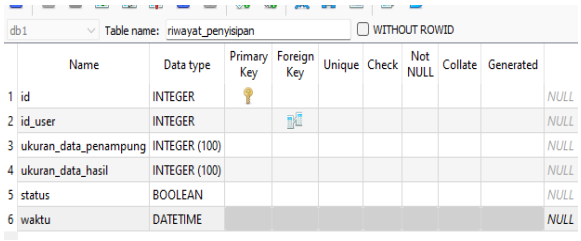
Basis data yang digunakan untuk Keamanan Informasi Citra pada sistem ini dibuat sebagai berikut:

1) File users berfungsi untuk menyimpan data pengguna yang digunakan untuk memasuki aplikasi, tabel users mempunyai field id, name, username dan password dengan primary key id, seperti terlihat pada gambar 14.



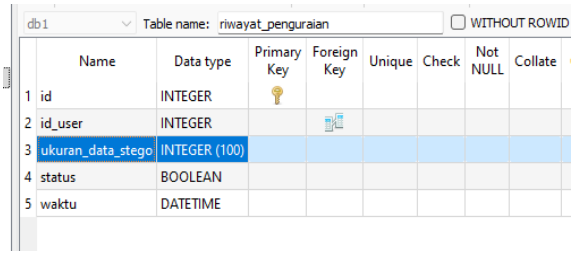
Gambar 14. Struktur File User

2) File Riwayat_Riwayat_penyisipan berfungsi untuk menyimpan data riwayat penyisipan pesan rahasia, tabel Riwayat_penyisipan mempunyai field id, id_user, ukuran_data_penampung, ukuran_data_hasil, status dan waktu dengan primary key id, seperti terlihat pada gambar 15.



Gambar 15. Struktur File Data Penyispan Pesan

3) File Riwayat_ penguraian berfungsi untuk menyimpan data riwayat penguraian pesan rahasia, tabel Riwayat_penguraian mempunyai field id, id_user ukuran_data_stego, status dan waktu dengan primary key id, seperti terlihat pada gambar 16.



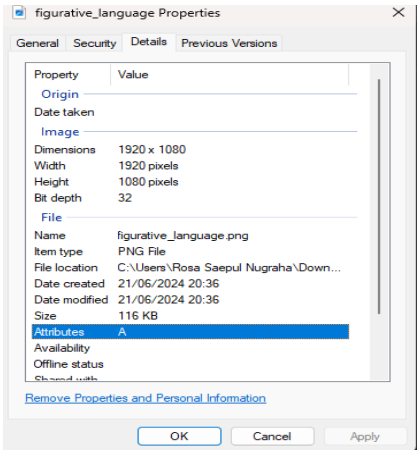
Gambar 16. Struktur File Data Pengurai Pesan

D. Algoritma Penyisipan Pesan

Proses Penyisipan Pesan pada Citra dilakukan berdasarkan Arsitektur Sistem Enkripsi Pesan seperti pada gambar 2 yang memiliki beberapa tahap. Dimulai dengan persiapan sampel gambar yang akan disisipkan pesan dan teks berisi pesan. Adapun langkahnya sebagai berikut :

1) Persiapan gambar.

Sampel citra penampung berukuran 116 KB dengan format PNG dan lebar pixel sebesar 1920 x 1080, seperti terlihat pada gambar 17.



Gambar 17. Struktur File Gambar

2) Sampel Pesan

Sample pesan rahasia yang akan disisipkan, sebagai contoh "**Meet kopi johny 19-06-2024 19**", dengan menggunakan 2 kunci rahasia, terdiri dari Key Pertama "**19kJmnkjiu029kmV**" dan Key Kedua "**8dJk2uhdJyLPqCnmkjsq83Jhf0982jNj**".

3) Konversi Hexadecimal

Setelah sample citra dan teks pesan sudah tersedia, selanjutnya melakukan konversi pesan ke bentuk hexadecimal (encoding ASCII), sehingga hasilnya sebagai berikut :

Tabel 4. Encoding Pesan

ASCII	*HEXADECIMAL*
Meet kopi john y 19-06-2024 19	4D 65 65 74 20 6B 6F 70 69 20 6A 6F 68 6E 79 20 31 39 2D 30 36 2D 32 30 32 34 20 31 39

Tabel 5. Encoding Key-1

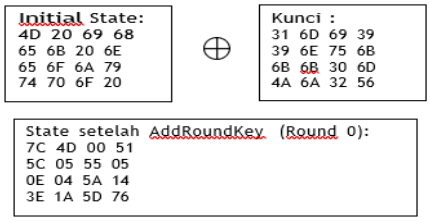
ASCII	*HEXADECIMAL*
19kJmnkjiu029kmV	31 39 6B 4A 6D 6E 6B 6A 69 75 30 32 39 6B 6D 56

Tabel 6. Encoding Key-2

ASCII	*HEXADECIMAL*
8dJk2uhdJyLPqCnmkjsq83Jhf0982jNj	38 64 4A 6B 32 75 68 64 4A 79 4C 50 71 43 6E 6D 6B 6A 73 71 38 33 4A 68 66 30 39 38 32 6A 4E 6A

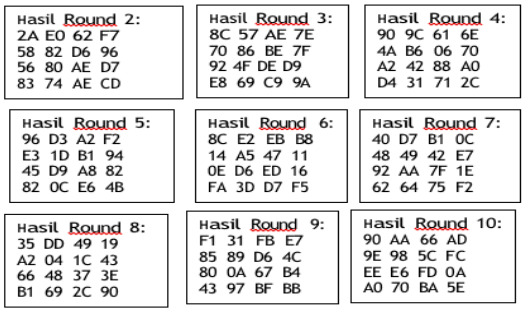
4) Enkripsi dengan algoritma Rijndael 128

Hexadecimal pesan, selanjutnya akan dihitung dengan menggunakan Hexadecimal yang dimiliki Key-1 dan Key-2. Tahap untuk key-1 dimulai dengan menghitung Initial Round menggunakan XOR plaintext dan kunci-1, seperti gambar berikut :



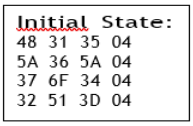
Gambar 18. Initial Round

Setelah Intial Round, Selanjutnya dilakukan beberapa proses putaran yang didalamnya terdapat perhitungan SubBytes, ShiftRows, MixColumns dan AddRoundKey, sehingga pada tahap Round ke Sepuluh selesai menghasilkan perhitungan sebagai berikut :



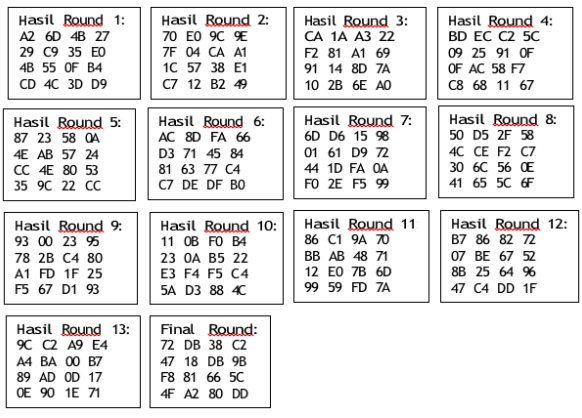
Gambar 19. Hasil Perhitungan Round sampai 9x

Hasil perhitungan Round-10 pada gambar 10 dipergunakan untuk keperluan perhitungan pada Key-2. Dengan cara perhitungan yang sama hanya kunci yang digunakan berbeda yaitu menggunakan Key-2. Untuk Initial Roud dengan Key-2 menghasilkan Initial State, seperti berikut :



Gambar 20. Intial State Key-2

Adapun Hasil perhitungan Round 1 sampai 14 dengan perhitungan SubBytes, ShiftRows, MixColumns dan AddRoundKey seperti di tahap sebelumnya, tervisualisasikan seperti berikut :



Gambar 21. Hasil Perhitungan Round Key-2

5) Penggabungan Blok

Dari hasil putaran masing-masing block kemudian dilakukan penyatuan kembali supaya menghasilkan sebuah chippert text berbentuk hexadecimal yang utuh, seperi berikut :

59 D5 DC 08 18 A0 17 B5 63 08 4B 8A CB 12 7C BD 11 9D 56 BD 1A 5E C2 08 7E D0 29 3A 31 67 ED 6B 72 47 F8 4F DB 18 81 A2 38 DB 66 80 C2 9B 5C DD

Setelah proses penggabungan blok selesai, maka dilanjutkan dengan encoding base64 pada hexadecimal pada cipertext tersebut.

Tabel 7. Encoding base64 cipertext

Plaintext	*Hexadecimal*	*Base 64*
kJ7uoKqY5nBmX P26rfwKXjW/Zxf BndrOH Z7216oQ5Z4=	59 D5 DC 08 18 A0 17 B5 63 08 4B 8A CB 12 7C BD 11 9D 56 BD 1A 5E C2 08 7E D0 29 3A 31 67 ED 6B 72 47 F8 4F DB 18 81 A2 38 DB 66 80 C2 9B 5C DD	WdXcCBigF 7VjCEuKyxJ 8vRGdVr0aX sIIftApOjFn7 WtyR/hP2xiB ojjbZoDCm1 zd

E. Fungsi Pengacakan Posisi Pixel PRNG

Setelah diperoleh cipertext dengan base64, maka pada tahap akhir dilakukan penyisipan kedalam citra.

1) Data cipertext

Data cipertext Base 64 dari hasil AES 256 :
WdXcCBigF7VjCEuKyxJ8vRGdV
r0aXsIIftApOjFn7WtyR/hP2xiBojjbZoDCm1zd

Tinggi citra penampung : **1080**
Lebar citra penampung : **1920**
Seed : **512**
Konversi kedalam Bit List :

0010101101010100011101000110101100110111010110100
0111001011
0001001101011010000010100010101100111001110000111
1000011010
0000110111010001110110100101100011011011010100011
1010001000

2) Method Posisi dan Asosiasi Pixel

Untuk menghasilkan pengambilan posisi pixel acak menggunakan method GetRandomflixelflositions, sedangkan untuk menghasilkan asosiasi pixel menggunakan method AssociateflixelWithDigits.

Dengan kedua method tersebut menghasilkan Citra yang sudah disisipkan pesan text dengan terenkripsi yang tervisualisasikan pada gambar berikut :

1779,520 : 010 | 11,1002 : 101 | 416,940 : 110 | 1346,431 : 110 | 1412,913 : 010 | 1770,213 : 001 | 388,577 : 011 | 1418,679 : 000 | 413,769 : 011 | 285,852 : 000 | 4,1068 : 110 | 1225,325 : 100 | 1600,605 : 001 | 1731,148 : 101 | 567,739 : 000 | 1910,594 : 010 | 853,389 : 011 | 1572,325 : 010 | 94,679 : 010 | 520,56 : 110 | 288,262 : 011 | 516,1036 : 101 | 325,83 : 000 | 753,71 : 110 | 1376,106 : 001 | 876,273 : 101 | 1510,346 : 110 | 22,983 : 101 | 1112,583 : 011 | 943,901 : 001 | 1309,524 : 101 | 234,402 : 010 | 1719,909 : 010 | 1338,992 : 000 | 409,336 : 110 | 1726,67 : 100 | 1498,607 : 010 | 1170,381 : 101 | 1250,921 : 110 | 1526,223 : 101 | 893,62 : 010 | 1402,607 : 010 | 942,38 : 110 | 984,946 : 111 | 288,717 : 100 | 1559,107 : 101 | 1701,304 : 111 | 840,108 : 000 | 435,140 : 010 | 437,1062 : 010 | 406,830 : 100 | 1842,1009 : 011 | 1306,360 : 100 | 1243,156 : 001 | 574,145 : 110 | 1649,542 : 110 | 886,158 : 010 | 81,756 : 100 | 214,1066 : 100 | 1506,384 : 100 | 869,754 : 011 | 621,1057 : 101 | 911,370 : 100 | 525,274 : 100 | 249,118 : 010 | 881,281 : 101 | 1064,17 : 100 | 869,886 : 111 | 1360,672 : 001 | 851,926 : 000 | 99,145 : 110 | 1626,144 : 000 | 779,35 : 011 | 1771,909 : 000 | 325,705 : 010 | 1241,586 : 101 | 831,803 : 100 | 207,724 : 001 | 1140,760 : 110 | 1324,1012 : 011 | 959,778 : 010 | 249,610 : 010 | 1141,145 : 010 | 957,0 : 100 | 531,354 : 100 | 1138,972 : 101 | 1150,21 : 100 | 838,248 : 110 | 615,89 : 011 | 412,398 : 101 | 1179,726 : 000 | 963,945 : 100 | 1735,736 : 000 | 1290,647 : 101 | 797,574 : 110 | 636,997 : 000 | 180,1008 : 010 | 1710,616 : 011 | 1101,212 : 110 | 1472,1048 : 110 | 874,956 : 101 | 547,619 : 001 | 130,442 : 000 | 552,197 : 110 | 876,1046 : 011 | 1564,178 : 011 | 1323,661 : 100 | 591,810 : 011 | 1778,167 : 011 | 368,371 : 101 | 948,382 : 010 | 1517,457 : 111 | 1361,310 : 011 | 890,811 : 101 | 698,65 : 000 | 554,497 : 111 | 381,551 : 100 | 280,839 : 101 | 245,361 : 010 | 21,613 : 010 | 1347,652 : 001 | 573,977 : 011 | 1512,391 : 110 | 1285,799 : 110 | 143,763 : 100 | 607,661 : 001 | 7,1066 : 010 | 1034,437 : 000 | 691,212 : 001 | 342,344 : 100 | 855,991 : 100 | 1432,135 : 111 | 447,501 : 100 | 1086,450 : 001 | 702,735 : 101 | 1557,441 : 001 | 410,768 : 010 | 1193,88 : 000 | 1584,818 : 100 | 1501,623 : 110 | 793,154 : 111 | 952,285 : 101 | 973,816 : 101 | 235,663 : 010 | 1052,111 : 011 | 713,108 : 010 | 279,1070 : 100 | 74,777 : 110 | 146,70 : 001 | 1303,1056 : 001 | 641,162 : 011 | 692,432 : 010 | 92,36 : 011 | 421,874 : 011 | 432,726 : 110 | 1157,737 : 100 | 1441,436 : 010 | 123,1030 : 001 | 1790,900 : 000 | 472,222 : 011 | 1066,251 : 011 | 46,872 : 011 | 965,534 : 010 | 1583,343 : 011 | 1747,1060 : 000 | 586,183 : 101 | 772,647 : 111 | 1164,197 : 010 | 746,80 : 011 | 1127,355 : 001 | 135,807 : 00

Gambar 22. Hasil Pengacakan Posisi Pixel PRNG

Gambar 23. Citra Stego hasil penyisipan

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Penelitian ini berhasil mengembangkan sistem pengamanan informasi berbasis citra digital menggunakan kombinasi teknik kriptografi Rijndael dan AES 256, algoritma LCG, dan steganografi LSB.

Metode ini mampu menyembunyikan pesan secara aman dengan mendistribusikan bit pesan secara acak menggunakan algoritma LCG. Kriptografi Rijndael dan AES 256 memastikan bahwa pesan tetap terlindungi meskipun posisi bit ditemukan, dengan kualitas visual yang sangat baik, dengan nilai PSNR di atas 40 dB dan nilai MSE yang rendah, menunjukkan bahwa perubahan visual tidak terdeteksi oleh mata manusia.

B. Saran

Sistem ini menawarkan solusi praktis dan efisien untuk pengamanan informasi dalam citra digital, menjadikannya relevan untuk aplikasi komunikasi rahasia.

Penelitian ini dapat diperluas dengan mengintegrasikan metode ini ke dalam platform berbasis web atau mobile untuk penggunaan yang lebih luas.

DAFTAR PUSTAKA

[1] W. Easttom, *Modern Cryptography*. Cham: Springer International Publishing, 2021. doi: 10.1007/978-3-030-63115-4.

[2] D. Megías, W. Mazurczyk, and M. Kuribayashi, "Data Hiding and Its Applications: Digital Watermarking and Steganography," *Applied Sciences*, vol. 11, no. 22, p. 10928, Nov. 2021, doi: 10.3390/app112210928.

[3] A. Siswanto, Y. Arta, E. Abdul Kadir, and Bimantara, "Text File Protection Using Least Significant Bit (LSB) Steganography and Rijndael Algorithm," 2021, pp. 205–213. doi: 10.1007/978-981-15-7990-5_20.

[4] E. Nirmala, "Penerapan Steganografi File Gambar Menggunakan Metode Least Significant Bit (LSB) dan Algoritma Kriptografi Advanced Encryption Standard (AES) Berbasis Android," *Jurnal Informatika Universitas Pamulang*, vol. 5, no. 1, p. 36, Mar. 2020, doi: 10.32493/informatika.v5i1.4646.

[5] J. Daemen *et al.*, "The Design of Rijndael AES-The Advanced Encryption Standard Springer-Verlag," 2001.

[6] J. Daemen and V. Rijmen, "Specification of Rijndael," in *Specification of Rijndael," in The Design of Rijndael: The Advanced Encryption Standard (AES)*, 2nd ed., vol., Switzerland: Springer, 2020, pp. 31–51. doi: 10.1007/978-3-662-60769-5_3.

[7] T. R. Kuncoro and R. Aditama, "ANALISIS KOMBINASI ALGORITMA KRIPTOGRAFI RSA DAN ALGORITMA STEGANOGRAFI LEAST SIGNIFICANT BIT (LSB) DALAM PENGAMANAN PESAN DIGITAL," *STATMAT : JURNAL STATISTIKA DAN MATEMATIKA*, vol. 1, no. 2, Aug. 2019, doi: 10.32493/sm.v1i2.2947.

[8] A. Muh. Ramadhani and Tasrif Hasanuddin, "Modifikasi Least Significant Bits pada Gambar sebagai Data Hiding Steganography," *Indonesian Journal of Data and Science*, vol. 2, no. 2, pp. 91–102, Jul. 2021, doi: 10.56705/ijodas.v2i3.48.

[9] I. M. Yusup, C. Carudin, and I. Purnamasari, "Implementasi Algoritma Caesar Cipher Dan Steganografi Least Significant Bit Untuk File Dokumen," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 6, no. 3, Dec. 2020, doi: 10.28932/jutisi.v6i3.2817.

[10] A. Gustiawan *et al.*, "Perancangan Aplikasi Steganografi Pada Citra Digital Menggunakan Metode Pixel Value Differencing Oleh : Perancangan Aplikasi Steganografi Pada Citra Digital Menggunakan Metode Pixel Value Differencing," *JUKI : Jurnal Komputer dan Informatika*, vol. 5, 2023.

[11] A. Hernandes, H. Hartini, and D. Sartika, "Steganografi Citra Menggunakan Metode Least Significant Bit (LSB) dan Linear Congruential Generator (LCG)," *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, vol. 5, no. 2, pp. 137–146, Mar. 2019, doi: 10.35957/jatisi.v5i2.134.

[12] A. Wahid, "Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi," Oct. 2020.

[13] K. Umam, "Digital Watermarking Using Advanced Encryption Standard (AES) And Steganography Methods to Increase The Level Of Security Of User Data In The Image Files On Internet Network," *Innovation in Research of Informatics (INNOVATICS)*, vol. 2, no. 2, Sep. 2020, doi: 10.37058/innovatics.v2i2.1178.

[14] M. Alanzy, R. Alomrani, B. Alqarni, and S. Almutairi, "Image Steganography Using LSB and Hybrid Encryption Algorithms," *Applied Sciences*, vol. 13, no. 21, p. 11771, Oct. 2023, doi: 10.3390/app132111771.

[15] Omg, "An OMG ® Unified Modeling Language ® Publication OMG ® Unified Modeling Language ® (OMG UML ®) OMG Document Number: Date," 2009. [Online]. Available: https://www.omg.org/spec/UML/20161101/PrimitiveTypes.xmi

[16] James. Rumbaugh, Ivar. Jacobson, and Grady. Booch, *The unified modeling language reference manual*. Addison-Wesley Longman, 2000.

[17] Gunawan Arie, Ningsih Sari, Lantana Dhieka Avrillia *PENGANTAR BASIS DATA*, 1st ed. Malang: PT. Literasi Nusantara Abadi Grup, 2023. [Online]. Available: www.penerbitlitnus.co.id