IMPLEMENTASI KRIPTOGRAFI DALAM PENYISIPAN PESAN PADA CITRA DIGITAL MENGGUNAKAN METODE *PLAYFAIR CIPHER* DAN *LEAST SIGNIFICANT BIT* (LSB)

Disusun Oleh:

Reza Firlanda Berliana	082011233003
Febri Anggi Tri Kusumaningsih	082011233004
Islandsius	082011233076
Akrom Fuadi	082011233079

Dosen Pengampu:

Drs. Edi Winarko, M.Cs.

PROGRAM STUDI MATEMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS AIRLANGGA

2023

KATA PENGANTAR

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa karena atas berkat dan karunia-Nya sehingga kami dapat menyelesaikan makalah yang berjudul "Implementasi Kriptografi Dalam PenyisipanPesan Pada Citra Digital Menggunakan Metode *Playfair Cipher* Dan *Least Significant Bit* (LSB)". Makalah ini dibuat untuk memenuhi tugas dalam mata kuliah Aplikasi Kriptografi.

Terima kasih kami sampaikan kepada Bapak Drs. Edi Winarko, M.Cs. yang telah membimbing kami semua selama satu semester ini dalam mata kuliah Aplikasi Kriptografi, sehingga ilmu yang diberikan dapat kami manfaatkan untuk menyelesaikan permasalahan dalam kehidupan sehari-hari.

Kami menyadari bahwa tugas akhir ini masih jauh dari kata sempurna. Oleh karena itu, kritik dan saran pembaca sangat kami harapkan untuk perbaikan tugas akhir kami selanjutnya. Kami mohon maaf apabila dalam tugas akhir ini terdapat kesalahan, baik dalam penulisan maupun isi makalah ini. Terima kasih.

Surabaya, 24 Oktober 2023

Penulis

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 LATAR BELAKANG

Keamanan informasi saat ini merupakan salah satu komponen terpenting di era perkembangan teknologi yang semakin pesat. Bagi beberapa orang, sebuah informasi termasuk sesuatu yang berharga maka dari itu terkadang seseorang tidakingin orang lain mengetahui informasi tersebut. Namun sering kali informasidisalahgunakan oleh pihak yang tidak bertanggung jawab untuk meraih keuntungan atau pun hanya untuk merusaknya.

Salah satu cara untuk mengatasi masalah keamanan informasi adalah dengan diterapkannya teknik kriptografi. Kriptografi adalah ilmu mengenai teknik enkripsi dimana data diacak menggunakan suatu kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yang tidak memiliki kunci deskripsi. Tujuan adanya kriptografi ialah agar sebuah data yang disampaikan hanya diketahui oleh orang yang dituju ataupun yang berhak untuk mengetahuinya sehingga tidak disalah gunakan oleh orang-orang yang tidak bertanggung jawab.

Kriptografi terdapat berbagai macam teknik yang dapat digunakan dalam upaya pengamanan data salah satu nya ialah *Playfair Cipher*. *Playfair cipher* tergolong dalam kriptografi klasik, *Playfair* merupakan *digraphs cipher* ialahsetiap proses enkripsi dilakukan pada dua huruf atau pasangan huruf. *Playfair Cipher* mengenkripsi pasangan huruf, bukan huruf tunggal seperti pada cipher klasik lainnya. Tujuannya untuk membuat analisis frekuensi menjadi sulit sebab frekuensi kemunculan huruf didalam cipherteks akan menjadi datar. Salah satu teknik dalam penyembunyian pesan. Steganografi adalah suatu ilmu seni dalam menyembunyikan informasi dengan memasukkan informasi atau pesan tersebut ke dalam media lain. Sehingga keberadaan informasi tersebut tidak diketahui orang lain. Media yang dapat menampung informasi tersebut adalah citra digital, video, audio, dan teks.

Didalam dunia teknologi tidak cukup apabila hanya menggunakankriptografi saja, ada baiknya menggunakan teknik steganografi. Steganografi juga merupakan lain yang mengetahuinya. Terdapat dua jenis dalam kriptografi yaitu kriptografi klasik dan kriptografi modern. Kriptografi klasik merupakan metode untuk mengubah data asli (*Plainteks*) ke bentuk sandi (*Cipherteks*) dengan menggunakan kunci yang sama.

Sedangkan Kriptografi Modern ialah Metode yang menggunakan dua buah kunci yakni kunci publik (*Public key*) yang dapat dipublikasikan dan kunci privat (*Private key*) ialah kunci yang harus dirahasiakan.

Salah satu metode yang dapat digunakan dalam steganografi adalah *Least Significant Bit* (LSB). LSB merupakan teknik penyembunyian pesan pada lokasi bit terendah dalam citra digital. Pesan rahasia akan dikonversikan ke dalam bentuk biner dan disembunyikan kedalam sebuah media penyembunyian berupa citra digital. Hasil dari penyembunyian pesan menggunakan metode LSB tidak memiliki perubahan sehingga sulit dibedakan oleh mata manusia.

Dari latar belakang masalah diatas, penulis bermaksud untuk mengimplementasikan salah satu jenis kriptografi yaitu algoritma *Playfair Cipher* untuk memenuhi kebutuhan keamanan pesan rahasia dan akan dikombinasikan dengan metode steganografi yaitu *Least Significant Bit* (LSB).

Kombinasi dari teknik kriptografi *Playfair Cipher* dan steganografi LSB diharapkan dapat lebih meningkatkan pengamanan pada informasi yang bersifat rahasia. Sebuah pesan rahasia akan dienkripsi terlebih dahulu menggunakan *Playfair Cipher* kemudian hasil kriptografi tersebut akan disembunyikan didalam citra digital menggunakan metode steganografi LSB.

1.2 RUMUSAN MASALAH

Berdasarkan latar belakang yang telah diuraikan di atas, maka dapat dibuatrumusan masalah sebagai berikut :

1. Bagaimana merancang dan mengimplementasikan sistem pengamanan pesan rahasia berupa teks menggunakan metode kriptografi *Playfair cipher* dengan *Least Significant Bit* (LSB) kedalam aplikasi berbasis website ?
2. Bagaimana melakukan proses enkripsi dan deskripsi dengan *Playfair cipher* dan *Least Significant Bit* (LSB) terhadap pesan rahasia?
3. Bagaimana proses penyisipan pesan kedalam citra digital menggunakan metode *Playfair Cipher* dan *Least Significant Bit* (LSB) ?

1.3 TUJUAN

Tujuan yang ingin dicapai dari penyusunan penelitian ini adalah sebagai berikut :

1. Merancang dan mengimplementasikan sistem pengamanan pesan berupa teks menggunakan metode kriptografi *Playfair cipher* dengan *Least Significant Bit* (LSB) kedalam aplikasi berbasis website .
2. Untuk mengetahui proses enkripsi dan deskripsi dengan *Playfair cipher* dan *Least Significant Bit* (LSB) terhadap pesan rahasia.
3. Untuk mengetahui proses penyisipan pesan kedalam citra digital menggunakan metode *Playfair Cipher* dan *Least Significant Bit* (LSB) terhadap pesan rahasia.

1.4 MANFAAT

Manfaat yang dapat diambil dari penyusunan penelitian ini adalah :

1. Dapat merancang dan mengimplementasikan sistem pengamanan pesan berupa teks menggunakan metode kriptografi *Playfair cipher* dengan *LeastSignificant Bit* (LSB) kedalam aplikasi berbasis website .
2. Dapat mengetahui dan melakukan enkripsi dan deskripsi dengan *Playfair cipher* dan *Least Significant Bit* (LSB) terhadap pesan rahasia.
3. Dapat mengetahui dan melakukan proses penyisipan pesan kedalam citra digital menggunakan metode *Playfair Cipher* dan *Least Significant Bit* (LSB) terhadap pesan rahasia.

BAB II
TINJAUAN PUSTAKA

2.1 KRIPTOGRAFI

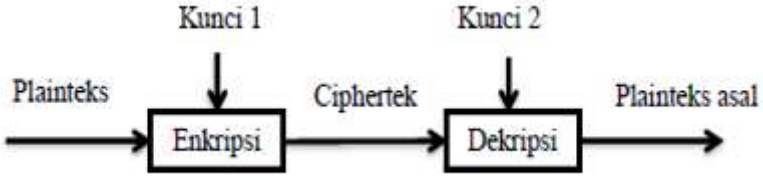
Kriptografi berasal dari bahasa Yunani "*cryptos*" artinya rahasia *(secret),* sedangkan *"graphein"* artinya tulisan rahasia *(writing)*. Kriptografi adalah ilmu ataupun seni yang mempelajari bagaimana membuat suatu pesan yang dikirim oleh pengirim dapat disampaikan kepada penerima dengan aman **(Schneier, 1996).** Ada beberapa definisi kriptografi yang telah dikemukakan di dalam berbagailiteratur. Definisi yang dipakai di dalam buku-buku yang lama (sebelum tahun 1980-an) menyatakan bahwa kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidakdapat dimengerti maknanya. Definisi ini mungkin cocok pada masa lalu dimana kriptografi digunakan untuk keamanan komunikasi penting seperti komunikasi dikalangan militer, diplomat, dan mata-mata. Namun saat ini kriptografi lebih sekedar *privacy*, tetapi juga untuk tujuan data *integrity*, *autentication*, dan non- repudiation.

Kriptografi merupakan salah satu teknik dari beberapa teknik keamanan data yang sering digunakan untuk mengamankan data, seperti halnya menyandikan pesan asli (*plaintext*) ke dalam bentuk pesan rahasia (*ciphertext*). Proses pengaman ini melibatkan algoritma dan kunci. Kunci yang telah dienkripsi dapat dengan mudah dideskripsi kembali, yaitu dari *ciphertext* menjadi *plaintext*. Oleh karena itu diperlukan algoritma kriptografi yang kuat. Namun teknik ini masih menimbulkan kecurigaan pada orang lain yang melihat pesan tersebut.

Menurut **Munir, (2010)**, Kriptografi adalah ilmu sekaligus seni untuk menjaga keamanan pesan. Keamanan pesan diperoleh dengan menyandikannya menjadi pesan yang tidak mempunyai makna. Pesan yang dirahasiakan dinamakanplainteks, sedangkan pesan hasil penyandian disebut cipherteks. Proses penyandian plainteks menjadi cipherteks disebut enkripsi dan proses membalikkan cipherteks menjadi plainteks asalnya disebut dekripsi.

Algoritma kriptografi adalah urutan langkah-langkah logis bagaimana menyembunyikan pesan dari orang-orang yang tidak berhak atas pesan tersebut. Algoritma kriptografi terdiri dari tiga fungsi dasar yaitu:

1. Enkripsi: merupakan hal yang sangat penting dalam kriptografi pengamanan data yang dikirimkan agar terjaga kerahasiaanya. Pesan asli disebut plaintext, yang diubah menjadi kode-kode yag tidak dimengerti. Enkripsi dapat diartikan dengan *cipher* atau kode. Sama halnya dengan kita tidak mengerti akan sebuah kata maka kita akan melihatnya didalam kamus atau daftar istilah. Beda halnya dengan enkripsi, untuk mengubah text asli kebentuk text kode kita menggunakan algoritma yang dapat mengkodekan data yang kita inginkan.
2. Dekripsi: merupakan kebalikan dari enkripsi. Pesan yang telah dienkripsi dikembalikan kebentuk asalnya (text asli), disebut dengan dekripsi pesan. Algoritma yang digunakan untuk dekripsi tentu berbeda dengan algoritma yang digunakan untuk enkripsi.
3. Kunci: yang dimaksud disini adalah kunci yang dipakai untuk melakukan enkripsi dan dekripsi. Kunci terbagi menjadi dua bagian, kunci rahasia(*private key*) dan kunci umum (*public key*).



2.2 PLAYFAIR CHIPER

Playfair pada tahun 1854 dan pertama kali digunakan oleh bangsa Inggris **(Stinson, 1995)**. Cipher ini mengenkripsi pasangan karakter (bigram atau digraf) bukan karaker tunggal seperti pada cipher klasik lainnya. Tujuannya adalah untuk membuat analisis frekuensi menjadi sangat sulit sebab frekuensi kemunculan karakter-karakter didalam *ciphertext* menjadi datar atau *flat* **(Sasongko, 2005).**

Pada proses penyandian matriks kunci akan diisi sesuai dengan urutan kemunculan huruf pada kunci. Huruf yang digunakan tidak boleh digunakan lagi, sedangkan huruf yang tidak digunakan kunci akan disusun setelahnya sesuai dengan urutan alphabet. Menurut *Stallings* Sebelum melakukan enkripsi, pesan yang akan dienkripsi (*plaintext*) diatur terlebih dahulu sebagai berikut:

1. Semua spasi dan karakter yang bukan alfabet harus dihilangkan dari *plaintext* (jika ada).
2. Jika ada huruf J pada *plaintext*, maka ganti huruf tersebut dengan huruf I.

3. Pesan yang akan dienkripsi ditulis dalam pasangan huruf (bigram).
4. Jika ada huruf yang sama dalam pasangan huruf, maka sisipkan huruf X atau Z di tengahnya. Huruf yang disisipkan sebaiknya huruf X karena sangat kecil kemungkinan terdapat huruf X yang sama dalam bigram, tidak seperti huruf Z
5. Jika jumlah huruf pada *plaintext* adalah ganjil maka pilih sebuah huruf tambahan yang dipilih oleh orang yang mengenkripsi dan tambahkan di akhir *plaintext*. Huruf tambahan dapat dipilih sembarang misalnya huruf Z atau X.

2.3 STEGANOGRAFI

Pada teknik kriptografi, data yang telah disandikan (*chiperteks*) tetap tersedia, maka dengan steganografi *cipherteks* dapat disembunyikan sehingga pihak ketiga tidak mengetahui keberadaannya. Kata steganografi (*steganography*) berasal dari bahasa Yunani yaitu *steganos* yang berarti tersembunyi atau terselubung dan *graphia* yang artinya menulis, sehingga arti steganografi adalah "menulis (tulisan) terselubung" **(Darmayanti, 2016)**.

Steganografi juga merupakan ilmu menyembunyikan teks pada media lain yang telah ada sedemikian sehingga teks yang tersembunyi menyatu dengan media itu. Menurut Cahyadi (2012), terdapat beberapa contoh media penyisipan pesan rahasia yang digunakan dalam teknik steganografi antara lain:

1. Teks

   Dalam algoritma steganografi yang menggunakan teks sebagai media penyisipan biasanya digunakan teknik NLP sehingga teks yang telah disisipkan pesan rahasia tidak akan mencurigakan untuk orang yang melihatnya.
2. Audio

   Format ini pun sering dipilih karena biasanya berkas dengan format ini berukuran relatif besar, sehingga dapat menampung pesan rahasia dalam jumlah yang besar pula.
3. Citra

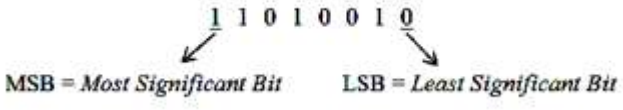
   Format ini juga sering digunakan karena format ini merupakan salah satu format file yang sering dipertukarkan dalam dunia internet. Alasan lainnya adalah banyaknya tersedia algoritma steganografi untuk media penampung yang berupa citra.
4. Video

   Format ini memang merupakan format dengan ukuran file yang relatif sangat

besar namun jarang digunakan karena ukurannya yang terlalu besar sehingga mengurangi kepraktisannya dan juga kurangnya algoritma yang mendukung format ini.

2.4 LEAST SIGNIFICANT BIT (LSB)

Metode *Least Significant Bit* (LSB) merupakan metode steganografi yang paling sederhana dan paling mudah diimplementasikan. LSB merupakan teknik penyembunyian pesan pada lokasi bit terendah dalam citra digital. Pesan rahasia akan dikonversikan ke dalam bentuk biner dan disembunyikan kedalam sebuah media penyembunyian berupa citra digital. Hasil dari penyembunyian pesan menggunakan metode LSB tidak memiliki perubahan sehingga sulit dibedakan oleh mata manusia. Untuk menjelaskan metode LSB ini kita menggunakan citra digital sebagai *cover – object*. Setiap pixel di dalam citra berukuran 1 sampai 3 *byte*. Pada susunan bit di dalam sebuah *byte* (1 *byte* = 8 bit), ada bit yang paling berarti yaitu MSB (*most significant bit*) dan bit yang paling kurang berarti yaitu LSB (*least significant bit*).



Dari contoh *byte* pada Gambar 2.4, bit 1 yang pertama (digaris bawahi) adalah MSB dan bit 0 yang terakhir (digaris bawahi) adalah LSB. Bit yang cocok untuk diganti dengan bit pesan adalah LSB, karena modifikasi hanya menggunakan nilai *byte* tersebut satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Misalnya *byte* tersebut di dalam citra memberikan persepsi warna merah, maka perubahan satu bit LSB hanya mengubah persepsi warna merah tidak terlalu berarti. Mata manusia tidak dapat membedakan perubahan sekecil ini. Sebagai ilustrasi, misalkan segmen pixel - pixel citra sebelum disisipkan pesan adalah:

00110011 10100010 11100010 01101111

Dan misalkan pesan rahasia (yang telah dikonversi ke biner) adalah 0110. Setiap bit pesan menggantikan posisi LSB dari segmen pixel – pixel citra menjadi:

00110010 10100011 11100011 10010000

2.5 CITRA DIGITAL

Citra adalah suatu representasi (gambaran), kemiripan atau imitasi dari suatu objek. Citra yang berupa keluaran dari suatu sistem perekaman data dapat bersifat optik berupa foto, bersifat analog berupa sinyal – sinyal video seperti gambar pada monitor televisi atau bersifat digital yang dapat langsung disimpan pada suatu media penyimpanan (Sutoyo, 2009).

Jenis-jenis citra digital antaralain :

1. Citra Biner

   Citra biner adalah citra yang hanya memiliki 2 warna, yaitu hitam dan putih. Oleh karena itu, setiap pixel pada citra biner cukup direpresentasikan dengan 1 bit. Alasan penggunaan citra biner adalah karena citra biner memiliki sejumlah keuntungan sebagai berikut:

   a. Kebutuhan memori kecil karena nilai derajat keabuan hanya membutuhkan representasi 1 bit.
   b. Waktu pemrosesan lebih cepat di bandingkan dengan citra hitam – putih ataupun warna.

2. Citra Grayscale

   Citra grayscale adalah citra yang nilai pixel-nya merepresentasikan derajat keabuan atau intensitas warna putih. Nilai intensitas paling rendah merepresentasikan warna hitam dan nilai intensitas paling tinggi merepresentasikan warna putih. Pada umumnya citra *grayscale* memiliki kedalaman pixel 8 bit (256 derajat keabuan), tetapi ada juga citra *grayscale* yang kedalaman pixel-nya bukan 8 bit, misalnya 16 bit untuk penggunaan yang memerlukan ketelitian tinggi.

3. Citra Warna

   Citra warna adalah citra yang nilai pixel-nya merepresentasikan warna tertentu. Setiap pixel pada citra warna memiliki warna yang merupakan kombinasi dari tiga warna dasar RGB (*red, green, blue*). Setiap warna dasar menggunakan penyimpanan 8 bit = 1 byte, yang berarti setiap warna mempunyai gradasi

sebanyak 255 warna. Berarti setiap pixel mempunyai kombinasi warna sebanyak $2^8.2^8.2^8 = 2^{24} = 16$ juta warna lebih. Itulah yang menjadikan alasan format ini disebut dengan *true color* karena mempunyai jumlah warna yang cukup besar sehingga bisa dikatakan hampir mencakup semua warna di alam.

2.6 PYTHON

Python adalah sebuah bahasa pemrograman tingkat tinggi yang dikembangkan oleh Guido van Rossum pada awal 1990-an. Salah satu keunggulan utama Python adalah sintaksnya yang mudah dibaca dan jelas, yang memungkinkan pengembang untuk menulis kode dengan cara yang lebih terstruktur dan efisien. Selain itu, Python dikenal dengan filosofi "baterai termasuk", yang berarti bahasa ini dilengkapi dengan berbagai pustaka dan modul yang mendukung berbagai tugas pemrograman, mulai dari pengembangan web hingga kecerdasan buatan.

Keuniversalan Python menjadikannya salah satu bahasa pemrograman yang paling populer di dunia. Ia digunakan dalam berbagai industri dan aplikasi, termasuk pengembangan web, analisis data, pemrosesan bahasa alami, dan banyak lagi. Fleksibilitasnya dalam menangani berbagai tugas dan kemampuan untuk diintegrasikan dengan bahasa pemrograman lain menjadikannya pilihan yang ideal bagi pengembang dan ilmuwan data.

Salah satu kekuatan utama Python adalah komunitas yang kuat dan aktif di baliknya. Komunitas Python yang besar terus berkontribusi dalam mengembangkan pustaka, alat, dan sumber daya pendidikan. Dengan demikian, Python tidak hanya menjadi sebuah bahasa pemrograman, tetapi juga sebuah ekosistem yang mendukung pertumbuhan dan inovasi dalam bidang teknologi informasi.

BAB III
METODOLOGI PENELITIAN

Pada penelitian ini terlebih dahulu mempelajari materi-materi dasar untuk penelitian ini seperti kriptografi, steganografi, serta mempelajari pembuatan web menggunakan PHP. Selanjutnya melakukan enkripsi dan proses penyisipan pesan kedalam sebuah gambar lalu melakukan proses deskripsi dan ekstraksi untuk mendapatkan pesan asli. Untuk implementasi dari proses enkripsi pesan, maka dilakukan langkah berikut :

1. User menginputkan *Plainteks.*
2. Apabila plainteks terdapat huruf **J** maka akan diganti dengan huruf **I**.
3. Plainteks akan disusun menjadi bigram.
4. Apabila huruf tidak memiliki pasangan bigram maka ditambahkan huruf **'X'** pada akhir bigram.
5. Inputkan Kunci.
6. Urutkan sesuai dengan kemunculan huruf kunci, kemudian urutkan huruf belum terpakai pada kunci.
7. Lakukan enkripsi menggunakan aturan algoritma Playfair Cipher dari *plainteks.* Jika dua huruf tidak pada baris yang sama atau kolom yang sama, maka huruf pertama diganti dengan huruf pada perpotongan baris huruf pertama dengan kolom huruf kedua
8. Setelah melakukan enkripsi terhadap seluruh *plainteks,* maka diperoleh *cipherteks.*

Proses penyisipan pesan kedalam sebuah gambar menggunakan metode *Least Significant Bit*. LSB mengganti setiap ujung bit citra dengan bit pesan yang akan disisipkan. Berikut implementasi dari metode ini adalah sebagai berikut:

1. Chipherteks yang sebelumnya didapat dari proses enkripsi di inputkan.
2. Setelah diperoleh hasil cipherteks, maka setiap karakter diubah kebentuk kode ASCII dan diubah kedalam bentuk biner agar dapat diimplementasikan ke *cover image* yang digunakan
3. Pesan yang akan disisipkan harus di representasikan kedalam biner.
4. Kemudian siapkan citra gambar yang akan menjadi wadah penampung pesan rahasia, kemudian ubah pixel citra ke biner.
5. Siapkan Data cipherteks yang setiap karakternya telah diubah kedalambentuk biner dan akan disisipkan kedalam bit gambar.

6. Melakukan Proses penyisipan dengan mengganti bit terakhir dari gambar dengan bit dari setiap karakter *ciphertext.* Dari proses penyisipan karakter *ciphertext* di peroleh s*tego image.* Perubahan piksel- piksel gambar hanya terjadi pada bit-bit paling belakang dari *stego image.* Dengan perubahan yang tidak signifikan tidak akan terdeteksi oleh mata manusia sehingga tidak akan mengandung kecurigaan.

Proses ektraksi (Decoding) merupakan cara untuk mengambil pesan yang disisipkan ke dalam gambar. implementasi dari proses ektraksi menggunakan *least significant bit* :

1. Mengubah nilai matriks citra s*tego image* kedalam bentuk biner . Kemudian mengambil bit terakhir dari setiap piksel , nilai biner tersebut merupakan biner pesan yang telah disisipkan kedalam citra.
2. Mengubah nilai biner pesan yang disisipkan kedalam bentuk desimal agar dapat diketahui pesan yang dimaksud. Nilai desimal inilah yang merupakan nilai bentuk ASCII dari pesan tersebut.

Proses dekripsi dilakukan setelah berhasil didapatnya *cipherteks* dari proses ekstraksi gambar. Deskripsi pesan menggunakan *playfair cipher* merupakan kebalikan dari prosesenkripsi, berikut adalah implementasi dari proses deskripsi:

1. Siapkan *Ciphertext.*
2. Lakukan deskripsi menggunakan aturan algoritma *Playfair Cipher* dari *ciphertext.*
3. Kemudian hasil dari enkripsi tersebut merupakan *plaintext* ataupun pesanasli

BAB IV
PEMBAHASAN

4.1 PERHITUNGAN MANUAL

Proses perhitungan dari prosedur enkripsi dengan metode *Playfair Cipher* dapat dijabarkan sebagai berikut:

1. Pesan rahasia = **BACA**
2. Kunci rahasia = **TES**
3. Plainteks akan disusun menjadi bigram : **BA CA**.
4. Apabila huruf tidak memiliki pasangan bigram maka ditambahkan huruf **X** pada akhir bigram. Plainteks menjadi **BA CA**.
5. Buat matriks kunci dengan ukuran 6 x 6, dengan cara urutkan sesuai dengan kemunculan huruf kunci, kemudian urutkan huruf belum terpakai pada kunci.

T	E	S	A	B	C
D	F	G	H	I	K
L	M	N	O	P	Q
R	U	V	W	X	Y
Z	␣	0	1	2	3
4	5	6	7	8	9

Gambar 4.1 Matriks Kunci “TES”

6. Lakukan enkripsi dari plainteks “BA” dan menghasilkan cipherteks “CB”.

T	E	S	**A**	**B**	C		T	E	S	A	**B**	**C**
D	F	G	H	I	K		D	F	G	H	I	K
L	M	N	O	P	Q		L	M	N	O	P	Q
R	U	V	W	X	Y		R	U	V	W	X	Y
Z	␣	0	1	2	3		Z	␣	0	1	2	3
4	5	6	7	8	9		4	5	6	7	8	9

Gambar 4.2 Matriks Enkripsi “BA”

7. Lakukan enkripsi dari plainteks “CA” dan menghasilkan cipherteks “TB”.

T	E	S	**A**	B	**C**
D	F	G	H	I	K
L	M	N	O	P	Q
R	U	V	W	X	Y
Z	␣	0	1	2	3
4	5	6	7	8	9

T	E	S	A	**B**	C
D	F	G	H	I	K
L	M	N	O	P	Q
R	U	V	W	X	Y
Z	␣	0	1	2	3
4	5	6	7	8	9

Gambar 4.3 Matriks Enkripsi "CA"

8. *Ciphertext* yang diperoleh = CBTB.

Setelah itu, *ciphertext* akan disisipkan ke dalam citra sampul. Proses perhitungan dari penyisipan data dengan metode LSB dapat dirincikan sebagai berikut:

1. Input citra sampul.

Gambar 4.4 Citra Sampul

2. Baca warna elemen RGB dari setiap piksel pada citra sampul.

Tabel 4.1 Tabel Citra Sampul

216,13,46	239,107,49	61,44,1	114,120,244	186,146,138
94,182,178	208,137,12	140,252,83	113,4,165	15,127,152
249,25,181	246,232,108	171,101,106	191,72,242	239,151,85
151,193,14	254,246,72	91,1,23	242,210,201	193,39,144
81,145,192	61,116,35	150,168,203	129,198,122	176,108,120

3. Baris pertama (baris paling atas) akan digunakan untuk menyimpan jumlahkarakter pada *ciphertext*. Jumlah karakter dibatasi maksimal 255 karakter,yang berarti bit panjang ciphertext yang akan disisipkan adalah sebesar 8 bit. Panjang *ciphertext* = 4 karakter. Nilai ini akan dikonversikan ke biner menjadi**0000 0100**.

   Sisipkan setiap bit ke dalam elemen warna RGB dari setiap piksel.

   Piksel (1, 1)

216	= 1101 1000	= 1101 100**0**	**= 216**
13	= 0000 1101	= 0000 110**0**	**= 12**
46	= 0010 1110	= 0010 111**0**	**= 46**

   Piksel (1, 2)

239	= 1110 1111	= 1110 111**0**	**= 238**
107	= 0110 1011	= 0110 101**0**	**= 106**
49	= 0011 0001	= 0011 000**1**	**= 49**

   Piksel (1, 3)

61	= 0011 1101	= 0011 110**0**	**= 60**
44	= 0010 1100	= 0010 110**0**	**= 44**
1	= 0000 0001		**= 1**

4. *Ciphertext* akan disisipkan mulai dari baris 2 dari citra sampul.

   Ciphertext yang akan disisipkan

   C = 67 = 0100 0011

   B = 66 = 0100 0010

   T = 84 = 0101 0100

   B = 66 = 0100 0010

   Bit yang akan disisipkan: **0100 0011 0100 0010 0101 0100 0100 0010**

   Piksel (2, 1)

94	= 0101 1110	= 0101 111**0**	**= 94**
182	= 1011 0110	= 1011 011**1**	**= 183**
178	= 1011 0010	= 1011 001**0**	**= 178**

Piksel (2, 2)

208	= 1101 0000	= 1101 000**0**	**= 208**
137	= 1000 1001	= 1000 100**0**	**= 136**
12	= 0000 1100	= 0000 110**0**	**= 12**

Piksel (2, 3)

140	= 1000 1100	=1000 110**1**	**= 141**
252	= 1111 1100	=1111 110**1**	**= 253**
83	= 0101 0011	=0101 001**0**	= **82**

Piksel (2, 4)

113	= 0111 0001	= 0111 000**1**	**= 113**
4	= 0000 0100	= 0000 010**0**	**= 4**
165	= 1010 0101	= 1010 010**0**	**= 164**

Piksel (2, 5)

15	= 0000 1111	= 0000 111**0**	**= 14**
127	= 0111 1111	= 0111 111**0**	**= 126**
152	= 1001 1000	= 1001 100**1**	= **153**

Piksel (3, 1)

249	= 1111 1001	= 1111 100**0**	**= 248**
25	= 0001 1001	= 0001 100**0**	**= 24**
181	= 1011 0101	= 1011 010**1**	= **181**

Piksel (3, 2)

246	= 1111 0110	= 1111 011**0**	**= 246**
232	= 1110 1000	= 1110 100**1**	**= 233**
108	= 0110 1100	= 0110 110**0**	**= 108**

Piksel (3, 3)

171	= 1010 1011	= 1010 101**1**	**= 171**
101	= 0110 0101	= 0110 010**0**	**= 100**
106	= 0110 1010	= 0110 101**0**	**= 106**

Piksel (3, 4)

191	= 1011 1111	= 1011 111**0**	**= 190**
72	= 0100 1000	= 0100 100**1**	**= 73**
242	= 1111 0010	= 1111 001**0**	**= 242**

Piksel (3, 5)

239	= 1110 1111	= 1110 111**0**	**= 238**
151	= 1001 0111	= 1001 011**0**	**= 150**
85	= 0101 0101	= 0101 010**0**	**= 84**

Piksel (4, 1)

161	= 1001 0111	= 1001 011**1**	**= 151**
193	= 1100 0001	= 1100 000**0**	**= 192**
14	= 0000 1110	= 0000 1110	**= 14**

5. Tampilkan citra stego yang diperoleh

Gambar 4.5 Citra Stego

Tabel 4.2 Citra Sampul Yang Telah Disisipi Pesan

216,12,46	**238,106,49**	**60,44**,1	114,120,244	186,146,138
94,183,178	**208,136,12**	**141,253,82**	**113,4,164**	**14,126,153**
248,24,181	**246,233,108**	**171,100,106**	**190,73,242**	**238,150,84**
151,192,14	254,246,72	91,1,23	242,210,201	193,39,144
81,145,192	61,116,35	150,168,203	129,198,122	176,108,120

Setelah diperoleh citra stego, maka file citra stego ini akan dikirimkan kepada penerima. Kemudian, penerima akan mengekstraksi data dari citra stego tersebut. Proses kerja dari ekstraksi data dari citra stego dapat dirincikan sebagai berikut:

1. Input citra stego

Gambar 4.6 Citra Stego Yang Akan Diekstraksi

Tabel 4.3 Citra Sampul Yang Akan Diekstraksi

216,12,46	**238,106,49**	**60,44**,1	114,120,244	186,146,138
94,183,178	**208,136,12**	**141,253,82**	**113,4,164**	**14,126,153**
248,24,181	**246,233,108**	**171,100,106**	**190,73,242**	**238,150,84**
151,192,14	254,246,72	91,1,23	242,210,201	193,39,144
81,145,192	61,116,35	150,168,203	129,198,122	176,108,120

2. Ekstraksi panjang *ciphertext* dari baris 1 elemen warna RGB piksel citra stego.

   Piksel (1, 1)

   216 = 1101 1000

   12 = 0000 1100

   46 = 0010 1110

   Piksel (1, 2)

   238 = 1110 1110

   106 = 0110 1010

   49 = 0011 0001

   Piksel (1, 3)

   60 = 0011 1100

   44 = 0010 1100

Bit terekstrak = 0000 0100 = 4, berarti panjang *ciphertext* adalah 4 karakter. Hal ini berarti bahwa harus diekstrak 4 * 8 = 32 bit dari citra stego.

3. Ekstrak bit *ciphertext* mulai dari baris 2 elemen warna RGB piksel citra stego.

   Piksel (2, 1)

   94 = 0101 1110

   183 = 1011 0111

   178 = 1011 0010

   Piksel (2, 2)

   208 = 1101 0000

   136 = 1000 1000

   12 = 0000 1100

   Piksel (2, 3)

   141 = 1000 1101

   253 = 1111 1101

   82 = 0101 0010

   Piksel (2, 4)

   113 = 0111 0001

   4 = 0000 0100

   164 = 1010 0100

   Piksel (2, 5)

   14 = 0000 1110

   126 = 0111 1110

   152 = 1001 1001

   Piksel (3, 1)

   248 = 1111 1000

   24 = 0001 1000

   181 = 1011 0101

   Piksel (3, 2)

   246 = 1111 011**0**

   233 = 1110 100**1**

   108 = 0110 110**0**

   Piksel (3, 3)

   171 = 1010 101**1**

   100 = 0110 010**0**

   106 = 0110 101**0**

Piksel (3, 4)

190 = 1011 1110

73 = 0100 1001

242 = 1111 0010

Piksel (3, 5)

238 = 1110 1110

150 = 1001 0110

84 = 0101 0100

Piksel (4, 1)

151 = 1001 0111

192 = 1100 0000

Bit terekstrak: **0100 0011 0100 0010 0101 0100 0100 0010**

Ciphertext yang diperoleh:

0100 0011 = 67 = **C**

0100 0010 = 66 = **B**

0101 0100 = 84 = **T**

0100 0010 = 66 = **B**

Setelah diperoleh *ciphertext* terekstrak, maka proses akan diakhiri dengan mendekripsi *ciphertext* tersebut. Proses dekripsi dengan menggunakan metode *Playfair Cipher* dapat dirincikan sebagai berikut:

1. *Ciphertext* = **CBTB**
2. Kunci rahasia = **TES**
3. *Ciphertext* akan disusun menjadi bigram : **CB TB.**
4. Buat matriks kunci dengan ukuran 6 x 6, dengan cara urutkan sesuai dengan kemunculan huruf kunci, kemudian urutkan huruf belum terpakai pada kunci.

T	E	S	A	B	C
D	F	G	H	I	K
L	M	N	O	P	Q
R	U	V	W	X	Y
Z	␣	0	1	2	3
4	5	6	7	8	9

Gambar 4.7 Matriks Kunci

5. Lakukan deskripsi dari cipherteks “CB” dan menghasilkan plainteks “BA”.

T	E	S	A	**B**	**C**
D	F	G	H	I	K
L	M	N	O	P	Q
R	U	V	W	X	Y
Z	␣	0	1	2	3
4	5	6	7	8	9

T	E	S	**A**	**B**	C
D	F	G	H	I	K
L	M	N	O	P	Q
R	U	V	W	X	Y
Z	␣	0	1	2	3
4	5	6	7	8	9

Gambar 4.8 Matriks Deskripsi “CB”

6. Lakukan enkripsi dari cipherteks “TB” dan menghasilkan plainteks “CA”.

T	E	S	A	**B**	C
D	F	G	H	I	K
L	M	N	O	P	Q
R	U	V	W	X	Y
Z	␣	0	1	2	3
4	5	6	7	8	9

T	E	S	**A**	B	**C**
D	F	G	H	I	K
L	M	N	O	P	Q
R	U	V	W	X	Y
Z	␣	0	1	2	3
4	5	6	7	8	9

Gambar 4.9 Matriks Deskripsi “TB

7. Plaintext yang diperoleh = **BACA**

4.2 IMPLEMENTASI PROGRAM

Akan digunakan program berbahasa python untuk menyelesaikan permasalahan yang ada, dimiliki file gambar berformat .jpg yang akan disisipi pesan rahasia dengan kunci random sesuai dengan keinginan user, lalu pesan tersebut disisipkan ke gambar dan akan dibuat gambar baru dengan format yang sama yaitu .jpg, untuk script dari programnya dapat dilihat pada bagian Lampiran.

BAB V
PENUTUP

5.1 KESIMPULAN

Setelah menyelesaikan proses manual dan program, penulis dapat menarik beberapa kesimpulan sebagai berikut:

1. Dapat digunakan untuk menyembunyikan pesan rahasia pada citra digital
2. Panjangnya pesan yang dapat disisipkan tergantung pada ukuran citra digital yang digunakan.
3. Perbadaan warna citra input dan citra hasil juga tidak kelihatan jelas.
4. Hasil *output* citra digital hanya berupa citra berformat BMP, karena proses penyimpnan data ke bentuk JPG akan mengubah warna piksel citra digital sehingga informasi yang disisipkan menjadi rusak atau hilang.
5. Pesan asli yang didapat dari proses deskripsi akan berubah menjadi huruf kapital, hal ini terjadi akibat amtriks kunci yang digunakan adalah huruf kapital.

5.2 SARAN

Beberapa saran yang berguna untuk pengembangan selanjutnya:

1. Dapat menambah fitur tutorial yang mampu menjelaskan prosedur kerja dari algoritma yang dibahas secara terperinci.
2. Pengembangan selanjutnya dapat membandingkan algoritma steganografi yang dibahas dengan algoritma lain sejenis untuk mengetahui kelebihan dan kekurangan algoritma yang dibuat

DAFTAR PUSTAKA

Furqan, Mhd., Sriani,. Sari, Indah Eka Yulia,. *Penerapan Metode Otsu dalam Melakukan Segmentasi Citra Pada Citra Naskah Arab*. Dalam jurnal*Matrik: Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer* ISSN: 2476-9843 *Vol.20, No.1*.

Hafiz, Aliy. 2019. *Steganografi Berbasis Citra Digital Untuk Menyembunyikan Data Menggunakan Metode Least Significant Bit (LSB)*, dalam jurnal *Jurnal Cendikia Vol. XVII.*

Hermawati, Fajar Astuti. 2013 .*Pengolahan Citra Digital Konsep & Teori*. CV. Andi Offset :Yogyakarta.

Indrajit, Richardus Eka. 2014. *Konsep dan Strategi Keamanan Informasi Di Dunia Cyber*. Graha Ilmu : Yogyakarta.

Laoli Desimer, dkk. 2020. *Penerapan Algoritma Hill Cipher Dan Least Significant Bit (LSB) Untuk Pengamanan Pesan Pada Citra Digital.Jurnal JISKa Vol. 4, No.3* . Medan: STMIK Pelita Nusantara.

Muljoto, dkk. 2017. *Pengolahan Citra Digital*. CV.Andi Offset : Yogyakarta. Munir, R. 2006. *Kriptografi, Cetakan Pertama*. Penerbit Informatika : Bandung. Murdowo, Sugeng. 2020 . *Manual Perhitungan Menggunakan Kriptografi Klasik*

Playfair Cipher. Semarang .*Jurnal INFOKAM Vol. XVI, No.1*.

Nasution, Yusuf Ramadhan,. Furqan, Mhd,. Sinaga, Meri. *Metode Spread Spectrum Dalam Pengamanan Data Teks Pada Citra Digital*. Dalam jurnal *Jurnal Sains Komputer & Informatika (J-Sakti)* ISSN: 2548-9771*Volume. 4, Nomer. 2*.

Sadikin, Rifki. 2012. *Kriptografi Untuk Keamanan Jaringan*. CV Andi Offset :Yogyakarta.

Setyaningsih, E. 2015. *Kriptografi & Implementasi Menggunakan Matlab*. CVAndi Offset: Yogyakarta.

Stalling, W.2010. *Cryptography and Network Security: Principles and Practice*.

5thedition: Practice Hall.

Simbolon, Ratna Wati. 2016. *Pengaman Transkrip Nilai Mahasiswa Menggunakan Kriptografi Playfair Cipher dan Steganografi Dengan Teknik Least Significant Bit (LSB). Jurnal Teknologi Informasi Dan Komunikasi Vol. 5, No.1.*

Sitorus, Michel . 2015. *Teknik Steganografi dengan Metode Least Significant Bit (LSB), Jurnal Ilmiah Fakultas Teknik LIMIT'S ISSN 0612- 1184* Vol.11, No.2.

Sumarno, dkk. 2018. *Pengamanan Berkas Dokumen Menggunakan Fungsi Algoritma Steganografi LSB. Jurnal Ilmu Komputer dan Informatika Vol.2, No. 01*. Pematang Siantar: AMIK Tunas Bangsa.

Wiyata. 2016. *Implementasi Steganografi Metode LSB Menggunakan Program PHP Untuk Keamanan Pesan Gambar*. *Jurnal ICT Learning Vol.2, No.2*. Jakarta Selatan: Universitas Budi Luhur.

LAMPIRAN

Lampiran 1. Program Enkripsi

```
import cv2
import numpy as np

def generate_playfair_matrix(key):
    alphabet = "ABCDEFGHIKLMNOPQRSTUVWXYZ"
    key = key.upper().replace("J", "I")
    key = "".join(sorted(set(key), key=key.index))
    combined_key = key + ''.join([char for char in alphabet if char not in
key])
    return [combined_key[i:i + 5] for i in range(0, 25, 5)]

def find_char_position(matrix, char):
    for i in range(5):
        for j in range(5):
            if matrix[i][j] == char:
                return i, j
    return None, None

def clean_text_for_playfair(text):
    text = text.upper().replace("J", "I")
       cleaned_text   =   "".join(char   for   char   in   text   if   char   in
"ABCDEFGHIKLMNOPQRSTUVWXYZ")
    cleaned_text += 'X' * (len(cleaned_text) % 2)
    return cleaned_text

def print_playfair_matrix(matrix):
    for row in matrix:
        print(row)

def playfair_encrypt(key, plaintext):
    matrix = generate_playfair_matrix(key)
    plaintext = clean_text_for_playfair(plaintext)
    ciphertext = []

    print_playfair_matrix(matrix)

    for i in range(0, len(plaintext), 2):
        char1, char2 = plaintext[i], plaintext[i + 1]
        print(f"Memproses karakter: {char1} dan {char2}")

    for i in range(0, len(plaintext), 2):
        char1, char2 = plaintext[i], plaintext[i + 1]

        position1 = find_char_position(matrix, char1)
```

```
        position2 = find_char_position(matrix, char2)

        if position1[0] is None or position1[1] is None or position2[0] is None or position2[1] is None:
            raise ValueError(f"Karakter {char1} atau {char2} tidak ditemukan dalam matriks!")

        row1, col1 = position1
        row2, col2 = position2

        if row1 == row2:
                    ciphertext.extend([matrix[row1][(col1 + 1) % 5], matrix[row2][(col2 + 1) % 5]])
        elif col1 == col2:
            ciphertext.extend([matrix[(row1 + 1) % 5][col1], matrix[(row2 + 1) % 5][col2]])
        else:
            ciphertext.extend([matrix[row1][col2], matrix[row2][col1]])

    return ''.join(ciphertext)

def text_to_binary(text):
    return ''.join(format(ord(char), '08b') for char in text)

def main():
    key = input("Masukkan kunci untuk enkripsi: ")
    plaintext = input("Masukkan pesan yang ingin Anda enkripsi: ")
    ciphertext = playfair_encrypt(key, plaintext)

    input_image_path = "c:\\Users\\KURO\\Desktop\\apkri\\example.jpg"
                                    output_image_path                    = "c:\\Users\\KURO\\Desktop\\apkri\\example_with_message.jpg"

    lsb_embed(ciphertext, input_image_path, output_image_path)

def lsb_embed(ciphertext, input_image_path, output_image_path):
    image = cv2.imread(input_image_path)

    if image is None:
            raise ValueError(f"File gambar {input_image_path} tidak dapat dibaca.")

    max_message_length = image.shape[0] * image.shape[1] * 3 // 8

    if len(ciphertext) > max_message_length:
        raise ValueError("Pesan terlalu panjang untuk disisipkan dalam citra ini.")
```

```
    binary_message = text_to_binary(ciphertext)
    message_index = 0

    for i in range(image.shape[0]):
        for j in range(image.shape[1]):
            for k in range(3):
                if message_index < len(binary_message):
                    pixel_value = image[i, j, k]
                                image[i,  j,  k]  =  pixel_value  &  0xFE  |
int(binary_message[message_index])
                    message_index += 1
                else:
                    break

    cv2.imwrite(output_image_path, image)
    print(f"Pesan berhasil disisipkan dalam {output_image_path}")

if __name__ == "__main__":
    main()
```

Lampiran 2. Program Dekripsi

```
import cv2
import numpy as np
import re

def validate_key(key):
    # Memastikan kunci hanya berisi alfabet kapital
    if not re.match("^[A-Z]+$", key):
        raise ValueError("Kunci harus berisi hanya alfabet kapital!")

def lsb_extract(image_path):
    image = cv2.imread(image_path)

    if image is None:
          raise ValueError(f"File gambar {image_path} tidak dapat dibaca.
Periksa path dan format file Anda.")

    extracted_binary = ""

    for i in range(image.shape[0]):
        for j in range(image.shape[1]):
            for k in range(3):  # Loop melalui channel warna (B, G, R)
                # Ambil bit paling tidak signifikan dari setiap piksel
                extracted_binary += str(image[i, j, k] & 1)

    # Memeriksa bahwa setiap karakter dalam extracted_binary adalah '0' atau
'1'
```

```
    if not all(bit in '01' for bit in extracted_binary):
        raise ValueError("String biner yang diekstrak tidak valid!")

    # Memastikan panjang string biner adalah kelipatan 8
    padding_length = 8 - (len(extracted_binary) % 8)
    extracted_binary += '0' * padding_length

    extracted_text = "".join([chr(int(extracted_binary[i:i+8], 2)) for i in
range(0, len(extracted_binary), 8)])

    return extracted_text

def generate_playfair_matrix(key):
    # Inisialisasi alfabet
    alphabet = "ABCDEFGHIKLMNOPQRSTUVWXYZ"

    # Konversi kunci ke huruf kapital dan hilangkan duplikasi
    key = key.upper().replace("J", "I")
    key = "".join(sorted(set(key), key=key.index))

    # Gabungkan kunci dengan alfabet tanpa duplikasi
    combined_key = key + ''.join([char for char in alphabet if char not in
key])

    # Buat matriks 5x5 untuk Playfair Cipher
    matrix = [combined_key[i:i + 5] for i in range(0, 25, 5)]

    return matrix

def find_position(matrix, char):
    for i in range(5):
        for j in range(5):
            if matrix[i][j] == char:
                return i, j
    # Jika karakter tidak ditemukan, kembalikan None
    return None

def is_valid_playfair_char(char):
   # Fungsi ini memeriksa apakah karakter adalah karakter valid untuk Playfair
Cipher.
    valid_chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    return char in valid_chars

def playfair_decrypt(key, ciphertext):
    matrix = generate_playfair_matrix(key)

    if len(ciphertext) % 2 != 0:
        ciphertext += 'X'
```

```
    plaintext = ""

    pairs = [ciphertext[i:i+2] for i in range(0, len(ciphertext), 2)]

    for pair in pairs:
        char1, char2 = pair[0], pair[1]

        # Mengganti 'J' dengan 'I' saat mendekripsi
        if char1 == 'J':
            char1 = 'I'
        if char2 == 'J':
            char2 = 'I'

        # Periksa apakah kedua karakter valid
                        if     not     (is_valid_playfair_char(char1)     and
is_valid_playfair_char(char2)):
            continue  # Abaikan pasangan ini jika salah satu karakter tidak
valid

        position1 = find_position(matrix, char1)
        position2 = find_position(matrix, char2)

        if position1 is None or position2 is None:
            raise ValueError(f"Karakter {char1} atau {char2} tidak ditemukan
dalam matriks!")

        row1, col1 = position1
        row2, col2 = position2

        if col1 == col2:
            plaintext += matrix[(row1 - 1) % 5][col1]
            plaintext += matrix[(row2 - 1) % 5][col2]
        else:
            plaintext += matrix[row1][col2]
            plaintext += matrix[row2][col1]

    return plaintext

def main():
    key = input("Masukkan kunci untuk dekripsi: ").upper()

    try:
        # Validasi kunci sebelum digunakan
        validate_key(key)
    except ValueError as e:
        print(e)
        return  # Keluar dari program jika kunci tidak valid
```

```
                                        stego_image_path                    =
"c:\\Users\\KURO\\Desktop\\apkri\\example_with_message.jpg"

    try:
        extracted_message = lsb_extract(stego_image_path)

        if not extracted_message:
            print("Tidak ada pesan yang diekstrak dari gambar.")
            return  # Keluar dari program jika tidak ada pesan yang diekstrak

        plaintext = playfair_decrypt(key, extracted_message)
        print(f"Pesan yang diekstrak dari citra adalah: {plaintext}")

    except ValueError as e:
        print(f"Kesalahan dalam proses dekripsi: {e}")
        return  # Keluar dari program jika ada kesalahan

if __name__ == "__main__":
    main()
```