STEGANOGRAFI *CIPHERTEXT* AES 256 PADA CITRA DIGITAL MENGGUNAKAN METODE *LEAST SIGNIFICANT BIT* (LSB)

SKRIPSI

Untuk memenuhi sebagian persyaratan

mencapai gelar Sarjana Komputer

Disusun oleh :

ABHIMATA AR RASYIID

NIM. 0710963026

PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2013

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

LEMBAR PERSETUJUAN

STEGANOGRAFI *CIPHERTEXT* AES 256 PADA CITRA DIGITAL MENGGUNAKAN METODE *LEAST SIGNIFICANT BIT* (LSB)

SKRIPSI

Disusun oleh :

ABHIMATA AR RASYIID

NIM. 0710963026

Telah diperiksa dan disetujui oleh :

Pembimbing I,	Pembimbing II,
Edy Santoso, Ssi., M.Kom	**Nurul Hidayat, SPd., M.Sc**
NIP. 197404142003121004	**NIP.196804302002121001**

LEMBAR PENGESAHAN

STEGANOGRAFI *CIPHERTEXT* AES 256 PADA CITRA DIGITAL MENGGUNAKAN METODE *LEAST SIGNIFICANT BIT* (LSB)

SKRIPSI

Untuk memenuhi sebagian persyaratan

mencapai gelar Sarjana Komputer

Disusun oleh :

ABHIMATA AR RASYIID

NIM. 0710963026

Skripsi ini telah diuji dan dinyatakan lulus tanggal 7 Juni 2013

Penguji I	Penguji II
Drs. Marji, MT	**Ahmad Afif S., S.Si., M.Kom**
NIP. 196708011992031001	**NIK. 82062316110425**

Penguji III

Imam Cholissodin, S.Si., M.Kom
NIK. 85071916110422

Mengetahui
Ketua Program Studi Teknik Informatika

Drs. Marji, MT
NIP. 196708011992031001

PERNYATAAN ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata didalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, Juni 2013

Mahasiswa,

Abhimata Ar Rasyiid

NIM 0710963026

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah SWT yang telah melimpahkan segala Rahmat, Karunia dan Hidayah-Nya sehingga Penulis dapat menyelesaikan skripsi dengan judul: ”**Steganografi *Chipertext* AES 256 Pada Citra Digital Menggunakan Metode *Least Significant Bit* (LSB)**”

Skripsi ini diajukan sebagai syarat ujian seminar skripsi dalam rangka untuk memperoleh gelar Sarjana Komputer di Fakultas PTIIK, Program Studi Ilmu Komputer, Universitas Brawijaya Malang. Atas terselesaikannya skripsi ini, Penulis mengucapkan terima kasih kepada :

1. Edy Santoso, SSi., M.Kom., selaku Dosen Pembimbing Skripsi I.
2. Nurul Hidayat, S.Pd., M.Sc., selaku Dosen Pembimbing Skripsi II.
3. Drs. Marji, MT., selaku Ketua Program Studi Teknik Informatika Program Teknologi Informasi & Ilmu Komputer Universitas Brawijaya.
4. Bayu Rahayudi, ST., MM., selaku Dosen Penasehat Akademik.
5. Ir. Sutrisno, MT., selaku Ketua Program Teknologi Informasi & Ilmu Komputer Universitas Brawijaya.
6. Segenap Bapak dan Ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada Penulis selama menempuh pendidikan di Program Teknologi Informasi & Ilmu Komputer Universitas Brawijaya.
7. Segenap staff dan karyawan di Program Studi Teknologi Informasi & Ilmu Komputer Universitas Brawijaya yang telah banyak membantu Penulis dalam pelaksanaan penyusunan skripsi ini.
8. Orang tua Penulis dan saudara-saudaraku atas segala dukungan materi dan doa restunya kepada Penulis.

9. Rekan-rekan Program Studi Teknik Informatika yang telah memberikan dukungannya kepada penulis.
11. IceFrog@gmail.com, yang telah memberi kreativitas dan menjalani malam-malam bersama skripsi yang telah disusun ini.
12. Dr. Anita Rahmawati Sholeh Putri, yang telah memberi dukungan baik dalam bentuk material maupun non material demi terselesaikannya skripsi ini.
13. Semua pihak yang telah membantu terselesaikannya skripsi ini yang tidak dapat kami sebutkan satu per satu.

Penulis menyadari bahwa skripsi ini tentunya tidak terlepas dari berbagai kekurangan dan kesalahan. Oleh karena itu, segala kritik dan saran yang bersifat membangun sangat Penulis harapkan dari berbagai pihak demi penyempurnaan penulisan skripsi ini.

Akhirnya penulis berharap agar skripsi ini dapat memberikan sumbangan dan manfaat bagi semua pihak yang berkepentingan.

Malang, Juni 2013

Abhimata Ar Rasyiid

Penulis

ABSTRAK

Abhimata Ar Rasyiid. 2013. : Steganografi *Chipertext* AES 256 Pada Citra Digital Menggunakan Metode *Least Significant Bit* (LSB)

Dosen Pembimbing : Edy Santoso, Ssi., M.Kom., dan Nurul Hidayat, SPd., M.Sc

Teknologi sekarang ini memudahkan manusia untuk melakukan pertukaran data atau informasi dan secara tidak langsung kebutuhan akan keamanan data yang dipertukarkan semakin meningkat. Seiring dengan perkembangan dan kemajuan teknologi teknik pengamanan data dengan menggunakan kriptografi maupun steganografi saja tidak dapat memenuhi kebutuhan akan keamanan data yang bersifat rahasia. Permasalahan tersebut menjadi dasar pemikiran untuk menggabungkan dua teknik pengamanan data, yaitu teknik kriptografi dan steganografi. Dalam prosesnya pesan berupa teks akan disandikan terlebih dahulu menggunakan algoritma AES 256 kemudian disisipkan ke dalam media penampung berupa citra digital dengan format bitmap 24- bit menggunakan algoritma LSB.

AES merupakan standar algoritma kriptografi terbaru yang dipublikasikan oleh NIST (National Institute of Standard and Technology) sebagai pengganti algoritma DES (Data Encryption Standard) sedangkan LSB digunakan karena tidak menimbulkan penurunan kualitas citra secara kasat mata.

Setelah dilakukan uji coba didapatkan teknik pengamanan data steganografi melalui kriptografi dapat diimplementasikan pada sebuah perangkat lunak dan kualitas citra yang dihasilkan relatif sama dengan citra aslinya jika dilihat secara kasat mata, hal ini juga ditunjukkan dari nilai PSNR yang relatif tinggi diatas 50 db. Faktor yang mempengaruhi kualitas citra yang dihasilkan adalah persentase jumlah bit yang berubah, semakin sedikit bit yang berubah akan semakin mirip dengan citra aslinya. Namun citra steganografi tidak tahan jika dilakukan manipulasi citra

Kata Kunci : kriptografi, steganografi, AES 256, LSB

repository.ub.ac.id

ABSTRACT

Abhimata Ar Rasyiid. 2013.: Steganography Ciphertext AES 256 At Image Using Method Least Significant Bit (LSB)

Advisor : Edy Santoso, Ssi., M.Kom., dan Nurul Hidayat, SPd., M.Sc

Technology is now this makes it easy humans easy to perform exchange of data or information and are not directly the need security of data that is exchanged increasing. Along with the development and technological advancements technique of securing data by using cryptography nor the steganography alone can not meet needs security of data that are confidential. Those problems becomes the basis of thought for combine the two techniques securing data, namely technique of cryptography and steganography. In the process messages in the form a text encoded first using a algorithm AES 256 then pasted into inside media receptacle the form of digital imagery with the format of bitmap 24 - bits using algorithm LSB.

AES is a standard cryptographic algorithms newest which published by the NIST (National Institute of Standard and Technology) as a substitute algorithm DES (Data Encryption Standard) whereas LSB used because not pose decline in image-quality by naked eye.

After the conducted trials obtained technique of securing data steganography through cryptographic can be implemented on a software and quality of the imagery the resulting relatively the same with the image of originals if viewed in naked eye, this case also indicated from the value of PSNR who relatively high above 50 db. Factors that affect the quality image of the resulting is the percentage number of bits that changed, increasingly slightly bits which changed will increasingly resemble with the image of its original. But stegoimage can’t endure against image manipulation.

Words Key: cryptography, steganography, AES 256, LSB

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi yang begitu pesat memungkinkan manusia untuk dapat berkomunikasi dan saling bertukar data atau informasi secara jarak jauh. Seiring dengan kemudahan tersebut kebutuhan akan keamanan terhadap kerahasiaan informasi yang dipertukarkan semakin meningkat. Begitu banyak pengguna seperti departemen pertahanan, perusahaan atau individu–individu tidak ingin informasi yang disampaikannya diketahui oleh orang lain. Oleh karena itu dikembangkanlah cabang ilmu yang mempelajari tentang cara–cara pengamanan data.

Salah satu cara dalam melakukan pengamanan data adalah dengan menggunakan kriptografi. Istilah kriptografi sudah sangat dikenal dalam dunia pengamanan data. Kriptografi merupakan ilmu sekaligus seni untuk menjaga keamanan pesan [SCH-96]. Kriptografi berbasis pada algoritma pengkodean data informasi yang mendukung kebutuhan dari dua aspek keamanan informasi, yaitu *secrecy* (perlindungan terhadap kerahasiaan data informasi) dan *authenticity* (perlindungan terhadap pemalsuan dan pengubahan informasi yang tidak diinginkan) [BUD-10].

Saat ini, AES (*Advanced Encryption Standard*) merupakan salah satu algoritma kriptografi yang cukup aman untuk melindungi data atau informasi yang bersifat rahasia. Pada tahun 2001, AES digunakan sebagai standar algoritma kriptografi terbaru yang dipublikasikan oleh NIST (*National Institute of Standard and Technology*) sebagai pengganti algoritma DES (*Data Encryption Standard*) yang sudah berakhir masa penggunaannya. Algoritma AES adalah algoritma kriptografi yang dapat mengenkripsi dan mendekripsi data dengan panjang kunci yang bervariasi, yaitu 128 bit, 192 bit, dan 256 bit [FED-01].

Pengamanan data dengan kriptografi memiliki kelemahan pada bentuk pesan yang tersandi seperti simbol atau kode - kode aneh sehingga keberadaan

pesan dapat langsung terdeteksi oleh indera manusia dan akan membuat penasaran yang akhirnya akan berusaha untuk mengetahui kode - kode aneh tersebut.

Berbeda dengan kriptografi, teknik pengamanan data Steganografi adalah ilmu dan seni menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan tersebut tidak dapat diketahui. Dengan steganografi, pesan yang dikirim tidak menarik perhatian dan media penampung tidak menimbulkan kecurigaan. Steganografi membutuhkan dua properti, yaitu media penampung dan pesan rahasia. Media penampung yang umum digunakan adalah gambar, suara, video, atau teks. Pesan yang disembunyikan dapat berupa sebuah artikel, gambar, daftar barang, kode program, atau pesan lain [MUN-06].

Metode steganografi yang paling sederhana dan paling mudah diimplementasikan adalah metode LSB (*Least Significant Bit*). Dan media penampung yang umumnya digunakan adalah citra digital [DUN-96]. Karena keterbatasan kemampuan manusia, maka citra digital yang telah disisipi pesan dengan metode LSB tidak dapat diketahui oleh mata manusia.

Seiring dengan perkembangan dan kemajuan teknologi teknik pengamanan data dengan menggunakan kriptografi maupun steganografi saja tidak dapat memenuhi kebutuhan akan keamanan data yang bersifat rahasia. Permasalahan tersebut menjadi dasar pemikiran untuk menggabungkan dua teknik pengamanan data, yaitu teknik kriptografi dan steganografi. Dalam implementasinya pesan akan disandikan terlebih dahulu untuk kemudian disisipkan ke dalam media penampung sehingga keberadaan pesan tidak dapat terdeteksi oleh indera manusia. Teknik penggabungan ini pernah dilakukan dalam penelitian sebelumnya namun tidak menggunakan algoritma AES 256 dan LSB melainkan menggunakan algoritma 3DES dan LSB [ELV-10]. Untuk penelitian dengan algoritma AES 256 juga pernah dilakukan akan tetapi tidak digabungkan dengan algoritma LSB hanya berfokus metode enkripsi AES 256 saja [PRI-12]. Yang membedakan penelitian ini dengan penelitian sebelumnya adalah penggabungan teknik AES 256 dan LSB, dengan penggabungan kedua teknik ini diharapkan tingkat keamanan data semakin tinggi sehingga dapat memenuhi kebutuhan akan keamanan data.

Berdasarkan beberapa hal - hal tersebut, maka judul yang diambil dalam tugas akhir ini adalah **"Steganografi Untuk Menyisipkan Teks Hasil Enkripsi AES 256 Pada Citra Digital Menggunakan Metode LSB".**

1.2 Rumusan Masalah

Rumusan masalah pada skripsi ini adalah sebagai berikut:

1. Bagaimana mengimplementasikan steganografi *ciphertext* AES 256 pada citra digital menggunakan metode *Least Significant Bit* (LSB).
2. Berapa nilai PSNR citra digital sesudah dilakukan penyisipan pesan rahasia dengan menggunakan metode steganografi.
3. Bagaimana ketahanan citra steganografi terhadap manipulasi citra.

1.3 Batasan Masalah

Agar pembahasan tidak melebar, maka batasan masalah dalam skripsi ini adalah:

1. File yang disisipkan berupa teks dengan format (.txt)
2. Jumlah karakter file teks yang akan disisipkan tidak melebihi kapasitas citra penampung.
3. Data yang digunakan berupa citra digital dengan *format bitmap* (*.bmp) 24 bit.
4. Panjang kunci maksimal yang digunakan untuk proses enkripsi adalah 32 karakter.
5. Kunci (key) yang digunakan untuk enkripsi adalah karakter ASCII.

1.4 Tujuan

Tujuan yang ingin dicapai dari skripsi ini adalah:

1. Mengimplementasikan teknik steganografi *chipertext* AES 256 pada citra digital menggunakan metode *Least Significant Bit* (LSB).
2. Menguji citra digital hasil penyisipan pesan rahasia dengan metode steganografi melalui nilai PSNR.
3. Mengetahui ketahanan citra steganografi terhadap manipulasi citra.

1.5 Manfaat

Dengan adanya teknik pengamanan data steganografi melalui kriptografi ini diharapkan mampu memenuhi kebutuhan akan keamanan data atau informasi rahasia yang akan dipertukarkan.

1.6 Sistematika Pembahasan

Sistematika penulisan skripsi terdiri dari 7 bab, yaitu:

BAB I PENDAHULUAN

Bab ini menguraikan latar belakang masalah yang akan dibahas, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan.

BAB II KAJIAN PUSTAKA DAN DASAR TEORI

Bab ini menjelaskan teori-teori kriptografi, steganografi, algoritma AES 256, Algoritma LSB dan teori lain yang merupakan konsep dasar penelitian.

BAB III METODE PENELITIAN

Bab ini menjelaskan metode yang dipakai dalam menyeselsaikan penelitian ini.

BAB IV PERANCANGAN

Bab ini menjelaskan tahapan bagaimana sistem steganografi *ciphertext* AES 256 pada citra digital menggunakan metode *Least Significant Bit* (LSB) dibangun.

BAB V IMPLEMENTASI

Bab ini membahas implementasi dari sistem yang telah dibuat.

BAB VI PENGUJIAN DAN ANALISIS

Bab ini menjelaskan tentang uji coba yang dilakukan beserta pembahasannya.

BAB VII PENUTUP

Pada bab ini akan menujukkan kesimpulan dan saran yang dapat digunakan untuk pengembangan berikutnya.

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

Berdasarkan judul penelitian ini Steganografi *Ciphertext* AES 256 pada Citra Digital Menggunakan Metode *Least Significant Bit* (LSB) penulis menemukan beberapa pustaka yang relevan untuk mendukung penelitian ini antara lain :

[YUN-09] menyatakan Kriptografi merupakan salah satu solusi atau metode pengamanan data yang tepat untuk menjaga kerahasiaan dan keaslian data, serta dapat meningkatkan aspek keamanan suatu data atau informasi. Metode ini bertujuan agar informasi yang bersifat rahasia dan dikirim melalui suatu jaringan, seperti LAN atau Internet, tidak dapat diketahui atau dimanfaatkan oleh orang atau pihak yang tidak berkepentingan. Kriptografi mendukung kebutuhan dua aspek keamanan informasi, yaitu perlindungan terhadap kerahasiaan data informasi dan perlindungan terhadap pemalsuan dan pengubahan informasi yang tidak diinginkan. Hasil penelitian menunjukkan bahwa algoritma AES dengan panjang kunci 256 *bit* dapat menyandikan isi suatu file sehingga dapat mengamankan file tersebut. Ukuran file enkripsi akan bertambah 11 *bytes* dari file asli karena adanya proses penambahan *header* yang berisi informasi ekstensi file. Dalam pengembangan sistem berikutnya diharapkan sistem dapat mempunyai fasilitas untuk menyembunyikan *folder* yang digunakan untuk menyimpan file enkripsi maupun file dekripsi.

[NIK-08] menyatakan pengamanan pada saat pengiriman suatu informasi yang bersifat rahasia perlu dilakukan untuk menghindari terjadinya penyalahgunaan, salah satu caranya dengan menyisipkan informasi rahasia tersebut pada media tertentu. Permasalahan yang timbul akibat dari penyisipan adalah masih terlihatnya perubahan yang mencolok. Pada penelitian tersebut pesan atau informasi rahasia disisipkan kedalam suatu citra digital yang diam (tidak bergerak). Teknik penyisipannya menggunakan pendekatan *Least Significant Bit* (LSB). Pendekatan LSB ini hanya menggunakan satu komponen warna sehingga pesan dapat disembunyikan secara efektif. Dari hasil perhitungan *Peak Signal to Noise Ratio* (PSNR), citra digital yang telah disisipi hanya

mengalami perubahan yang rendah, hal ini dibuktikan melalui besar rata-rata nilai PSNR sebesar 99,9973% yang artinya bahwa hanya terjadi kerusakan citra sebesar 0,0027% dibandingkan citra digital asli.

Dari penelitian-penelitian yang dijelaskan tersebut hanya berfokus pada algoritma AES 256 saja atau algoritma LSB saja, tidak ada proses penggabungan dari kedua algoritma tersebut sehingga yang membedakan penelitian ini dengan penelitian sebelumnya adalah dalam penelitian ini digabungkan dua buah algoritma yaitu algoritma AES 256 dan algoritma LSB yang mana dengan penggabungan ini diharapan mampu memenuhi kebutuhan akan keamanan data atau informasi yang akan dipertukarkan.

2.1 Kriptografi

Kriptografi (*cryptography*) berasal dari Bahasa Yunani: ”*cryptos*” artinya ”*secret*” (rahasia), sedangkan ”*graphein*” artinya ”*writing*” (tulisan). Jadi, kriptografi berarti ”*secret writing*” (tulisan rahasia) [MUN-06].

Kriptografi (*cryptography*) merupakan ilmu dan seni untuk menjaga pesan agar aman. “*Crypto*” berarti “*secret*” (rahasia) dan “*graphy*” berarti “*writing*” (tulisan). Pesan atau informasi yang dapat dibaca disebut sebagai *plaintext*. Teknik yang membuat pesan menjadi tidak dapat dibaca adalah enkripsi. Proses enkripsi membuat pesan yang awalnya dapat dibaca menjadi tidak dapat dibaca, pesan hasil dari enkripsi tersebut dikenal dengan istilah *ciphertext*. Deskripsi merupakan proses kebalikan dari enkripsi, yaitu membuat *ciphertext* menjadi *plaintext* [RAH-01].

Kriptografi merupakan studi teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, autentikasi. Teknik ini digunakan untuk mengubah data ke dalam kode-kode tertentu, dengan tujuan informasi yang disimpan atau ditransmisikan melalui jaringan yang tidak aman (misalnya internet) tidak dapat dibaca oleh siapa pun kecuali orang-orang yang berhak [MEN-96].

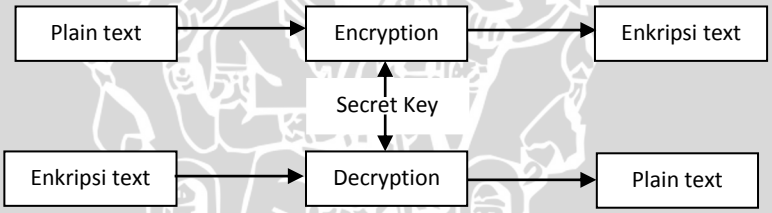
2.2 Algoritma Kriptografi

Terdapat dua jenis algoritma kriptografi berdasar jenis kuncinya, yaitu:

a. Algoritma Simetris

Algoritma simetris disebut juga sebagai algoritma konvensional, yaitu algoritma yang menggunakan kunci yang sama untuk proses enkripsi dan deskripsinya. Keamanan algoritma simetris tergantung pada kuncinya. Algoritma simetris sering juga disebut algoritma kunci rahasia, algoritma kunci tunggal atau algoritma satu kunci. Dua kategori yang termasuk pada algoritma simetris ini adalah algoritma *block cipher* dan *stream cipher* [KUR-04].

Pada Gambar 2.1. diperlihatkan skema algoritma simetri yang hanya membutuhkan satu buah kunci yang sama.



Gambar 2.1 Algoritma Kriptografi Simetris

Kelebihan algoritma kriptografi simetris adalah:

1. Algoritma ini dirancang sehingga proses enkripsi/dekripsi membutuhkan waktu yang singkat.
2. Ukuran kunci relatif lebih pendek.
3. Algoritmanya bisa menghasilkan *cipher* yang lebih kuat.
4. Autentikasi pengiriman pesan langsung diketahui dari *ciphertext* yang diterima, karena kunci hanya diketahui oleh pengirim dan penerima pesan saja.

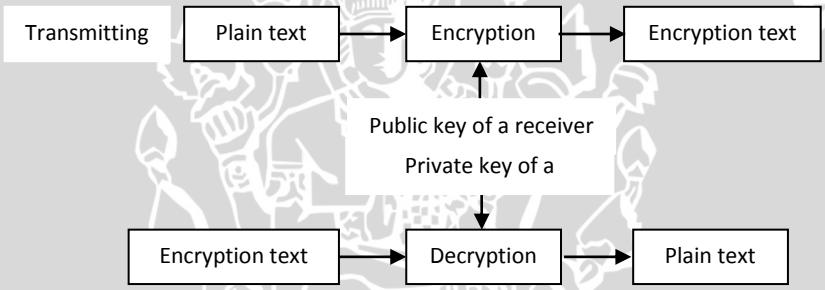
Kelemahan algoritma kriptografi simetris adalah:

1. Kunci harus dikirim melalui saluran yang aman. Kedua entitas yang berkomunikasi harus menjaga kerahasiaan kunci ini.

2. Kunci harus sering diubah, mungkin pada setiap sesi komunikasi [MUN-04].

b. Algoritma Asimetri

Algoritma asimetrik atau biasa disebut algoritma kunci publik dirancang sedemikian sehingga kunci yang digunakan untuk mengenkripsi dan mendekripsi berbeda. Sehingga kunci dekripsi tidak dapat dihitung dari kunci enkripsi. Algoritma tersebut disebut *public-key* karena kunci enkripsi dapat dibuat secara *public*. Orang asing dapat menggunakan kunci enkripsi tersebut untuk mengenkripsi sebuah pesan, tetapi hanya orang tertentu dengan kunci dekripsi sepadan dapat mendekripsi pesan tersebut. Dalam sistem ini kunci enkripsi sering disebut *public key* sedangkan key dekripsi sering disebut *private key* [KUR-04]. Pada Gambar 2.2 diperlihatkan skema algoritma asimetri yang menggunakan dua buah kunci.



Gambar 2.2 Algoritma Kriptografi Asimetris

Kelebihan algoritma kriptografi asimetri adalah:

1. Hanya *Private key* yang harus benar-benar rahasia/aman.
2. Sangat jarang untuk perlu merubah *public key* dan *private key*.

Kelemahan algoritma kriptografi asimetri adalah:

1. Ukuran kunci lebih besar dari pada algoritma kunci simetri.
2. Tidak adanya jaminan bahwa *public key* benar-benar aman [MUN-04].

2.2.1 AES (*Advanced Encryption Standard*)

Input dan *output* dari algoritma AES terdiri dari urutan data sebesar 128 *bit*. Urutan data yang sudah terbentuk dalam satu kelompok 128 *bit* tersebut disebut juga sebagai blok data atau *plaintext* yang nantinya akan dienkripsi menjadi *ciphertext*. *Cipher key* dari AES terdiri dari *key* dengan panjang 128 *bit*, 192 *bit*, atau 256 *bit*. Perbedaan panjang kunci akan mempengaruhi jumlah *round*

yang akan diimplementasikan pada algoritma AES ini. Pada Tabel 2.1 diperlihatkan jumlah *round* / putaran (Nr) yang harus diimplementasikan pada masing – masing panjang kunci [MUN-06].

Tabel 2.1 Perbandingan Jumlah Round dan *Key*.

	Jumlah Key (Nk)	Ukuran Block (Nb)	Jumlah Putaran (Nr)
AES – 128	4	4	10
AES – 192	6	4	12
AES – 256	8	4	14

Pada dasarnya, operasi AES dilakukan terhadap *array of byte* dua dimensi yang disebut dengan *state*. *State* mempunyai ukuran *NROWS* X *NCOLS*. Pada awal enkripsi, data masukan yang berupa in0, in2, in3, in4, in5, in6, in7, in8, in9, in10, in11, in12, in13, in14, in15 disalin ke dalam *array state*. *State* inilah yang nantinya dilakukan operasi enkripsi / dekripsi. Kemudian keluarannya akan ditampung ke dalam *array out*. Pada Gambar 2.3 diperlihatkan proses penyalinan dari *input bytes*, *state array*, dan *output bytes.*

In_0	In_4	In_8	In_{12}
In_1	In_5	In_9	In_{13}
In_2	In_6	In_{10}	In_{14}
In_3	In_7	In_{11}	In_{15}

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

Out_0	Out_4	Out_8	Out_{12}
Out_1	Out_5	Out_9	Out_{13}
Out_2	Out_6	Out_{10}	Out_{14}
Out_3	Out_7	Out_{11}	Out_{14}

Gambar 2.3 *Input Bytes*, *State Array*, dan *Output Bytes.*

Pada saat permulaan, *input bit* pertama kali akan disusun menjadi suatu *array byte* dimana panjang dari *array byte* yang digunakan pada AES adalah sepanjang 8 *bit* data. *Array byte* inilah yang nantinya akan dimasukkan atau di*copy* ke dalam *state* dengan urutan dimana r (*row*/baris) dan c (*column*/kolom) seperti yang diperlihatkan pada persamaan 2.1.

$$s[r,c] = in[r+4c] \text{ untuk } 0 \leq r < 4 \text{ dan } 0 \leq c < Nb \qquad \textbf{(2.1)}$$

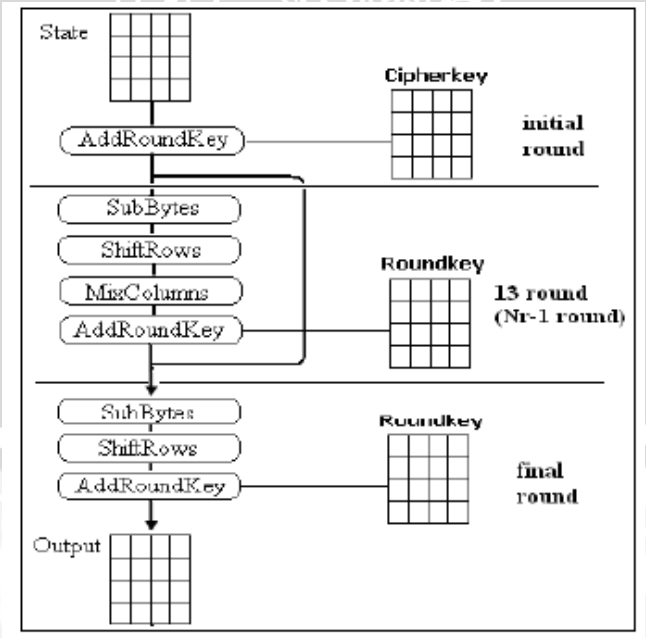
sedangkan dari state akan dicopy ke output dengan urutan sesuai dengan persamaan 2.2.

$$out[r+4c] = s[r,c] \text{ untuk } 0 \leq r < 4 \text{ dan } 0 \leq c < Nb \quad \textbf{(2.2)}$$

[MUN-06].

2.2.2 Proses Enkripsi AES

Proses enkripsi algoritma AES terdiri dari 4 jenis transformasi bytes, yaitu *SubBytes*, *ShiftRows*, *Mixcolumns*, dan *AddRoundKey*. Pada awal proses enkripsi, *input* yang telah di*copy*kan ke dalam *state* akan mengalami transformasi byte *AddRoundKey*. Setelah itu, *state* akan mengalami transformasi *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* secara berulang-ulang sebanyak Nr. Proses ini dalam algoritma AES disebut sebagai *round function*. *Round* yang terakhir agak berbeda dengan *round-round* sebelumnya dimana pada *round* terakhir, *state* tidak mengalami transformasi *MixColumns* [MUN-06]. Ilustrasi proses enkripsi AES dapat digambarkan seperti pada Gambar 2.4.



Gambar 2.4 Ilustrasi Proses Enkripsi AES.

2.2.2.1 Add round Key

Pada proses enkripsi dan dekripsi AES proses *AddRoundKey* sama, sebuah *round key* ditambahkan pada *state* dengan operasi XOR. Setiap *round key* terdiri dari Nb *word* dimana tiap *word* tersebut akan dijumlahkan dengan *word* atau kolom yang bersesuaian dari *state* seperti yang diperlihatkan pada persamaan 2.3.

$$\left[S'_{0,c}, S'_{1,c}, S'_{2,c}, S'_{3,c}\right] = [S_{0,c}, S_{1,c}, S_{2,c}, S_{3,c}] \oplus [W_{round*Nb+c}] untuk\ 0 \le c \le Nb \qquad \textbf{(2.3)}$$

[wi] adalah *word* dari *key* yang bersesuaian dimana i = round*Nb+c. Transformasi *AddRoundKey* pada proses enkripsi pertama kali pada *round* = 0 untuk *round* selanjutnya *round* = *round* + 1, pada proses dekripsi pertama kali pada *round* = 14 untuk *round* selanjutnya *round* = *round* – 1 [MUN-06].

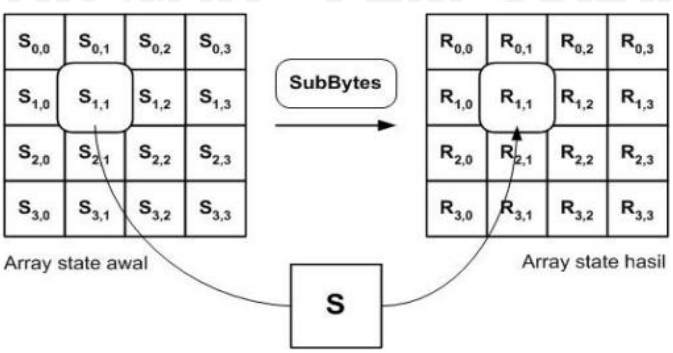
2.2.2.2 Sub Bytes

SubBytes merupakan transformasi *byte* dimana setiap elemen pada *state* akan dipetakan dengan menggunakan sebuah tabel substitusi (S-Box) yang ditunjukkan pada Tabel 2.2

Tabel 2.2 Subtitusi *Box*.

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

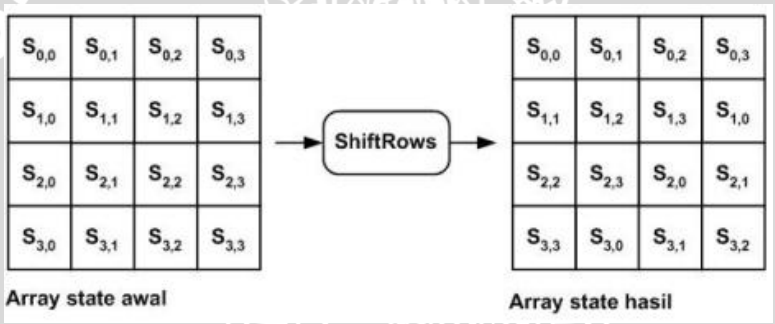
Untuk setiap *byte* pada *array state*, misalkan *S*[*r*, *c*] = *xy*, yang dalam hal ini *xy* adalah *digit* heksadesimal dari nilai *S*[*r*, *c*], maka nilai substitusinya, dinyatakan dengan *S*'[*r*, *c*], adalah elemen di dalam tabel subtitusi yang merupakan perpotongan baris *x* dengan kolom *y*. Pada Gambar 2.5 diperlihatkan pengaruh pemetaan *byte* pada setiap *byte* dalam *state* [MUN-06].



Gambar 2.5 Pengaruh Pemetaan pada Setiap *Byte* dalam *State*.

2.2.2.3 Shift Rows

Transformasi *Shiftrows* pada dasarnya adalah proses pergeseran *bit* dimana *bit* paling kiri akan dipindahkan menjadi *bit* paling kanan (rotasi *bit*) [MUN-06]. Proses pergeseran *Shiftrow* ditunjukkan dalam Gambar 2.6.



Gambar 2.6 Transformasi *ShiftRows*.

2.2.2.4 Mix Columns

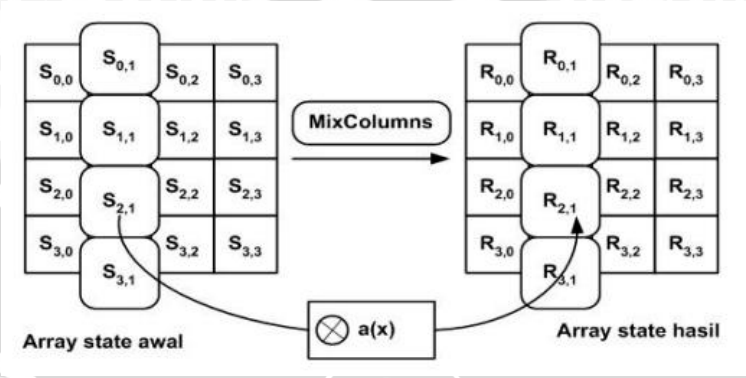
MixColumns mengoperasikan setiap elemen yang berada dalam satu kolom pada state. Secara lebih jelas, transformasi *mixcolumns* dapat dilihat pada perkalian matriks dalam persamaan 2.4.

$$\begin{bmatrix} S'0,c \\ S'1,c \\ S'2,c \\ S'3,c \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} = \begin{bmatrix} S0,c \\ S1,c \\ S2,c \\ S3,c \end{bmatrix} \text{ untuk } 0 \text{ d'' } c < Nb \quad \textbf{(2.4)}$$

Hasil dari perkalian matriks diatas dapat dianggap seperti perkalian yang diperlihatkan dalam persamaan 2.5.

$$
\begin{aligned}
S'0,c &= (\{02\}\cdot S'0,c)\oplus(\{03\}\cdot S'1,c)\oplus S'2,c\oplus S'3,c \\
S'1,c &= S'0,c\oplus(\{02\}\cdot S'1,c)\oplus(\{03\}\cdot S'2,c)\oplus S'2,c \\
S'2,c &= S'0,c\oplus S'1,c\oplus(\{02\}\cdot S'2,c)\oplus(\{03\}\cdot S'3,c) \\
S'3,c &= (\{03\}\cdot S'0,c)\oplus S'1,c\oplus S'2,c\oplus(\{02\}\cdot S'3,c)
\end{aligned}
\qquad \textbf{(2.5)}
$$

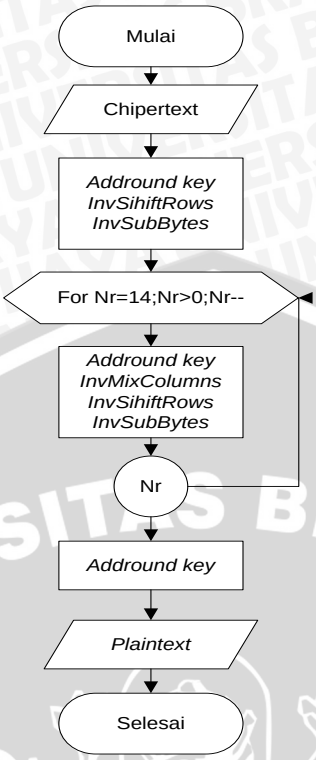
Ilustrasi operasi mixcolumns dapat dilihat pada Gambar 2.7



Gambar 2.7 Operasi *MixColumn*() pada *state* per kolom [MUN-06].

2.2.3 Proses Dekripsi AES

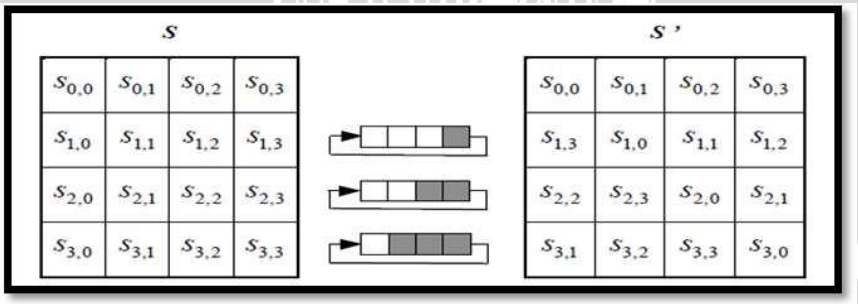
Transformasi *cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES [MUN-06]. Transformasi byte yang digunakan pada *invers cipher* adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*. Algoritma dekripsi dapat dilihat pada Gambar 2.8.



Gambar 2.8 Ilustrasi Proses Dekripsi AES.

2.2.3.1 Inv Shift Rows

InvShiftRows adalah transformasi byte yang berkebalikan dengan transformasi *ShiftRows*. Pada transformasi *InvShiftRows*, dilakukan pergeseran *bit* ke kanan sedangkan pada *ShiftRows* dilakukan pergeseran *bit* ke kiri [MUN-06]. Ilustrasi transformasi *InvShiftRows* terdapat pada Gambar 2.9.



Gambar 2.9 Transformasi *InvShiftRows*.

2.2.3.2 Inv Sub Bytes

InvSubBytes juga merupakan transformasi *bytes* yang berkebalikan dengan transformasi *SubBytes*. Pada *InvSubBytes*, tiap elemen pada state dipetakan

dengan menggunakan tabel *Inverse S-Box* [FED-01]. Tabel *Inverse S-Box* akan ditunjukkan dalam Tabel 2.3 .

Tabel 2.3 *Inverse S-Box.*

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

2.2.3.3 Inv Mix Columns

Setiap kolom dalam *state* dikalikan dengan matrik perkalian dalam AES. Perkalian dalam matrik dapat dituliskan seperti pada persamaan 2.6.

$$\begin{bmatrix} S'0,c \\ S'1,c \\ S'2,c \\ S'3,c \end{bmatrix} \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} = \begin{bmatrix} S0,c \\ S1,c \\ S2,c \\ S3,c \end{bmatrix} \quad \textbf{(2.6)}$$

Hasil dari perkalian dalam matrik diperlihatkan dalam persamaan 2.7.

$$\begin{aligned} S'0,c &= (\{0E\}\cdot S0,c)\oplus(\{0B\}\cdot S1,c)\oplus(\{0D\}\cdot S0,c)\oplus(\{09\}\cdot S3,c) \\ S'1,c &= (\{09\}\cdot S0,c)\oplus(\{0E\}\cdot S1,c)\oplus(\{0B\}\cdot S0,c)\oplus(\{0D\}\cdot S3,c) \\ S'2,c &= (\{0D\}\cdot S0,c)\oplus(\{09\}\cdot S1,c)\oplus(\{0E\}\cdot S0,c)\oplus(\{0B\}\cdot S3,c) \\ S'3,c &= (\{0B\}\cdot S0,c)\oplus(\{0D\}\cdot S1,c)\oplus(\{09\}\cdot S0,c)\oplus(\{0E\}\cdot S3,c) \end{aligned} \quad \textbf{(2.7)}$$

[MUN-06].

2.3 Steganografi

Kata steganografi (*steganography*) berasal dari bahasa Yunani *steganos*, yang artinya tersembunyi atau terselubung, dan *graphia* yang artinya menulis,

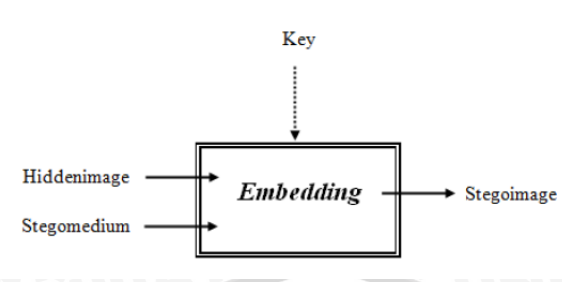
sehingga arti steganografi adalah "menulis (tulisan) terselubung" [CVE-04]. Dengan steganografi, dapat menyisipkan pesan rahasia ke dalam media lain dan mengirimkannya tanpa ada yang menyadari keberadaan pesan tersebut [KRE-04].

Steganografi adalah seni dan ilmu berkomunikasi dengan cara menyembunyikan keberadaan komunikasi itu. Berbeda dengan Kriptografi, di mana musuh diperbolehkan untuk mendeteksi, menangkal dan memodifikasi pesan tanpa bisa melanggar keamanan tempat tertentu yang dijamin oleh suatu *cryptosystem*, tujuan dari steganografi adalah untuk menyembunyikan pesan dalam pesan lainnya dengan cara yang tidak memungkinkan musuh untuk mendeteksi bahwa ada pesan kedua. Secara umum, teknik steganografi yang baik harus memiliki visual/*imperceptibility* statistik yang baik dan *payload* yang cukup [KEK-08].

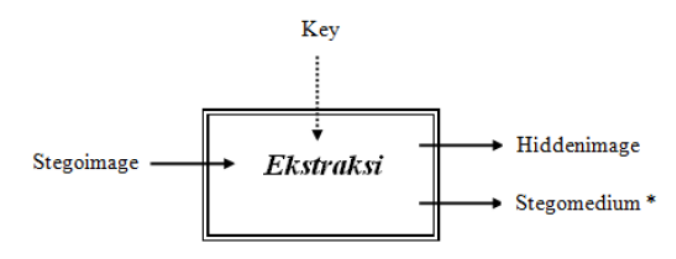
Steganografi membutuhkan dua media, yaitu penampung dan data yang akan disisipkan. Secara teori, semua berkas digital yang ada di dalam komputer dapat digunakan sebagai media penampung, misalnya citra berformat JPG, GIF, BMP, atau di dalam musik MP3, atau bahkan di dalam sebuah film dengan format WAV atau AVI. Semua dapat dijadikan tempat bersembunyi, asalkan berkas tersebut memiliki bit-bit yang tidak signifikan atau terdapat *redundant bits* yang dapat dimodifikasi. Setelah dimodifikasi, berkas media tersebut tidak akan terganggu fungsinya dan kualitasnya tidak akan jauh berbeda dengan aslinya.

2.3.1 Proses Steganografi

Secara umum, terdapat dua proses didalam steganografi. Yaitu proses *embedding* untuk menyembunyikan pesan dan ekstraksi untuk mengekstraksi pesan yang disembunyikani. Proses - proses tersebut dapat dilihat pada Gambar 2.10 dan Gambar 2.11



Gambar 2.10 *Embedding* Citra



Gambar 2.11 Ekstraksi Citra

Keterangan :
→ = input/output
⇢ = input optional
* = dalam banyak kasus tidak kembali/ hilang

Pada Gambar 2.10 diperlihatkan proses penyembunyian pesan dimana di bagian pertama, dilakukan proses *embedding hiddenimage* yang hendak disembunyikan secara rahasia ke dalam *stegomedium* sebagai media penyimpanan, dengan memasukkan kunci tertentu (*key*), sehingga dihasilkan media dengan data tersembunyi di dalamnya (*stegoimage*). Pada Gambar 2.11, dilakukkan proses ekstraksi pada *stegoimage* dengan memasukkan *key* yang sama sehingga didapatkan kembali *hiddenimage*. Kemudian dalam kebanyakan teknik steganografi, ekstraksi pesan tidak akan mengembalikan s*tegomedium* awal persis sama dengan *stegomedium* setelah dilakukan ekstraksi bahkan sebagian besar mengalami kehilangan. Karena saat penyimpanan pesan tidak dilakukan pencatatan kondisi awal dari *stegomedium* yang digunakan untuk menyimpan pesan [COX-08].

2.3.2 *Least Significant Bit* (LSB)

Strategi penyembunyian data citra yang digunakan untuk menyisipkan citra kedalam media citra adalah dengan metode *Least Significant Bit* (LSB).

Dimana bit data citra akan digantikan dengan bit paling rendah dalam media citra. Pada *file* citra 24 bit setiap piksel pada citra terdiri dari susunan tiga warna, yaitu merah, hijau dan biru (RGB) yang masing-masing disusun oleh bilangan 8 bit (1 byte) dari 0 sampai 255 atau dengan format biner 00000000 sampai 11111111. Informasi dari warna biru berada pada bit 1 sampai bit 8, dan informasi warna hijau berada pada bit 9 sampai dengan bit 16, sedangkan informasi warna merah berada pada bit 17 sampai dengan bit 24.

Istilah algoritma substitusi LSB adalah skema yang paling sederhana untuk menyembunyikan pesan dalam sebuah citra host. Beliau mengganti bit yang tidak signifikan dari masing-masing piksel dengan sedikit aliran pesan terenkripsi. Penerima dapat mengambil pesan dengan menguraikan LSB dari setiap piksel dari *stegoimage* dengan kunci yang diberikan. Karena hanya sedikit yang signifikan dari piksel yang berubah maka secara visual tidak terlihat oleh manusia [KEK-08].

Metode LSB merupakan teknik substitusi pada steganografi. Biasanya, arsip 24-bit atau 8-bit digunakan untuk menyimpan citra digital. Representasi warna dari piksel–piksel bisa diperoleh dari warna–warna primer, yaitu merah, hijau dan biru. Citra 24-bit menggunakan 3 byte untuk masing–masing piksel, dimana setiap warna primer direpresentasikan dengan ukuran 1 byte. Penggunaan citra 24-bit memungkinkan setiap piksel direpresentasikan dengan nilai warna sebanyak 16.777.216. Dua bit dari saluran warna tersebut biasa digunakan menyembunyikan data yang akan mengubah jenis warna piksel-nya menjadi 64 warna. Hal itu akan mengakibatkan sedikit perbedaan yang tidak bisa dideteksi secara kasat mata oleh manusia [ARI-09].

Untuk menjelaskan metode ini, digunakan citra digital sebagai *stegomedium*. Pada setiap byte terdapat bit yang tidak signifikan. Misalnya pada byte 00011001, maka bit LSB-nya adalah 1. Untuk melakukan penyisipan pesan, bit yang paling tepat untuk diganti dengan bit pesan adalah bit LSB, sebab pengubahan bit tersebut hanya akan mengubah nilai byte-nya menjadi satu lebih tinggi atau satu lebih rendah. Sebagai contoh, urutan bit berikut ini menggambarkan 3 piksel pada *stegomedium* 24-bit.

(00100111 11101001 11001000)

(00100111 11001000 11101001)

(11001000 00100111 11101001)

Pesan yang akan disisipkan adalah karakter A yang nilai biner-nya adalah 01000001 (ASCII), maka akan dihasilkan *stegoimage* dengan urutan bit sebagai berikut:

(00100110 11101001 11001000)

(00100110 11001000 11101000)

(11001000 00100111 11101001)

Terlihat hanya tiga bit rendah yang berubah (bit dengan garis bawah), untuk mata manusia maka tidak akan tampak perubahannya. Secara rata-rata dengan metode ini hanya setengah dari data bit rendah yang berubah, sehingga bila dibutuhkan dapat digunakan bit rendah kedua bahkan ketiga [LES-06].

2.4 Citra Digital

Citra (*image)* atau gambar sebagai salah satu komponen multimedia memegang peranan penting sebagai bentuk informasi visual. Citra memiliki karakteristik yang tidak berbeda dengan yang dimiliki oleh teks, yaitu citra kaya akan informasi.

Ditinjau dari sudut matematisnya, citra merupakan fungsi menerus (*continue*) dari intensitas cahaya pada bidang dua dimensi. Sumber cahaya yang menuju obyek, dan dipantulkan kembali oleh obyek tersebut ditangkap oleh alat-alat optik, misalnya mata manusia, kamera pemindai (*scanner*) kamera digital, dan sebagainya, sehingga obyek citra tersebut dapat terekam.

Citra digital merupakan suatu larik/*array* dua dimensi atau suatu matrik yang elemen-elemennya menyatakan tingkat keabuan dari elemen gambar, jadi informasi yang terkandung di dalamnya bersifat diskrit [ARY-92].

Semua citra digital yang ditampilkan di layar komputer adalah sederetan atau sekumpulan piksel *(picture element)*. Citra tersebut dikatakan sebagai citra

digital karena bentuk representasinya yang berupa bilangan . Oleh komputer akan dikenal dalam urutan ‘0’ dan ‘1’.

2.4.1 Struktur Data Citra Digital

Suatu citra digital berbentuk matriks, di mana elemen-elemen matriks dapat diakses melalui indeksnya, yaitu baris dan kolom [MUN-04]. Sebuah citra digital berukuran N x M, dengan keterangan sebagai berikut:

1. N = jumlah baris (panjang/tinggi matriks) → $0 \leq y \leq N-1$
2. M = jumlah kolom (lebar matriks) → $0 \leq x \leq M-1$
3. L = intensitas warna maksimal (derjat keabuan) → $0 \leq F(x,y) \leq L-1$

Matrik citra digital berukuran N x M diperlihatkan oleh Gambar 2.12

$$f(0,0) \approx \begin{pmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \cdots & f(1,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,M-1) \end{pmatrix}$$

Gambar 2.12 Matriks citra digital berukuran NxM.

2.4.2 Citra *Bitmap*

Format BMP, disebut dengan *bitmap* adalah sebuah format citra yang digunakan untuk menyimpan citra bitmap digital. Pada citra berformat BMP (*bitmap*) yang tidak terkompresi, piksel citra disimpan dengan kedalaman warna 1, 4, 8, 16, 24, atau 32 bit per piksel. Terjemahan bebas bitmap adalah pemetaan bit. Artinya nilai intensitas piksel di dalam citra dipetakan ke sejumlah bit tertentu. Peta bit umumnya adalah 8, yang berarti setiap piksel panjangnya 8 bit. Delapan bit ini mempresentasikan nilai intensitas piksel. Dengan demikian ada sebanyak 2^8 = 256 derajat keabuan, mulai dari 0 (00000000) sampai 255 (11111111).

Pada umumnya citra bitmap terdiri dari 4 blok data *BMP header*, *Bit Information (DIB header)*, *Color Palette*, dan *Bitmap Data*. *BMP header* berisi informasi umum dari citra *bitmap* yang berada pada bagian awal file citra dan digunakan untuk mengidentifikasi citra. *Bit information* berisi informasi detail dari citra bitmap, yang akan digunakan untuk menampilkan citra pada layar.

Color palette berisi informasi warna yang digunakan untuk indeks warna bitmap, dan *bitmap data* berisi data citra yang sebenarnya, piksel per piksel.

Model ruang warna yang digunakan pada citra *bitmap* adalah RGB (*red, green, dan blue*). Sebuah ruang warna RGB dapat diartikan sebagai semua kemungkinan warna yang dapat dibuat dari tiga warna dasar *red, green,* dan *blue*. RGB sering digunakan di dalam sebagian besar aplikasi komputer karena dengan ruang warna ini tidak diperlukan transformasi untuk menampilkan informasi di layar monitor [MUN-04].

Terdapat tiga macam citra dalam format BMP, adalah sebagai berikut:

1. **Citra biner**. Citra biner hanya memiliki dua nilai keabuan 0 dan 1. Oleh kerena itu 1 bit telah cukup untuk mempresentasikan nilai piksel.
2. **Citra hitam-putih** (*grayscale*).
3. **Citra berwarna**. Citra berwarna adalah citra yang lebih umum. Warna yang terlihat didalam citra bitmap merupakan kombinasi dari tiga komponen warna R (Red), G (Green) dan B (Blue). Pada citra 256 warna, setiap piksel memiliki panjang 8-bit, akan tetapi komponen RGB-nya disimpan dalam tabel RGB yang disebut *palet.*

Format citra 4-*bit* (16 warna), hampir sama dengan format citra 8-*bit*. Pada citra 4-*bit* dan citra 8-*bit,* warna suatu piksel diacu dari tabel informasi palet *entry* ke-k (*k* merupakan nilai rentang 0-15 untuk citra 16 warna dan 0-155 untuk citra 256 warna). Sebagai contoh pada gambar 2.13, piksel pertama bernilai 2, warna piksel pertama ini ditentukan oleh komponen RGB pada *palet* warna *entry* ke-2, yaitu R=14, G=13 dan B=16. Piksel kedua serupa dengan piksel pertama. Piksel ketiga bernilai 1, warna ditentukan oleh komponen RGB pada *palet* warna *entry* ke-1, yaitu R=20, G=45 dan B=24. Demikian seterusnya untuk piksel-piksel lainnya. Khusus untuk citra hitam-putih 8-*bit,* komponen R,G dan B suatu piksel bernilai sama dengan data bitmap piksel tersebut. Jadi piksel dengan nilai data bitmap 129, memiliki nilai R=129, G=129 dan B=129. Format citra 8-*bit* dapat dilihat pada Gambar 2.13 [MUN-04].

<header berkas>

<header*bitmap*>

<palet warna RGB>

	R	G	B
1	20	45	24
2	14	13	16
3	12	17	15
…			
256	46	78	25

<data *bitmap*>

2 2 1 1 1 3 5 . . .

Gambar 2.13 Format citra 8-bit.

Pada citra 24-*bit* setiap piksel panjangnya 24-*bit*, karena setiap bit langsung menyatakan komponen warna merah (8-bit), komponen warna hijau (8-*bit*) dan komponen warna biru (8-*bit*). Citra 24-*bit* juga disebut citra 16 juta warna karena mampu menghasilkan 2^{24} =16.777.216 kombinasi warna. Contohnya seperti pada Gambar 2.13 piksel pertama memiliki nilai R=20, G=19 dan B=21. Piksel kedua memiliki nilai R=24, G=23 dan B=24 dan demikian seterusnya. Format citra 24-*bit* dapat dilihat pada Gambar 2.14 [MUN-04].

<header berkas>

<header*bitmap*>

<data*bitmap*>

20 19 21 24 24 23 24 . . .

Gambar 2.14 Format citra 24-bit.

2.5 *Peak Signal To Noise Ratio* (PSNR)

Peak Signal to Noise Ratio (PSNR) adalah perbandingan antara nilai maksimum dari sinyal yang diukur dengan besarnya derau yang berpengaruh pada sinyal tersebut. PSNR biasanya diukur dalam satuan desibel. PSNR digunakan untuk mengetahui perbandingan kualitas citra sebelum dan sesudah disisipkan pesan. Untuk menentukan PSNR perhitungannya dinyatakan pada persamaan 2.8.

$$\text{PSNR} = 20xLog10\left(255 \Big/ \sqrt{\frac{1}{MN}\sum_{Y=1}^{M}\sum_{X=1}^{N}[I(x,y)-I'(x,y)]^2}\right) \quad \textbf{(2.8)}$$

Dimana :

m = panjang citra tersebut (dalam *pixel*)

n = lebar citra tersebut (dalam *pixel*)

(x,y) = koordinat masing-masing *pixel*

I = nilai bit pada citra asli

I' = nilai bit pada citra steganografi

Nilai PSNR yang besar akan lebih baik. Nilai kualitas image secara PSNR ditunjukkan oleh Tabel 2.4

Tabel 2.4 Nilai Kualitas Image Secara PSNR

PSNR (db)	Kualitas Image
60	Sangat Baik
50	Baik
40	Layak/pantas
30	Tidak Baik
20	Buruk

[SIG-09].

BAB III

METODE PENELITIAN

Pada bab metodologi dan perncangan ini, akan dibahas metode atau langkah-langkah yang digunakan dalam penelitian ini.

Berikut ini langkah-langkah yang dilakukan dalam penelitian ini.

1. Mempelajari teori-teori dari literatur dan artikel yang berhubungan dengan penelitian.

   Mempelajari teori tentang algoritma AES 256, algoritma *Least Significant Bit* serta mempelajari bagaimana teknis untuk menggabungkan kedua algoritma tersebut dari buku, jurnal, artikel dan dari penelitian-peneilitian sebelumnya.
2. Melakukan analisis dan membuat rancangan model perangkat lunak dengan mengimplementasikan metode yang akan digunakan.

   Setelah mempelajari algoritma AES 256, *Least Significant Bit* dan cara penggabungannya, melakukan analisis apa saja input yang diperlukan dan bagaimana output yang akan ditampilkan. Dari analisis tersebut dibentuk flowchart dan gambaran antarmuka yang akan ditampilkan.
3. Membuat perangkat lunak berdasarkan analisa dan perancangan yang telah dilakukan.

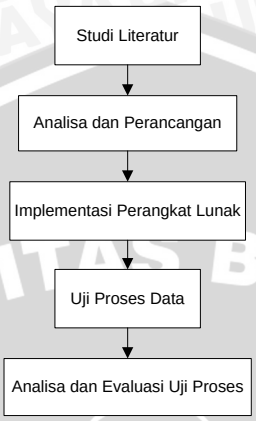
   Setelah membuat flowchart dan gambaran antarmuka mulai membuat perangkat lunak berdasarkan flowchart dan antarmuka tersebut.
4. Melakukan implementasi algoritma dengan perangkat lunak yang telah dibuat.

   Setelah proses pembuatan perangkat lunak selesai dilakukan proses pengujian terkait dengan kelayakan perangkat lunak dan implementasi dari algortima AES 256 dan *Least Significant Bit*.

5. Melakukan analisis dan evaluasi hasil implementasi algoritma.

Setelah dilakukan pengujian dilakukan analisis terhadap hasil dari pengujian sehingga dapat ditarik kesimpulan berdasarkan hasil tersebut.

Langkah-langkah penelitian ini ditunjukkan oleh Gambar 3.1



Gambar 3.1 langkah-langkah Penelitian

BAB IV

PERANCANGAN

Pada bab ini akan dijelaskan rancangan yang akan digunakan serta langkah-langkah implementasi algoritma AES 256 dan *Least Significant Bit* (LSB) sehingga dapat dianalisa secara implementatif dalam penelitian ini.

4.1 Analisa Sistem

Pada subbab ini akan dibahas mengenai deskripsi sistem dan batasan sistem.

4.1.1 Dekripsi Umum Sistem

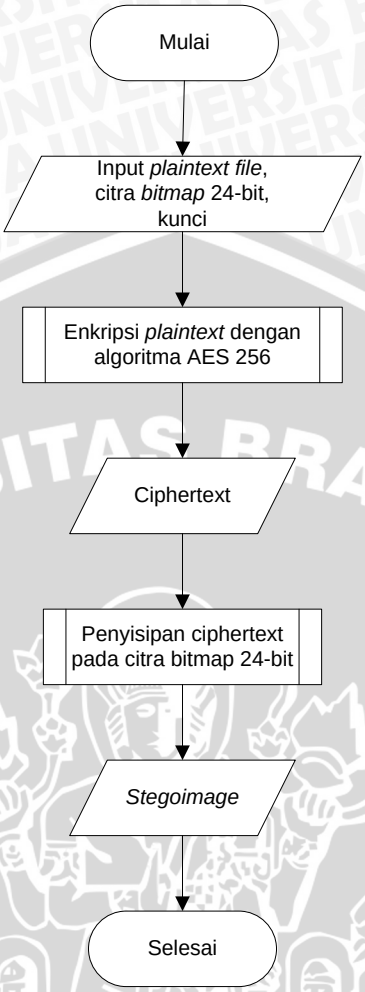
Sistem yang akan dibuat adalah sebuah aplikasi perangkat lunak yang mengimplementasikan dua metode, yaitu kriptografi dengan algoritma AES 256 dan steganografi dengan teknik *Least Significant Bit* (LSB). Aplikasi ini terdiri dari dua proses utama, yaitu pengamanan data dan penguraian data.

Pengamanan data melakukan enkripsi terhadap data yang berupa *plaintext file*, kemudian dikonversikan menjadi *hexadecimal* dan membaginya dalam blok – blok dengan panjang 128 bit. Kemudian tiap blok tersebut dilakukan enkripsi menggunakan algoritma AES 256, kemudian *ciphertext* hasil enkripsi tersebut disisipkan ke dalam sebuah citra *bitmap* 24-bit menggunakan algoritma LSB. Sedangkan penguraian data, *ciphertext* diurai dari citra, berupa *stegoimage* dengan prosedur pengurai data dari image, kemudian *ciphertext* didekripsi dengan algoritma AES 256 sehingga kembali menjadi *plaintext file* seperti semula.

Proses pengamanan data digambarkan dengan langkah-langkah sebagai berikut :

1. Memasukkan data pesan berupa *plaintext file*, citra *bitmap* 24-bit, dan kunci.
2. Mengenkripsi *plaintext* menggunakan algoritma AES 256 sehingga menghasilkan *ciphertext*.
3. Menyisipkan ciphertext ke dalam citra *bitmap* 24-bit sehingga menghasilkan *stegoimage*.

Langkah – langkah pengamanan data ditunjukkan oleh Gambar 4.1.

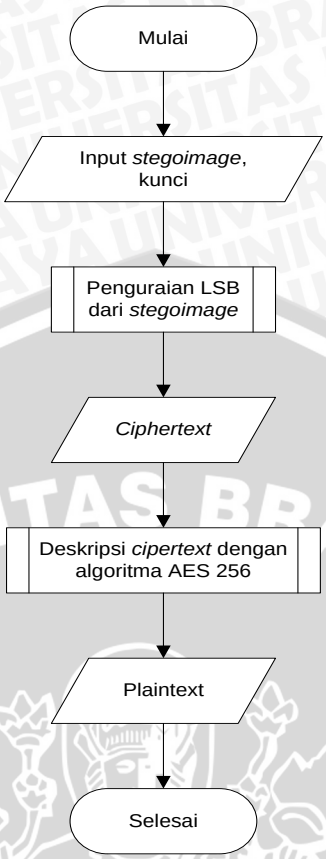


Gambar 4.1 Proses Pengamanan Data

Proses penguraian data digambarkan dengan langkah–langkah sebagai berikut :

1. Memasukkan *stegoimagefile* dan kunci.
2. Menguraikan LSB dari *stegoimage* menjadi *ciphertext*.
3. Mendekripsi *ciphertext*.

Langkah – langkah penguraian data ditunjukkan oleh Gambar 4.2.



Gambar 4.2 Proses Penguraian Data

4.1.2 Batasan Sistem

1. Citra digital yang digunakan untuk menyisipkan *ciphertext* adalah citra digital dengan format *bitmap* 24-bit (*.bmp).
2. Kunci (key) yang digunakan untuk enkripsi adalah karakter ASCII.

4.2 Perancangan Perangkat Lunak

4.2.1 Perancangan Pengamanan Data

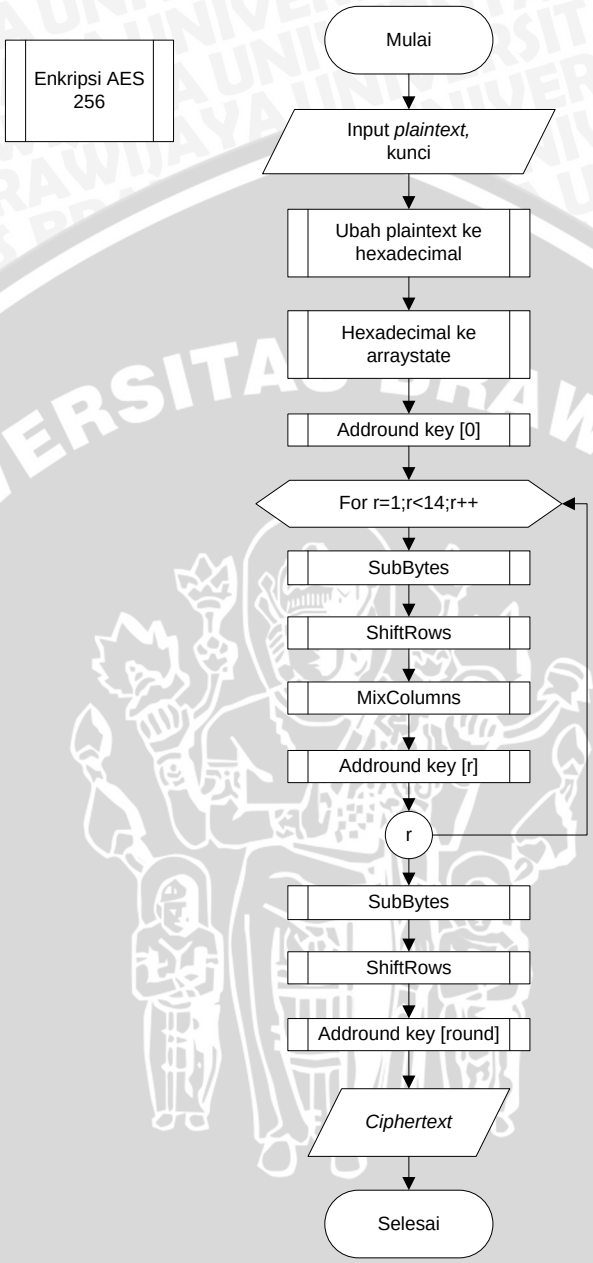
Subbab ini akan membahas mengenai perancangan pengamanan data yang mengimplementasikan kriptografi algoritma AES 256 dan steganografi LSB.

4.2.1.1 Proses Enkripsi AES 256

Pada proses enkripsi dilakukan penyisipan *chiper key* sebagai kunci rahasia agar hasil enkripsi bisa dibaca kembali.

Enkripsi pada algoritma AES 256 dilakukan dengan beberapa metode yakni, penambahan *round key*, *sub bytes*, *shift rows*, dan *mix columns* serta *key*

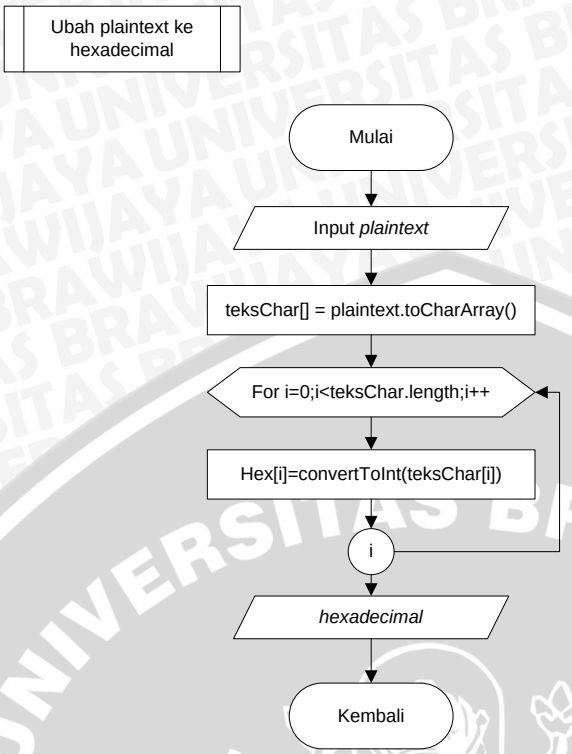
schedule untuk membangkitkan key baru. Alur proses enkripsi ditunjukkan pada Gambar 4.3.



Gambar 4.3 Proses Enkripsi AES 256

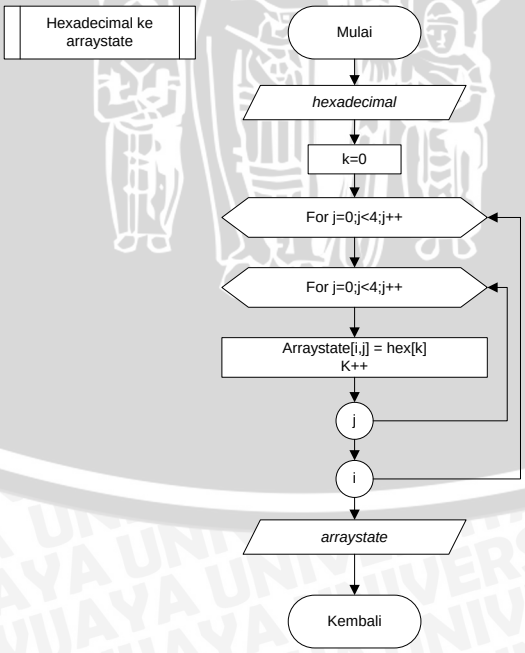
Secara umum deskripsi alur proses enkripsi berdasarkan Gambar 4.3 adalah:

1. Diambil data *plain text.*
2. Data *plain text* diubah ke bentuk *hexadecimal*. Proses ubah *plaintext* ke *hexadecimal* di tunjukan Gambar 4.4



Gambar 4.4 Proses Ubah *Plaintext* Ke *Hexadecimal*

3. Dilakukan pembentukan dari *hexadecimal* ke matrik *arraystate* (bentuk matrik 4x4). Proses pembentukan dari *hexadecimal* ke matrik *arraystate* (bentuk matrik 4x4) ditunjukan pada Gambar 4.5.

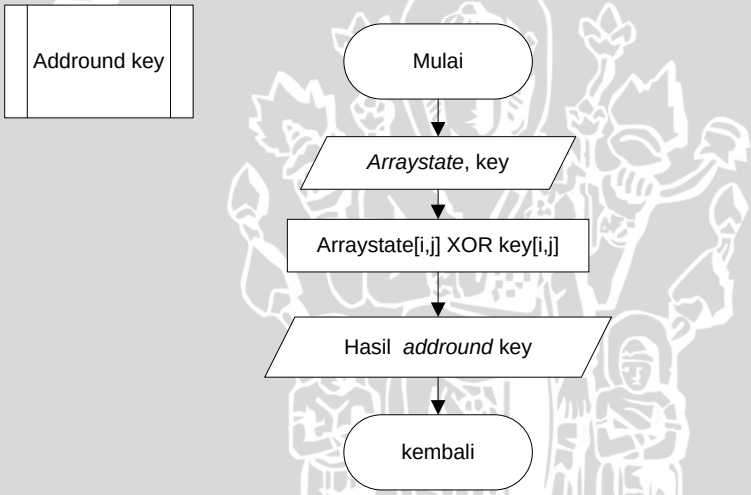


Gambar 4.5 Pembentukan *Hexadecimal* ke *Arraystate* (matrik 4x4)

4. Dilakukan proses penambahan *round key* menggunakan *chiper key*.
5. Dilakukan 13 kali perulangan untuk proses *sub bytes, shift rows, mix columns,* dan penambahan *round key*. Pada proses penambahan *round key* menggunakan *chiper key* baru yang sudah dibangkitkan dari proses *key schedule*.
6. Pada iterasi ke-14 dilakukan proses *sub bytes, shift rows*, dan penambahan *round key* ke-14.
7. Dihasilkan teks hasil enkripsi.

4.2.1.2 Add Round Key

Pada proses penambahan *round key* dilakukan penyisipan *chiper key*. Perancangan proses *addround key* terdapat pada Gambar 4.6.

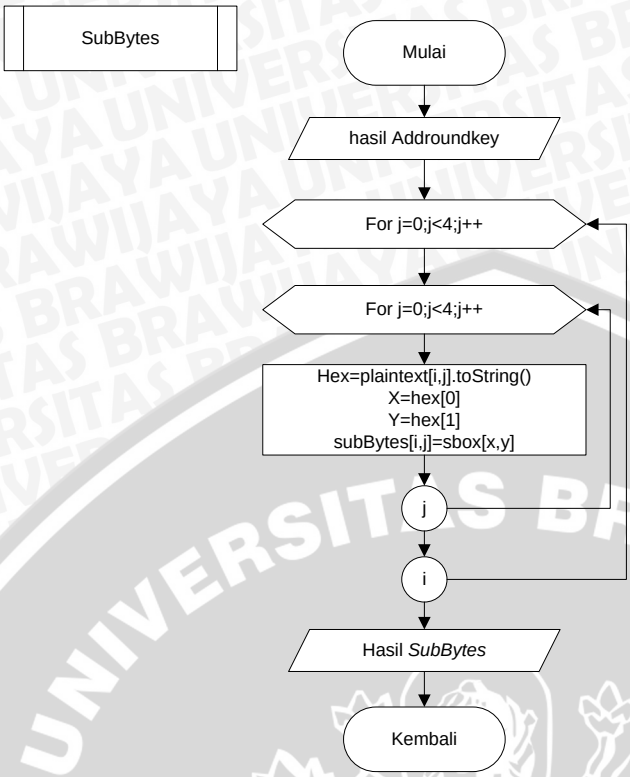


Gambar 4.6 Proses *addRound Key*

1. Diambil data teks yang sudah diubah ke bentuk *arraystate* (Gambar 4.5).
2. Dilakukan perhitungan XOR dengan *chiper key*.
3. Dihasilkan *array state* hasil penambahan *chiper key*.

4.2.1.3 Sub Bytes

Pada proses *sub bytes* dilakukan pemetakan setiap *byte* dari *array state* dengan menggunakan tabel substitusi S-*Box*. Perancangan proses *sub bytes* terdapat pada Gambar 4.7.

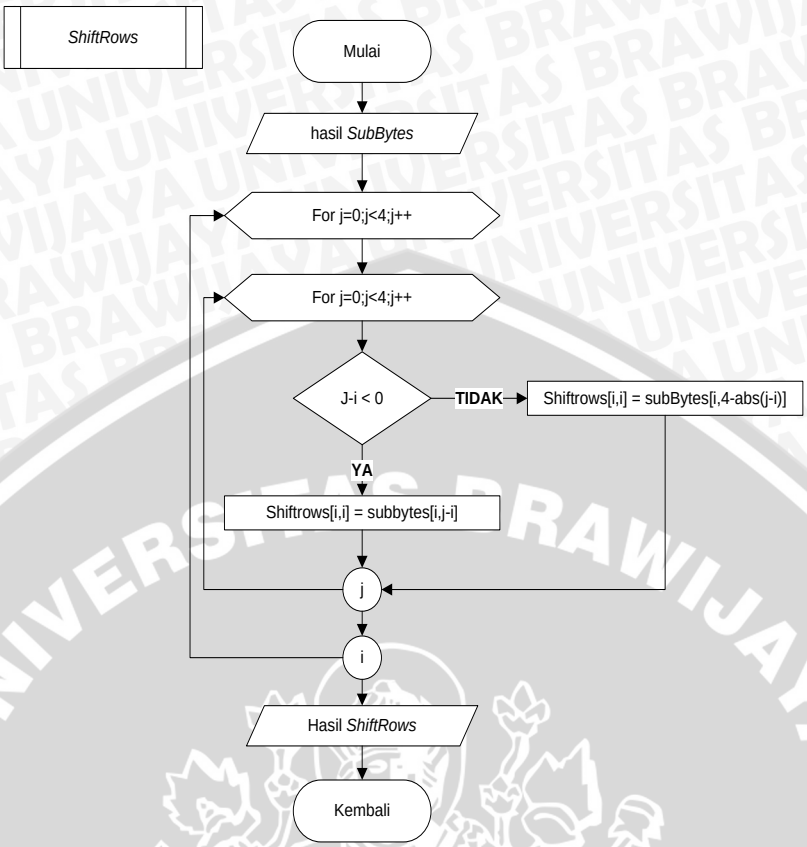


Gambar 4.7 *Sub Bytes*

1. Diambil hasil *addround key* (gambar 4.6).
2. Dilakukan perulangan untuk mengganti matrik hasil *round key* ke matrik baru hasil *sub bytes* dengan cara mengambil *hexadecimal*, untuk karakter pertama dijadikan parameter X dan karakter kedua dijadikan parameter Y. Parameter X dan Y dijadikan sebagai indeks untuk S-*Box*. Untuk tabel S-Box ditunjukkan pada bab 2.
3. Dihasilkan *array state* hasil *sub bytes*.

4.2.1.4 Shift Rows

Pada proses *shift rows* dilakukan pergeseran kekiri secara *wrapping* pada 3 baris terakhir *array state*. Perancangan proses *shift rows* terdapat pada Gambar 4.8.

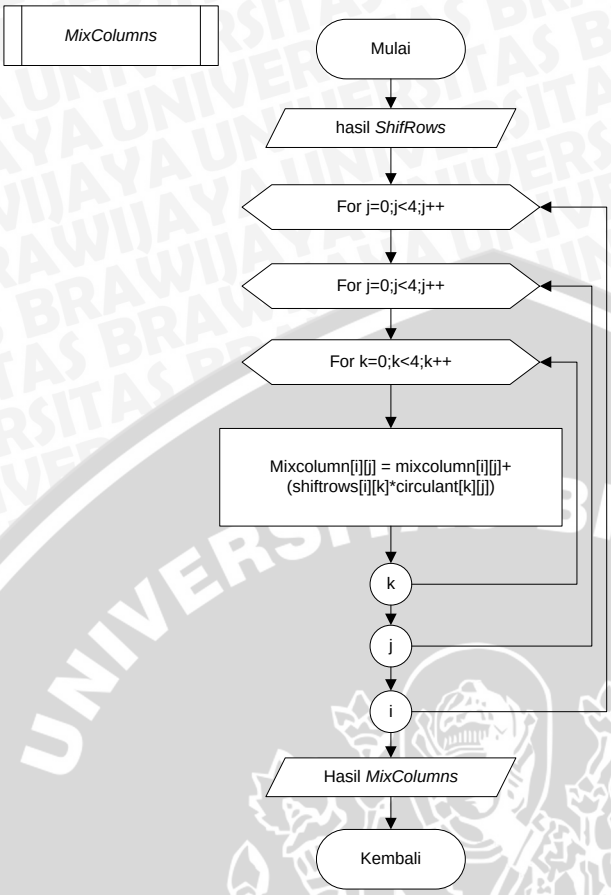


Gambar 4.8 *Shift Rows*

1. Diambil *array state* dari hasil *sub bytes* (Gambar 4.7).
2. Dilakukan perulangan untuk menggeser *array state* mulai dari baris 1 sebanyak 0 pergeseran (tidak ada pergeseran), baris 2 sebanyak 1 pergeseran, baris 3 sebanyak 2 pergeseran, dan baris 4 sebanyak 3 kali pergeseran. Pergeseran dilakukan ke kiri menurut indeks *j*, sehingga sebesar *j-i*. Apabila *j-i* bernilai minus, maka dilakukan pergeseran sebesar 4-*abs(j-i)*.
3. Dihasilkan *array state* hasil *shift rows*.

4.2.1.5 Mix Columns

Pada proses *mix columns* dilakukan perkalian setiap kolom *array state* dengan polinom. Perancangan proses *mix columns* terdapat pada Gambar 4.9.



Gambar 4.9 *Mix Columns*

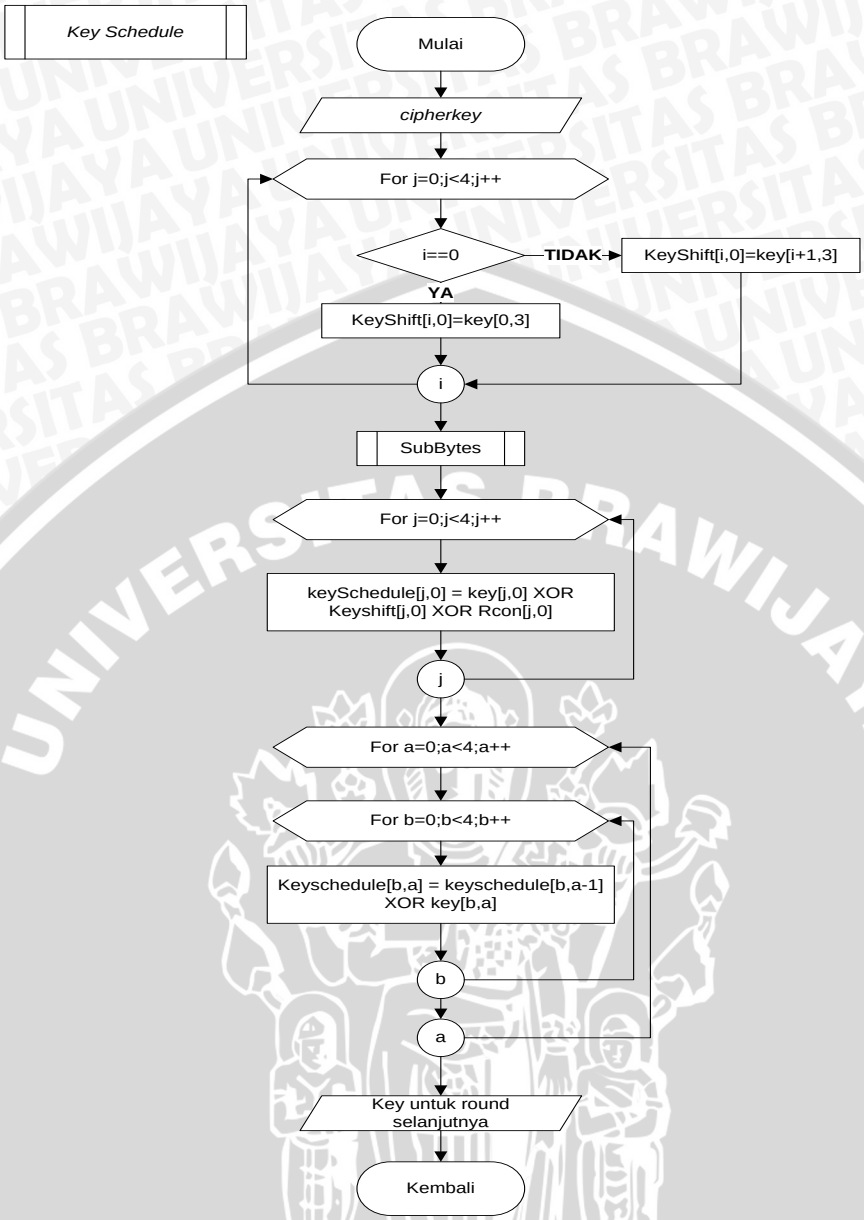
1. Diambil *array state* hasil *shift rows* (gambar 4.8).
2. Dilakukan perkalian matriks antara *array state* dengan *circulant* matrik dengan cara 3 kali perulangan.
3. Dihasilkan *array state* hasil *mix columns*.

4.2.1.6 Key Schedule

Pada key *schedule* dilakukan proses untuk membangkitkan *key* baru dari *chiper key*. Perancangan proses *key schedule* terdapat pada Gambar 4.10.

1. Diambil *chiper key*.
2. Diambil kolom terakhir pada *array state* untuk digeser ke atas. Indeks *i* menunjukkan baris, sehingga pergeseran dihitung dengan *i*+1 dan pada *i* sama dengan 0 digeser pada indeks 3.
3. Dilakukan proses *sub bytes* pada kolom hasil pergeseran.

4. Dilakukan XOR hasil *sub bytes* tersebut dengan *Rcon* untuk kolom pertama pada key *schedule*.
5. Untuk kolom kedua diambil berdasarkan kolom pertama. Dilakukan XOR untuk kolom pertama key *schedule* dengan kolom kedua pada *chiper key*.
6. Untuk kolom selanjutnya mengikuti *point* langkah nomor 5 sampai dengan kolom keempat.
7. Dihasilkan *key* baru.
8. *Key* baru tersebut nantinya akan dilakukan proses yang sama untuk mendapatkan *key* baru selanjutnya.



Gambar 4.10 *Key Schedule*

4.2.1.7 Proses Penyisipan *Ciphertext* Dengan LSB

Pesan yang telah disamarkan (*ciphertext*) dapat disisipkan ke dalam gambar/citra digital. Metode steganografi yang digunakan untuk menyisipkan/menyembunyikan *ciphertext* ke dalam citra adalah metode LSB. Langkah-langkah penyembunyian *ciphertext* ke dalam citra digital adalah sebagai berikut :

1. Diberikan masukan *ciphertext* dan citra digital bitmap 24-bit.
2. Ditambahkan penanda akhir pada *ciphertext*, yaitu bit – bit dari karakter '$3#' sebagai penanda akhir *ciphertext*.
3. Diperiksa apakah bit – bit *ciphertext* tersebut dapat disisipkan ke dalam gambar berdasarkan ukurannya. Kapasitas gambar, yaitu :

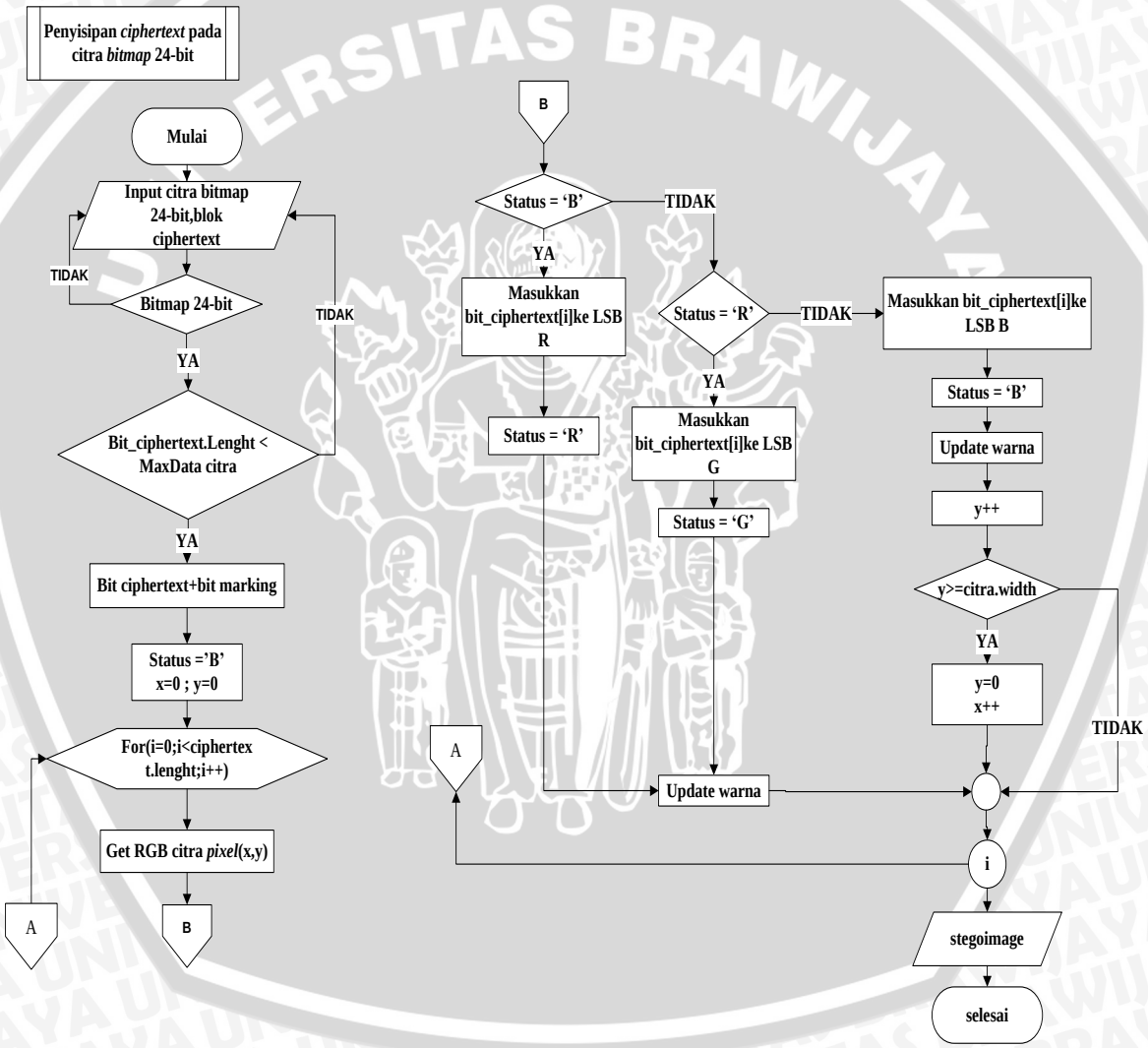
   $$MaxData\ citra = (citra.width * citra.height) * 3$$

   Jika tidak, maka akan diminta masukan *file* citra yang lebih besar.
4. Diinisialisasi koordinator (x,y) = (0,0) sebagai koordinator piksel pertama dan status piksel = 'B'.
5. Dilakukan iterasi dari i = 0 hingga i < panjang pesan keseluruhan yang telah ditambah bit – bit *marking*. Selama iterasi ini berlangsung, dilakukan :
   a. Dibaca piksel RGB citra dengan koordinat f(x,y) dimulai dari f(0,0).
   b. Diperiksa status piksel apakah status piksel = 'B'. Jika status piksel = 'B', bit_ciphertext[i] disisipkan ke LSB R dan status piksel diperbarahui = 'R', warna diperbaharui, dan iterasi dilanjutkan. Jika status piksel ≠ 'B', lanjut ke langkah c).
   c. Diperiksa status piksel apakah status piksel = 'R'. Jika status piksel = 'R', bit_ciphertext[i] disisipkan ke LSB G dan status piksel diperbarahui = 'G', warna diperbaharui,dan iterasi dilanjutkan. Jika status piksel ≠ 'R', lanjut ke langkah d).
   d. bit_ciphertext[i] disisipkan ke LSB B dan status piksel diperbarahui = 'B', warna diperbaharui.
   e. Koordinat f(x,y) diperbaharui dengan menambah nilai y sehingga pembacaan piksel bergeser ke kanan.

f. Diperiksa apakah y> citra.*width*. Jika y> citra.*width*, y diperbaharui y = 0 dan nilai x ditambah x = x+1 sehingga pembacaan piksel mulai dari awal baris berikutnya, paling kiri. Jika tidak, iterasi langsung dilanjutkan.

6. Setelah seluruh pesan disisipkan, diperoleh *stegoimage* dari *bitmap* 24-bit.

Flowchart dari langkah – langkah proses penyisipan pesan ditunjukkan oleh Gambar 4.11.



Gambar 4.11 Proses Penyisipan Pesan

4.2.2 Perancangan Penguraian Data

4.2.2.1 Proses Penguraian Pesan Dari *Stegoimage*

Penguraian pesan dari citra, yaitu mengambil bit – bit karakter yang tersembunyi di dalam tiap – tiap piksel citra. Setiap piksel memiliki 3 bit pesan yang disimpan dalam setiap LSB RGB. Setiap karakter yang akan terbentuk memerlukan 8 bit. Setelah diperoleh minimal 3 karakter atau 24 bit pesan, perangkat akan melakukan pemeriksaan, apakah 3 karakter tersebut identik dengan 3 karakter penanda akhir pesan atau *marking*. Jika identik, penguraian pesan dihentikan dan jika tidak, akan dilanjutkan penguraiannya hingga ditemukan 3 karakter terakhir yang identik dengan *marking* tersebut.

Langkah – langkah proses penguraian *ciphertext* adalah sebagai berikut :

1. Dimasukkan *stegoimage*, yaitu citra *bitmap* 24-bit yang mengandung *ciphertext* dan diinisialisasi pula bit *marking*.
2. Diinisialisasi sebuah *stringciphertext* = “ “.
3. Dilakukan iterasi dari i = 0 hingga i <citra.width agar pembacaan *pixel* bergeser ke bawah. Selama iterasi berlangsung dilakukan pula :
   a. Iterasi j = 0 hingga j <citra.height sehingga ketika dilakukan terlebih dahulu pembacaan *pixel* ke kanan.
   b. Diambil nilai bit LSB R sebagai *string*, kemudian nilai *string ciphertext* diperbaharui dengan menambah string *ciphertext* itu sendiri dengan nilai bit LSB R.
   c. Dilakukan pemeriksaan kondisi jika panjang *ciphertext* lebih dari atau sama dengan 24 dan 24 bit yang terakhir sama dengan *marking*, maka penguraian *ciphertext* dihentikan. Jika tidak, dilanjutkan ke langkah d.
   d. Diambil nilai bit LSB G sebagai *string*, kemudian nilai *string ciphertext* diperbaharui dengan menambah string *ciphertext* itu sendiri dengan nilai bit LSB G.
   e. Dilakukan pemeriksaan kondisi jika panjang *ciphertext* lebih dari atau sama dengan 24 dan 24 bit yang terakhir sama dengan *marking*,

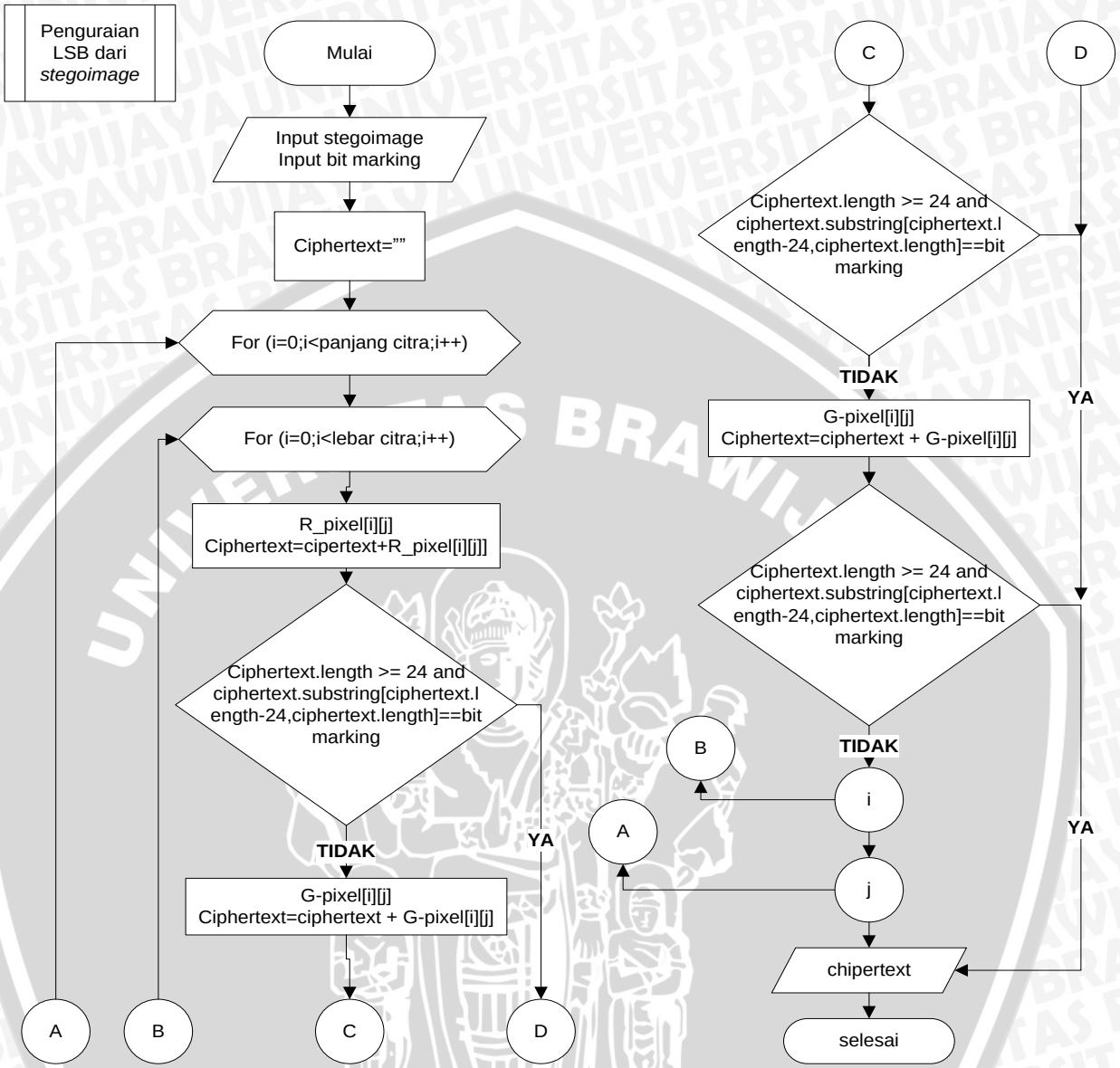
maka penguraian *ciphertext* dihentikan. Jika tidak, dilanjutkan ke langkah f.

f. Diambil nilai bit LSB B sebagai *string*, kemudian nilai *string ciphertext* diperbaharui dengan menambah string *ciphertext* itu sendiri dengan nilai bit LSB B.

g. Dilakukan pemeriksaan kondisi jika panjang *ciphertext* lebih dari atau sama dengan 24 dan 24 bit yang terakhir sama dengan *marking*, maka penguraian *ciphertext* dihentikan. Jika tidak, dilanjutkan kembali iterasinya.

4. Diperoleh bit – bit *ciphertext* beserta *marking*.

Flowchart penguraian *ciphertext* ditunjukkan oleh Gambar 4.12.

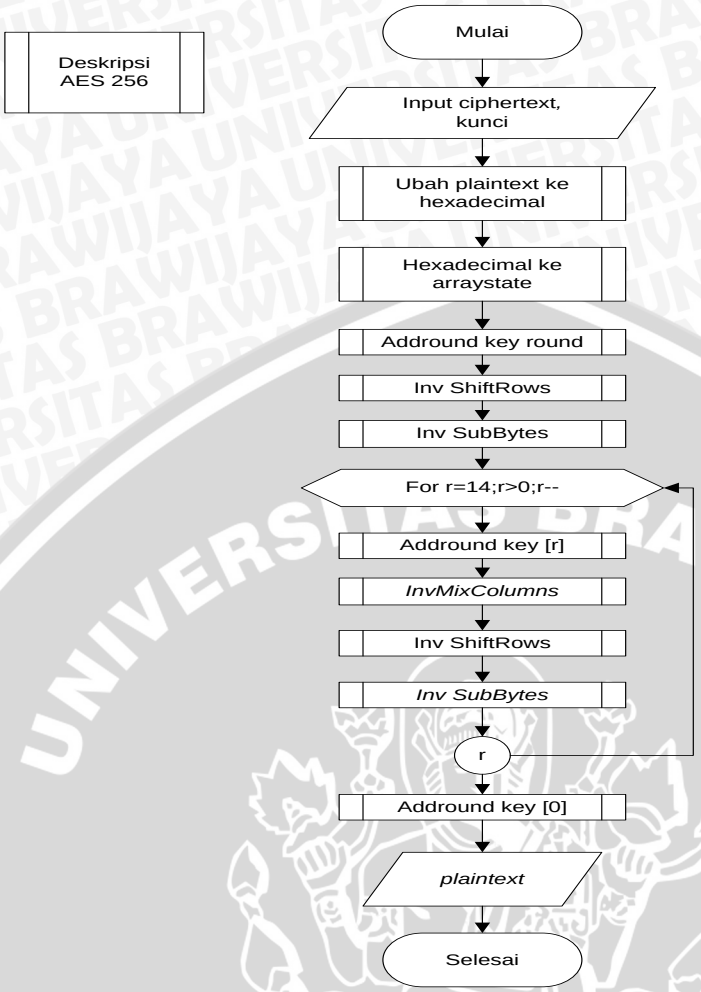


Gambar 4.12 Proses Penguraian Pesan

4.2.2.2 Proses Dekripsi AES 256

Pada proses dekripsi dilakukan penyisipan *chiper key* sama seperti pada proses enkripsi.

Dekripsi pada algoritma AES 256 dilakukan dengan beberapa metode yakni, penambahan *inverse round key, inverse sub bytes, inverse shift rows,* dan *inverse mix columns*. Alur proses Dekripsi ditunjukkan pada Gambar 4.13.



Gambar 4.13 Proses Dekripsi

Secara umum deskripsi alur proses dekripsi berdasarkan Gambar 4.13 adalah:

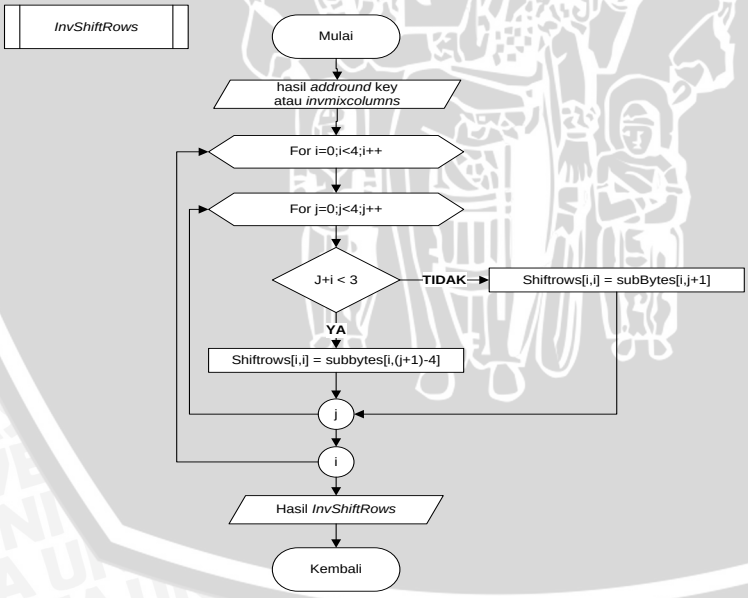
1. Diambil data *chipper text*
2. Data diubah ke bentuk *hexadecimal*.
3. Dilakukan proses penambahan *round key* dengan *chiper key* atau *key schedule* terakhir.
4. Dilakukan perulangan sebanyak 13 kali untuk proses *inverse shift rows*, *inverse sub bytes*, penambahan *round key*, dan *inverse mix columns*.
5. Pada proses penambahan *round key* dilakukan penambahan *chiper key* dengan *key* baru kebalikan dari proses enkripsi yakni dari *key schedule* terakhir ke awal.

6. Pada iterasi atau putaran terakhir dilakukan proses *inverse shift rows*, *inverse sub bytes*, dan penambahan *round key*.
7. Dihasilkan hasil dekripsi.

4.2.2.3 Inverse Shift Rows

Pada proses *inverse shift rows* dilakukan pergeseran kekanan secara *wrapping* pada 3 baris terakhir *array state*. Perancangan proses *inverse shift rows* terdapat pada *flowchart* Gambar 4.14.

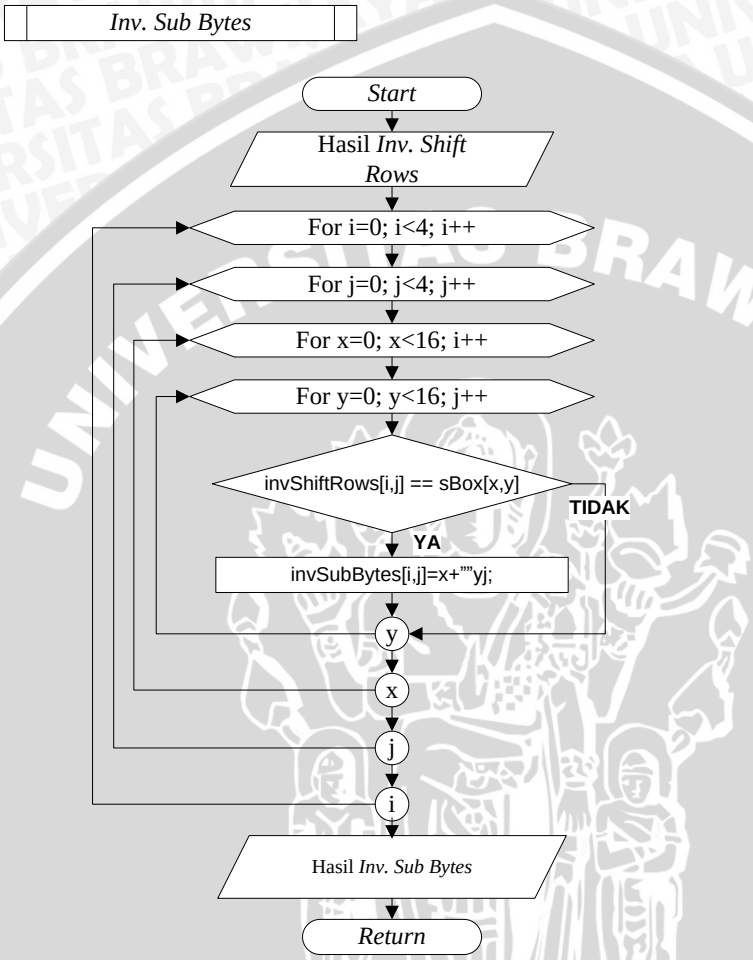
1. Diambil *array state* hasil *inverse mix columns* atau hasil penambahan *round key*.
2. Dilakukan perulangan untuk menggeser *array state* mulai dari baris 1 sebanyak 0 pergeseran, baris 2 sebanyak 1 pergeseran, baris 3 sebanyak 2 pergeseran, dan baris 4 sebanyak 3 kali pergeseran. Pergeseran dilakukan ke kanan menurut indeks *j*, sehingga sebesar *j+i*. Apabila *j+i* bernilai lebih dari 3, maka dilakukan pergeseran sebesar *(j+i)-4*.
3. Dihasilkan *array state* hasil *inverse shift columns*.



Gambar 4.14 *Inverse Shift Row*

4.2.2.4 Inverse Sub Bytes

Pada proses *inverse sub bytes* dilakukan pemetakan setiap *byte* dari *array state* dengan mengambil indeks dari tabel substitusi S-*Box* untuk disubtitusi. Perancangan proses *inverse sub bytes* terdapat pada *flowchart* Gambar 3.15.

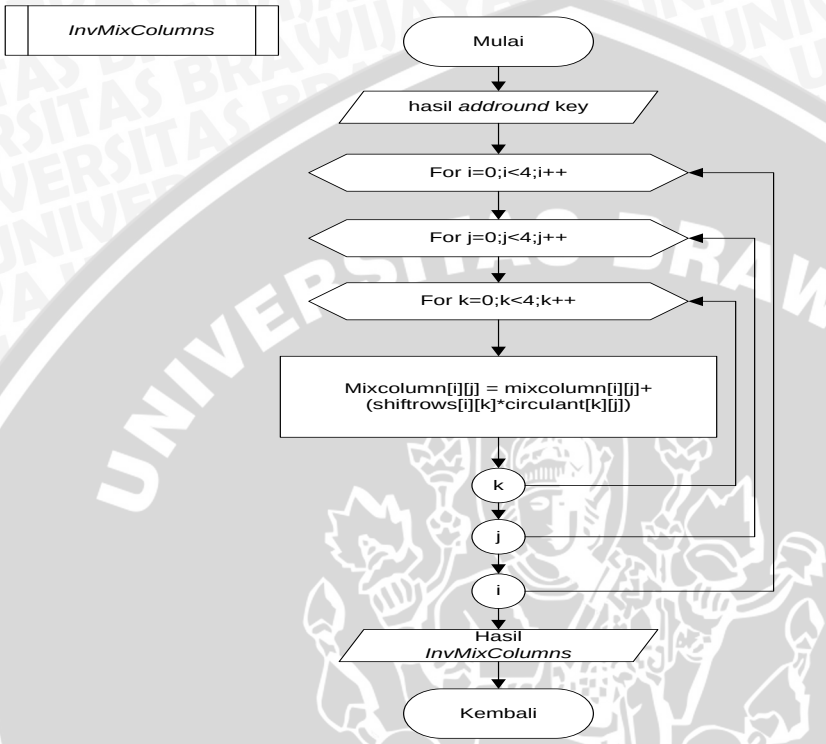


Gambar 4.15 *Inverse Sub Bytes*

1. Diambil hasil *inverse shift rows*.
2. Dilakukan perulangan untuk mengganti matrik hasil *inverse shift rows* ke matrik baru hasil *inverse sub bytes* dengan cara mengambil *hexadecimal*. Jika *array state inverse shift rows* merupakan nilai dari S-*Box*, maka diambil indeks X dan Y pada S-*Box* untuk nilai *inverse sub bytes*.
3. Dihasilkan *array state* hasil *inverse sub bytes*.

4.2.2.5 Inverse Mix Columns

Pada proses *inverse mix columns* dilakukan perkalian setiap kolom *array state* dengan *inverse* polinom. Perancangan proses *inverse mix columns* terdapat pada *flowchart* gambar 4.16.

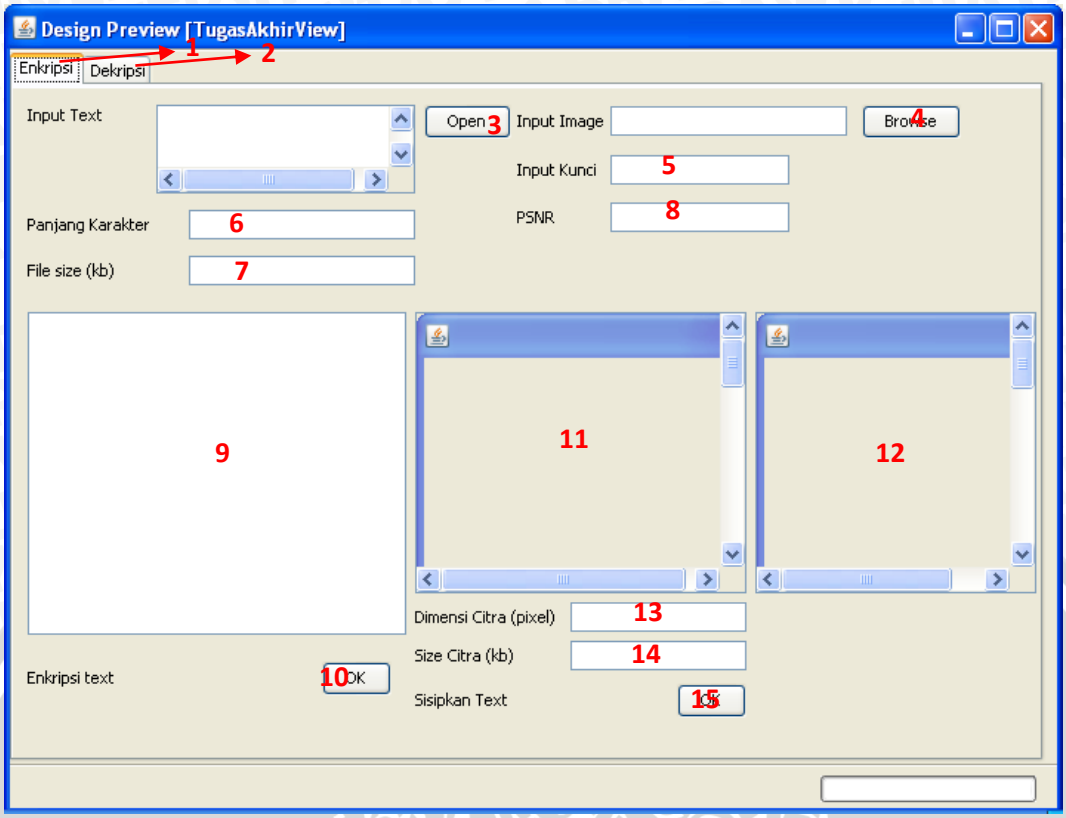


Gambar 4.16 *Inverse Mix Columns*

1. Diambil *array state* hasil *addround key*.
2. Dilakukan perkalian matriks antara *array state* dengan *circulant* matriks dengan cara 3 kali perulangan.
3. Dihasilkan *array state* hasil *inverse mix columns*.

4.3 Perancangan Antarmuka

Perancangan antar muka untuk perangkat lunak penyembunyian *ciphertext* ini terbagi menjadi dua bagian, yaitu tab pengamanan pesan dan tab penguraian pesan. Tab pengamanan pesan ditunjukkan oleh Gambar 4.17, sedangkan tab penguraian pesan ditunjukkan oleh Gambar 4.18.

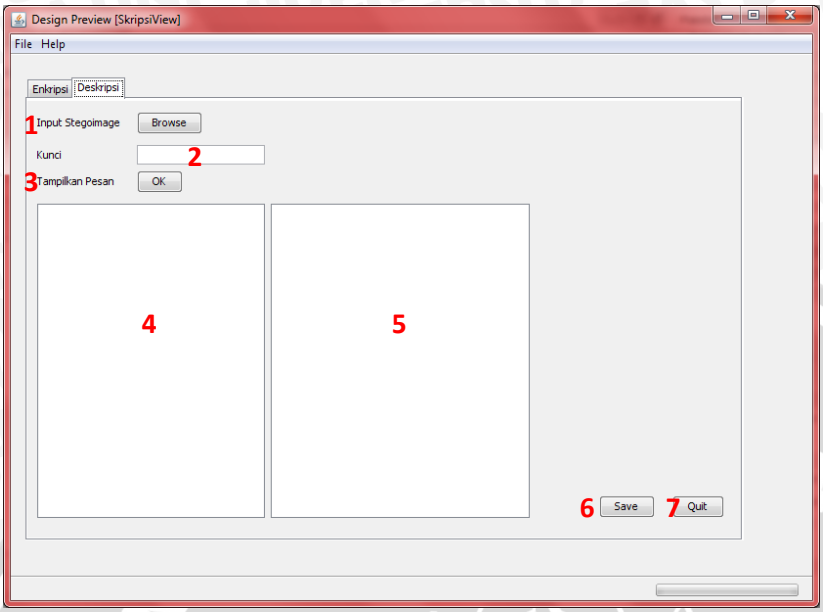


Gambar 4.17 Tab Pengamanan Pesan

Keterangan Gambar 4.17 adalah sebagai berikut:

1. *Tab* enkripsi.
2. *Tab* deskripsi.
3. *Button* untuk memuat *input text*
4. *Button* untuk memuat *input image*
5. *Textfield* untuk memasukkan kunci.
6. *Textfield* untuk mengetahui panjang pesan yang dimasukkan.
7. *Textfield* untuk mengetahui ukuran pesan yang dimasukkan.
8. *Area* menampilkan nilai PSNR dari *stegoimage*.
9. *Text Area* untuk menampilkan *ciphertext.*
10. *Button* untuk mengenkripsi pesan.
11. *Canvas* untuk menampilkan *image file* yang telah dimuat.
12. *Canvas* untuk menampilkan *stegoimage*.
13. *Textfield* untuk mengetahui dimensi image yang dimasukkan.
14. *Textfield* untuk mengetahui size image yang dimasukkan.
15. *Button* untuk melakukan prosedur penyembunyian *ciphertext*..

Tab yang menampilkan antarmuka penguraian pesan



Gambar 4.18 Tab Penguraian Pesan

Keterangan untuk Gambar 4.18 adalah sebagai berikut:

1. *Button* untuk memuat *stegoimage*.
2. *Textfiled* untuk memasukkan kunci.
3. *Button* untuk melakukan prosedur mengambil *ciphertext* dari *stegoimage* sekaligus mendekripsinya menjadi pesan berupa *plaintext*.
4. *Canvas* untuk menampilkan *stegoimage*.
5. *Textfield* untuk menampilkan pesan yang telah berupa *plaintext*.
6. *Button* untuk menyimpan pesan ke dalam *local disk*.
7. *Button* untuk menutup perangkat lunak.

4.4 Perancangan Uji Coba dan Evaluasi

Perancangan pengujian perangkat lunak ini dimaksudkan agar dapat mengetahui kinerja dari perangkat lunak. Selain itu, sebagai bahan untuk mengevaluasi hasil dari implementasi analisa dan perancangan perangkat lunak.

4.4.1 Pengujian Fungsional Perangkat Lunak

Pengujian fungsional perangkat lunak ini dilakukan bertujuan untuk memastikan bahwa perangkat lunak yang dihasilkan mampu melakukan pengamanan sekaligus penguraian data. Proses yang dilakukan adalah 3 buah *text*

files yang masing – masing diamankan ke dalam 2 buah *image files*. Proses pengamanannya adalah masing – masing *text file* dienkripsi terlebih dahulu kemudian disembunyikan ke dalam *image files* tersebut sehingga dihasilkan *stegoimage*. *Stegoimage* yang akan dihasilkan harus identik secara kasat mata. Tabel pengujian fungsionalitas perangkat lunak proses kriptografi ditunjukkan oleh Tabel 4.1.

Tabel 4.1 Pengujian Fungsionalitas Perangkat Lunak Proses Kriptografi

No.	Proses Pegamanan			
	Pesan Asli	Ciphertext	Key Untuk Dekripsi	Hasil Dekripsi

Proses berikutnya adalah menguraikan pesan dari *stegoimage*. Proses penguraiannya adalah *ciphertext* yang tersembunyi di dalam *stegoimage*, kemudian didekripsi agar dapat terbaca kembali. Tabel pengujian fungsionalitas perangkat lunak proses steganografi ditunjukkan oleh tabel 4.2.

Tabel 4.2 Pengujian Fungsionalitas Perangkat Lunak Proses Steganografi

No.	Proses Steganografi			
	File Citra	Pesan	Penyisipan	Penguraian

4.4.2 Pengujian Kinerja Perangkat Lunak

Skenario pengujian kinerja perangkat lunak antara lain:

1. Memasukkan *filetext* dan *file* citra yang berbeda – beda.
2. Menghitung nilai PSNR (*Peak Signal to Noise Ratio*). Nilai PSNR yang semakin besar menyatakan sinyal keluaran semakin mirip dengan sinyal asli.

Dari proses penyisipan pesan ke dalam *file* citra tentunya akan ada perbedaan kualitas citra sebelum dan sesudah proses penyisipan pesan, untuk mengetahui seberapa besar penurunan kualitas citra maka akan dilakukan perhitungan nilai PSNR seperti yang telah dijelaskan pada bab sebelumnya. Tabel 4.3 adalah tabel pengujian kinerja perangkat lunak dengan menggunakan *file text* dan *file* citra yang berbeda. Sedangkan tabel 4.4 adalah tabel pengujian kinerja perangkat lunak dengan menggunakan *file text* dan *file* citra yang sama akan tetapi dengan panjang kunci yang berbeda

Tabel 4.3 Pengujian Kinerja Perangkat Lunak Teks

No	Pesan	File Citra Asal	Citra Steganografi	Nilai PSNR

Tabel 4.4 Pengujian Kinerja Perangkat Lunak Kunci

No	Pesan	File Citra Asal	Ciphertext	Karakter	Kunci	Karakter	Nilai PSNR

4.4.3 Pengujian Ketahanan Citra Steganografi

Pengujian ketahanan citra steganografi ini dilakukan bertujuan untuk mengetahui ketahanan citra steganografi terhadap manipulasi citra seperti cropping, rotasi, dan penambahan efek sephia. Dalam prosesnya akan digunakan file text yang akan disisipkan ke file citra kemudian file citra yang sudah terisisipi akan dilakukan manipulasi citra. Setelah itu akan dilakukan proses dekripsi untuk mengetahui apakah pesan dapat kembali seperti pesan asli. Tabel pengujian ketahanan citra steganografi ditunjukkan oleh Tabel 4.4.

Tabel 4.5 Pengujian Ketahanan Citra Steganografi

No	Jenis Serangan	Perubahan Citra	Hasil Dekripsi	Error

4.5 Perhitungan Manual

4.5.1 Proses Enkripsi

Data yang digunakan dalam perhitungan manual yakni *plaintext* "ilmu komputer 07". Dari data *plaintext* tersebut diubah menjadi bentuk *hexadecimal* yakni menjadi "69 6C 6D 75 20 6B 6F 6D 70 75 74 65 72 20 30 37". Pada perhitungan manual ini digunakan AES 256 dengan ukuran blok 4, panjang kunci 4 blok dan terdapat 14 round. Hasil dari *hexadecimal plaintext* tersebut dipindahkan ke *array state* sebesar 4 blok seperti pada Gambar 4.19.

69	6C	6D	75
20	6B	6F	6D
70	75	74	65
72	20	30	37

Gambar 4.19 *Array State* Awal

4.5.1.1 Add Round Key

Dalam proses penambahan *round key* dilakukan perhitungan XOR *array state* dengan *chiperkey*. *Chiperkey* yang digunakan pada penelitian ini ditunjukkan pada gambar 4.20.

2B	28	AB	09
7E	AE	F7	CF
15	D2	15	4F
16	A6	88	3C

Gambar 4.20 *Chiperkey*

Dilakukan perhitungan XOR untuk *array state* dengan *chiperkey* terdapat pada Gambar 4.21.

69	6C	6D	75
20	6B	6F	6D
70	75	74	65
72	20	30	37

⊕

2B	28	AB	09
7E	AE	F7	CF
15	D2	15	4F
16	A6	88	3C

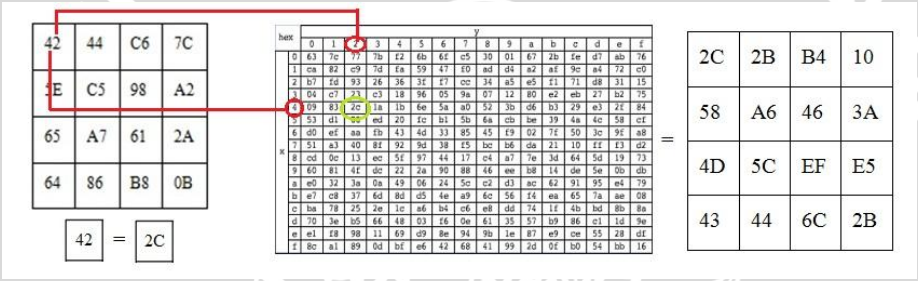
=

42	44	C6	7C
5E	C5	98	A2
65	A7	61	2A
64	86	B8	0B

Gambar 4.21 Perhitungan Tambah *Round Key*

4.5.1.2 Round 1 Sub Bytes

Pada langkah *sub bytes* pada *round* ke-1 dilakukan substitusi hasil *array state* pada penambahan *round key* pada sub bab 4.5.1 dengan S-*Box*. Hasil substitusi dari *sub bytes* terdapat pada Gambar 4.22.

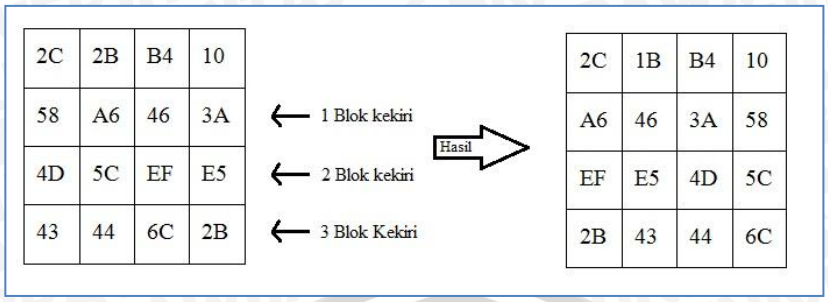


Gambar 4.22 Hasil *Round* 1 *Sub Bytes*

Nilai *hexadecimal* pada *array state* dijadikan indeks pada S-*Box*. pada *hexadecimal* terdapat 2 digit, digit pertama sebagai indeks x dan digit kedua sebagai indeks y pada S-*Box*.

4.5.1.3 Round 1 Shift Rows

Pada langkah *shift rows* dilakukan pergeseran blok *array* pada hasil *sub bytes*. Pergeseran dilakukan pada blok *array* baris ke-2 sampai dengan 4. Pergeseran dilakukan sebanyak 1 blok ke kiri pada baris ke 2, 2 blok kekiri pada baris ke 3, dan 3 blok kekiri pada baris ke 4. Hasil dari *shift rows* dapat dilihat pada Gambar 4.23.



Gambar 4.23 Hasil *Round* 1 *Shift Rows*

4.5.1.4 Round 1 Mix Columns

Pada langkah *mix columns* dilakukan perhitungan *multiplication* untuk *array state* hasil *shift rows* dengan *circulant* matriks. Hasil *mix columns* dapat dilihat pada Gambar 4.24.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 2C & 1B & B4 & 10 \\ A6 & 46 & 3A & 58 \\ EF & E5 & 4D & 5C \\ 2B & 43 & 44 & 6C \end{bmatrix} = \begin{bmatrix} 6D & 5A & 34 & F8 \\ 7A & E0 & 53 & 28 \\ 32 & 49 & D8 & 44 \\ 6B & 08 & 38 & EC \end{bmatrix}$$

Gambar 4.24 Hasil *Mix Columns*

{02}.{2C}+{03}.{A6}+{01}.{EF}+{01}.{2B}={6D}

{02}.{1B}+{03}.{46}+{01}.{E5}+{01}.{43}={5A}

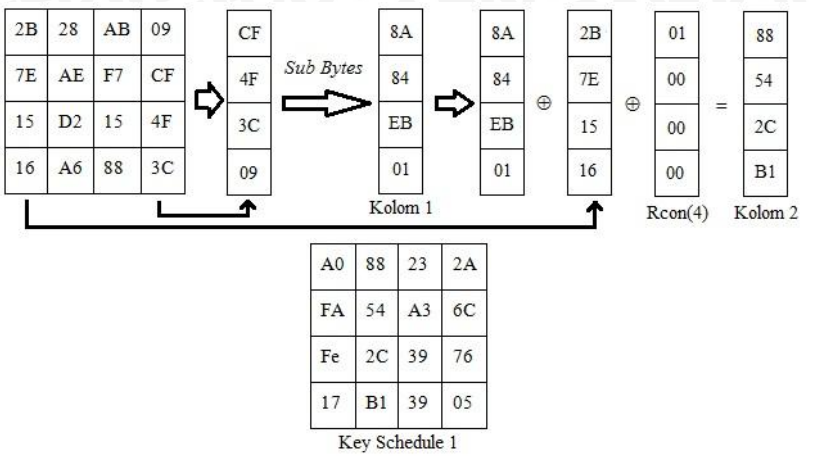
{02}.{B4}+{03}.{3A}+{01}.{4D}+{01}.{44}={34}

{02}.{10}+{03}.{58}+{01}.{5C}+{01}.{6C}={F8}

Perhitungan dilakukan pada semua sampai pada *hexadecimal* terakhir sehingga diperoleh hasil sesuai pada gambar 4.24.

4.5.1.5 Key Schedule

Pada langkah ini dilakukan pembangkitan *key* baru dari *chiper key* yang nantinya akan digunakan para proses penambahan *key schedule* dari *round* 1 sampai dengan 10. Pembangkitan *key* baru dapat dilihat pada gambar 4.25.



Gambar 4.25 Hasil *Key Schedule* 1

Setelah terbentuk *key schedule* ke-1, dihitung untuk *key schedule* ke-2 sampai dengan ke-14 dengan proses yang sama dengan Gambar 4.25.

4.5.1.6 Round 1 Add Round Key

Dalam proses penambahan *round key* pada *round* 1 dilakukan perhitungan XOR *array state* dengan *key schedule ke-1*. Hasil penambahan *round* key pada *round* 1 dapat dilihat pada Gambar 4.26.

CD	D2	17	D2
80	B4	F0	44
CC	65	E1	32
7C	B9	01	E9

Gambar 4.26 Hasil *Round* 1 *Add Round Key*

4.5.1.7 Round 14

Pada perhitungan putaran ke-2 sampai dengan ke-13 dilakukan perhitungan yang sama untuk *sub bytes, shift rows, mix columns*, dan *add round key*.

4.5.1.8 Penyisipan *Ciphertext* Pada *Image*

Pesan yang akan disisipkan berupa rangkaian bit *ciphertext*. Setiap bit tersebut disisipkan pada nilai LSB RGB dari piksel – piksel citra. Citra yang diberikan harus berupa bitmap 24-bit. Jika bukan, maka akan diberikan peringatan untuk mengganti *file* citra tersebut. Kemudian diperiksa kesesuaian kapasitas penyimpanannya terhadap bit – bit *ciphertext* tersebut.

1. Dimasukkan citra *bitmap* 24-bit. Misalnya citra berukuran 10x10 yang ditunjukkan oleh Gambar 4.27.

Gambar 4.27 Citra 10 x 10

Berikut nilai RGB dari piksel – piksel gambar tersebut ditunjukkan oleh Tabel 4.5.

Tabel 4.6 Nilai RGB Citra 10 x 10

Indeks Piksel	R		G		B	
	Dec		Dec		Dec	
(0,0)	200	11001000	23	00010111	150	10010110
(0,1)	221	11011101	27	00011011	174	10101110
(0,2)	231	11100111	35	00100011	192	11000000
(0,3)	234	11101010	35	00100011	193	11000001
(0,4)	228	11100100	11	00001011	175	10101111
(0,5)	225	11100001	0	00000000	159	10011111
(0,6)	230	11100110	13	00001101	185	10111001
(0,7)	223	11011111	20	00010100	176	10110000
(0,8)	222	11011110	45	00101101	181	10110101
(0,9)	221	11011101	39	00100111	174	10101110
(1,0)	208	11010000	0	00000000	155	10011011
(1,1)	208	11010000	13	00001101	160	10100000
(1,2)	217	11011001	27	00011011	169	10101001
(1,3)	232	11101000	23	00010111	186	10111010
(1,4)	224	11100000	0	00000000	154	10011010
(1,5)	224	11100000	0	00000000	140	10001100
(1,6)	226	11100010	1	00000001	170	10101010
(1,7)	219	11011011	18	00010010	169	10101001
(1,8)	222	11011110	46	00101110	178	10110010

(1,9)	223	11011111	45	00101101	180	10110100
(2,0)	221	11011101	1	00000001	170	10101010
(2,1)	222	11011110	15	00001111	172	10101100
(2,2)	219	11011011	19	00010011	167	10100111
(2,3)	224	11100000	12	00001100	166	10100110
(2,4)	223	11101000	0	00000000	119	1110111
(2,5)	212	11010100	0	00000000	97	1100001
(2,6)	216	11011000	0	00000000	121	1111001
(2,7)	206	11001110	0	00000000	134	10000110
(2,8)	222	11011110	36	00100100	175	10101111
(2,9)	221	11011101	30	00011110	177	10110001
(3,0)	230	11100110	5	00000101	171	10101011
(3,1)	229	11100101	0	00000000	158	10011110
(3,2)	224	11100000	0	00000000	130	10000010
(3,3)	215	11010111	0	00000000	120	1111000
(3,4)	166	10100110	29	00011101	55	110111
(3,5)	173	10101101	37	00100101	60	111100
(3,6)	192	11000000	0	00000000	76	1001100
(3,7)	202	11001010	0	00000000	109	1101101
(3,8)	212	11010100	3	00000011	128	10000000
(3,9)	219	11011011	1	00000000	159	10011111
(4,0)	236	11101100	40	00101000	190	10111110
(4,1)	230	11100110	0	00000000	143	10001111
(4,2)	224	11100000	12	00001100	117	1110101
(4,3)	182	10110110	63	00111111	84	1010100
(4,4)	170	10101010	144	10010000	52	110100
(4,5)	168	10101000	129	10000001	71	1000111
(4,6)	176	10110000	81	01010001	87	1010111
(4,7)	184	10111000	0	00000000	57	111001
(4,8)	200	11001000	0	00000000	84	1010100
(4,9)	217	11011001	0	00000000	131	10000011
(5,0)	236	11101100	40	00101000	190	10111110
(5,1)	236	11101100	33	00100001	188	10111100
(5,2)	235	11101011	0	00000000	162	10100010
(5,3)	217	11011001	176	10110000	139	10001011
(5,4)	215	11010111	174	10101110	139	10001011
(5,5)	200	11001000	161	10100001	135	10000111
(5,6)	200	11001000	47	00101111	110	1101110
(5,7)	198	11000110	0	00000000	80	1010000
(5,8)	209	11010000	0	00000000	120	1111000
(5,9)	220	11011100	1	00000001	157	10011101
(6,0)	233	11101001	43	00101011	189	10111101
(6,1)	234	11101010	37	00100101	183	10110111
(6,2)	232	11101000	0	00000000	150	10010110

(6,3)	235	11101011	0	00000000	141	10001101
(6,4)	233	11101001	0	00000000	148	10010100
(6,5)	234	11101010	0	00000000	149	10010100
(6,6)	220	11011100	0	00000000	112	1110000
(6,7)	201	11001001	0	00000000	91	1011011
(6,8)	205	11001101	0	00000000	127	1111111
(6,9)	217	11011001	26	00011010	162	10100010
(7,0)	235	11101011	29	00011101	193	11000001
(7,1)	233	11101001	2	00000010	182	10110110
(7,2)	234	11101010	0	00000000	158	10011110
(7,3)	232	11101000	0	00000000	158	10011110
(7,4)	231	11100111	0	00000000	153	10011001
(7,5)	230	11100110	0	00000000	128	10000000
(7,6)	222	11011110	0	00000000	136	10001000
(7,7)	215	11010111	0	00000000	120	1111000
(7,8)	211	11010011	6	00000110	142	10001110
(7,9)	213	11010101	25	00011001	160	10100000
(8,0)	236	11101100	25	00011001	193	11000001
(8,1)	232	11101000	0	00000000	175	10101111
(8,2)	230	11100110	0	00000000	171	10101011
(8,3)	232	11101000	0	00000000	174	10101110
(8,4)	230	11100110	8	00001000	171	10101011
(8,5)	224	11100000	0	00000000	156	10011100
(8,6)	223	11011111	0	00000000	154	10011010
(8,7)	219	11011011	0	00000000	144	10010000
(8,8)	221	11011101	1	00000001	154	10011010
(8,9)	223	11011111	14	00001110	154	10011010
(9,0)	234	11101010	0	00000000	179	10110011
(9,1)	232	11101000	0	00000000	164	10100100
(9,2)	233	11101001	0	00000000	182	10110110
(9,3)	232	11101000	0	00000000	184	10111000
(9,4)	227	11100011	0	00000000	174	10101110
(9,5)	221	11011101	0	00000000	163	10100011
(9,6)	221	11011101	0	00000000	161	10100001
(9,7)	219	11011011	0	00000000	158	10011110
(9,8)	222	11011110	0	00000000	161	10100001
(9,9)	220	11011100	5	00000101	172	10101100

Bit *ciphertext* yang akan disembunyikan :

“10110011111010000010100110110100111011010000100001110010001
11110110010010000111001000010100011100110111000100000011100 1
00110000101101000011000010111001101101001000001010100101001 0

1110011111110101100001100011001000111110111011111010000101101011011010010100”

ciphertext.Length = 256 bit

2. Ditambah marking pesan ‘$3#’ berupa nilai binernya.

“001001000011001100100011”

ciphertext.Length = 256 + 24 = 280

3. Apakah *ciphertext*> MaxData citra ?

280> (16 * 16) * 3

280>768

Tidak, penyisipan dapat dilanjutkan.

4. Index_pixel(x,y) = (0,0) dan status piksel = ‘B’.
5. Dilakukan iterasi i = 0 hingga i <280

Iterasi ke-0

f(0,0); status = ‘B’

bit_ciphertext[0] = 1

LSB R = 11001000

→LSB R = 1100100**1** ; status = ‘R’

update warna

Iterasi ke-1

f(0,0); status = ‘R’

bit_ciphertext[1] = 0

LSB G = 00010111

→LSB G = 0001011**0** ; status = ‘G’

update warna

Iterasi ke-2

f(0,0); status = ‘G’

bit_ciphertext[2] = 1

LSB B = 10010110

→LSB B = 0001011**1** ; status = ‘B’

Update warna

f(0,0+1) = f(0,1)

y tidak lebih besar atau sama dengan lebar citra.

Iterasi ke-3

f(0,1); status = ‘B’

bit_ciphertext[3] = 1

LSB R = 11011101

→LSB R = 1101110**1**; status = ‘R’

update warna

Iterasi ke-4

f(0,1); status = ‘R’

bit_ciphertext[4] = 0

LSB G = 00011011

→LSB G = 0001101**0**; status = ‘G’

update warna

Iterasi ke-5

f(0,1); status = ‘G’

bit_ciphertext[5] = 0

LSB B = 10101110

→LSB B = 0001011**0** ; status = ‘B’

Update warna

f(0,1+1) = f(0,2)

y tidak lebih besar atau sama dengan lebar citra.

Iterasi ke-6

f(0,2); status = ‘B’

bit_ciphertext[6] = 1

LSB R = 11100111

→LSB R = 1110011**1**; status = ‘R’

update warna

Iterasi ke-7

f(0,2); status = ‘R’

bit_ciphertext[7] = 1

LSB G = 00100011

→LSB G = 0010001**1**; status = ‘G’

update warna

Iterasi ke-8

f(0,2); status = ‘G’

bit_ciphertext[8] = 1

LSB B = 11000000

→LSB B = 1100000**1**; status = ‘B’

Update warna

f(0,2+1) = f(0,3)

y tidak lebih besar atau sama dengan lebar citra.

Iterasi ke-9

f(0,3); status = ‘B’

bit_ciphertext[9] = 1

LSB R = 11101010

→LSB R = 1110101**1**; status = ‘R’

update warna

Iterasi ke-10

f(0,3); status = ‘R’

bit_ciphertext[10] = 1

LSB G = 00100011

→LSB G = 0010001**1**; status = ‘G’

update warna

Iterasi ke-11

f(0,3); status = ‘G’

bit_ciphertext[11] = 0

LSB B = 11000001

→LSB B = 1100000**0**; status = ‘B’

Update warna

f(0,3+1) = f(0,4)

y tidak lebih besar atau sama dengan lebar citra.

Iterasi ke-12

f(0,4); status = ‘B’

bit_ciphertext[12] = 1

LSB R = 11100100

→LSB R = 11100101; status = ‘R’

update warna

Iterasi ke-13

f(0,4); status = ‘R’

bit_ciphertext[13] = 0

LSB G = 00001011

→LSB G = 00001010; status = ‘G’

update warna

Iterasi ke-14

f(0,4); status = ‘G’

bit_ciphertext[14] = 0

LSB B = 10101111

→LSB B = 10101110; status = ‘B’

Update warna

f(0,4+1) = f(0,5)

y tidak lebih besar atau sama dengan lebar citra.

Iterasi ke-15

f(0,5); status = ‘B’

bit_ciphertext[15] = 0

LSB R = 11100001

→LSB R = 11100000; status = ‘R’

update warna

Iterasi ke-16

f(0,5); status = ‘R’

bit_ciphertext[16] = 0

LSB G = 00000000

→LSB G = 00000000; status = ‘G’

update warna

Iterasi ke-17

f(0,5); status = ‘G’

bit_ciphertext[17] = 0

LSB B = 10011111

→LSB B = 1001111**0**; status = ‘B’

Update warna

f(0,5+1) = f(0,6)

y tidak lebih besar atau sama dengan lebar citra.

……

……

……

Iterasi ke-27

f(0,9); status = ‘B’

bit_ciphertext[27] = 1

LSB R = 11011101

→LSB R = 1101110**1**; status = ‘R’

update warna

Iterasi ke-28

f(0,9); status = ‘R’

bit_ciphertext[28] = 0

LSB G = 00100111

→LSB G = 0010011**0**; status = ‘G’

update warna

Iterasi ke-29

f(0,9); status = ‘G’

bit_ciphertext[29] = 1

LSB B = 10101110

→LSB B = 1010111**1**; status = ‘B’

Update warna

f(0,9+1) = f(0,10)

y lebih besar atau sama dengan lebar citra.

→f(x+1,y=0) = f(1,0)

Iterasi ke-30

f(1,0); status = ‘B’

bit_ciphertext[30] = 0

LSB R = 11010000

→LSB R = 1101000**0**; status = ‘R’

update warna

……

……

……

Iterasi ke-277

f(9,2); status = ‘R’

bit_ciphertext[277] = 0

LSB G = 00000000

→LSB G = 0000000**0**; status = ‘G’

update warna

Iterasi ke-278

f(9,2); status = ‘G’

bit_ciphertext[278] = 1

LSB B = 10110110

→LSB B = 1101000**1**; status = ‘B’

update warna

f(9,2+1) = f(9,3)

y tidak lebih besar atau sama dengan lebar citra.

Iterasi ke-279

f(9,3); status = ‘B’

bit_ciphertext[279] = 1

LSB R = 11101000

→LSB R = 1110100**1**; status = ‘R’

update warna

6. Diperoleh *stegoimage*, yaitu citra *bitmap* 24-bit yang mengandung pesan berupa *ciphertext*.

4.5.2 Proses Dekripsi

4.5.2.1 Penguraian Pesan Dari *Stegoimage*

Untuk memperoleh *ciphertext* yang telah disisipkan ke dalam citra, dilakukan penguraian pesan dari *stegoimage*. Penguraian dilakukan dengan mengambil bit – bit LSB RGB dari stegoimage, kemudian dirangkai menjadi bit

ciphertext. Penguraian dihentikan ketika 24 bit terakhir yang telah diambil adalah *marking* pesan, yaitu bit – bit dari '$3#'.

Bit marking = "001001000011001100100011"

Iterasi yang dilakukan adalah iterasi ke-i untuk indeks baris *stegoimage* dan iterasi ke-j untuk indeks kolom *stegoimage*.

Iterasi (0,0)

R = 1100100**1**

R_pixel[0][0] = 1

ciphertext = ciphertext + R_pixel[0][0]

ciphertext = "" + "1" = "1"

ciphertext.Length < 24 AND 24 bit ≠ bit marking

G = 0001011**0**

G_pixel[0][0] = 0

ciphertext = ciphertext + G_pixel[0][0]

ciphertext = "1" + "0" = "10"

ciphertext.Length < 24 AND 24 bit ≠ bit marking

B = 0001011**1**

B_pixel[0][0] = 1

ciphertext = ciphertext + B_pixel[0][0]

ciphertext = "10" + "1" = "101"

ciphertext.Length < 24 AND 24 bit ≠ bit marking

Iterasi (0,1)

R = 1101110**1**

R_pixel[0][1] = 1

ciphertext = ciphertext + R_pixel[0][1]

ciphertext = “101” + “1” = “1011”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

G = 0001101**0**

G_pixel[0][1] = 0

ciphertext = ciphertext + G_pixel[0][1]

ciphertext = “1011” + “0” = “10110”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

B = 0001011**0**

B_pixel[0][1] = 0

ciphertext = ciphertext + B_pixel[0][1]

ciphertext = ”10110” + “0” = “101100”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

Iterasi (0,2)

R = 1110011**1**

R_pixel[0][2] = 1

ciphertext = ciphertext + R_pixel[0][2]

ciphertext = “101100” + “1” = “1011001”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

G = 0010001**1**

G_pixel[0][2] = 1

ciphertext = ciphertext + G_pixel[0][2]

ciphertext = “1011001” + “1” = “10110011”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

B = 1100000**1**

B_pixel[0][2] = 1

ciphertext = ciphertext + B_pixel[0][2]

ciphertext = ”10110011” + “1” = “101100111”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

Iterasi (0,3)

R = 1110101**1**

R_pixel[0][3] = 1

ciphertext = ciphertext + R_pixel[0][3]

ciphertext = “101100111” + “1” = “1011001111”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

G = 0010001**1**

G_pixel[0][3] = 1

ciphertext = ciphertext + G_pixel[0][3]

ciphertext = “1011001111” + “1” = “10110011111”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

B = 1100000**0**

B_pixel[0][3] = 0

ciphertext = ciphertext + B_pixel[0][3]

ciphertext = ”10110011111” + “0” = “101100111110”

ciphertext.Length < 24 AND 24 bit ≠ bit marking

……

……

……

Iterasi (9,2)

R = 1110100**0**

R_pixel[9][2] = 0

ciphertext = ciphertext + R_pixel[9][2]

ciphertext =

“10110011111010000010100110110100111011010000100001110010001111101100100100001110010000101000111001101110001000000111001001001101000011000010111001101101001000001010100101001010111001111111010110000110001100100011111011101111101000010110101110110100101000010010000110011001” + “0”

ciphertext =

“10110011111010000010100110110100111011010000100001110010001111101100100100001110010000101000111001101110001000000111001001001101000011000010111001101101001000001010100101001010111001111111010110000110001100100011111011101111101000010110101110110100101000010010000110011001100”

ciphertext.Length ≥ 24 AND 24 bit ≠ bit marking

G = 0000000**0**

G_pixel[9][2] = 0

ciphertext = ciphertext + G_pixel[9][2]

ciphertext=

“10110011111010000010100110110100111011010000100001110010001111101100100100001110010000101000111001101110001000000111001001000011010000110000101110011011010010000010101001010010111001111111010110000110001100100011111101110111110100001011010111011010010100001001000011001100100” + “0”

ciphertext =

“10110011111010000010100110110100111011010000100001110010001111101100100100001110010000101000111001101110001000000111001001000011010000110000101110011011010010000010101001010010111001111111010110000110001100100011111101110111110100001011010111011010010100001001000011001100100”

ciphertext.Length ≥ 24 AND 24 bit ≠ bit marking

B = 1101000**1**

B_pixel[9][2] = 1

ciphertext = ciphertext + B_pixel[9][2]

ciphertext =

“10110011111010000010100110110100111011010000100001110010001111101100100100001110010000101000111001101110001000000111001001000011010000110000101110011011010010000010101001010010111001111111010110000110001100100011111101110111110100001011010111011010010100001001000011001100100” + “1”

ciphertext =

“10110011111010000010100110110100111011010000100001110010001111101100100100001110010000101000111001101110001000000111001001000011010000110000101110011011010010000010101001010

0101110011111110101100001100011001000111110111011111010000101101011101100101000010010000110011001000111”

ciphertext.Length ≥ 24 AND 24 bit ≠ bit marking

Iterasi (9,3)

R = 1110100**1**

R_pixel[9][3] = 1

ciphertext = ciphertext + R_pixel[9][3]

ciphertext =

“1011001111101000001010011011010011101101000010000111001000111110110010010000111001000010100011100110111000100000011100100011000010110100001100001011100110110100100000101010010100101110011111110101100001100011001000111110111011111010000101101011101100101000010010000110011001000111” + “1”

ciphertext =

“1011001111101000001010011011010011101101000010000111001000111110110010010000111001000010100011100110111000100000011100100011000010110100001100001011100110110100100000101010010100101110011111110101100001100011001000111110111011111010000101101011101100101000**001001000011001100100011**”

ciphertext.Length ≥ 24 AND 24 bit = bit marking

“001001000011001100100011”

→Break

Iterasi dihentikan, kemudian diperoleh bit – bit *ciphertext* beserta bit *marking*.

4.5.2.2 Data *Plaintext*

Ciphertext hasil enkripsi akan dilakukan proses dekripsi untuk mendapatkan *plaintext*. *Hexadecimal* yang terbentuk dari enkripsi adalah "44 D2 8F 83 7D 59 87 C7 3B 11 15 C0 0A 1D 4A 23", selanjutnya diubah ke bentuk *array state* 4 blok.

4.5.2.3 Add Round Key

Pada proses *add round key* dilakukan perhitungan XOR untuk *array state* awal dengan hasil *key schedule*. Hasil dari *add round key* terdapat pada gambar 4.28.

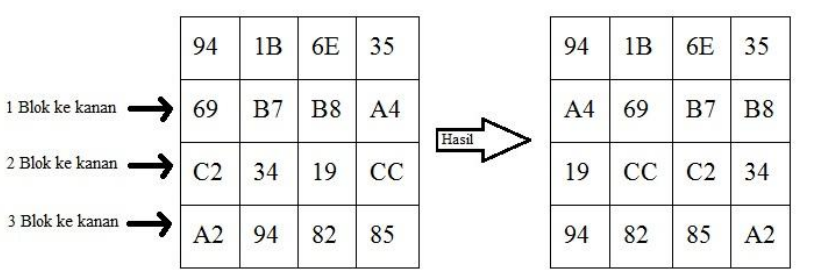
94	1B	6E	35
69	B7	B8	A4
C2	34	19	CC
A2	94	82	85

Gambar 4.28 Hasil *add Round Key*

Setelah didapatkan hasil *add round key*, selanjutnya akan dilakukan iterasi sebanyak 13 putaran. Pada setiap putaran dilakukan proses *inverse shift rows*, *inverse sub bytes*, tambah *round key*, dan *inverse mix columns*.

4.5.2.4 *Round* 1 *Inverse Shift Rows*

Pada langkah *inverse shift rows* dilakukan pergeseran blok *array* pada hasil *add round key*. Pergeseran dilakukan pada blok *array* baris ke-2 sampai dengan 4. Pergeseran dilakukan sebanyak 1 blok ke kanan pada baris ke 2, 2 blok kekanan pada baris ke 3, dan 3 blok kekanan pada baris ke 4. Hasil dari *inverse shift rows* dapat dilihat pada Gambar 4.29.



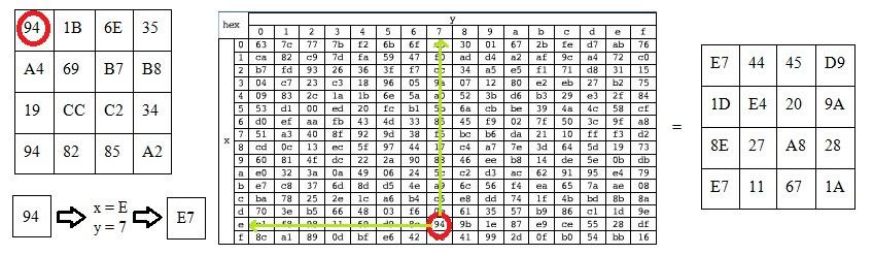
	94	1B	6E	35
1 Blok ke kanan →	69	B7	B8	A4
2 Blok ke kanan →	C2	34	19	CC
3 Blok ke kanan →	A2	94	82	85

94	1B	6E	35
A4	69	B7	B8
19	CC	C2	34
94	82	85	A2

Gambar 4.29 Hasil *Round* 1 *Inverse Shift Rows*

4.5.2.5 *Round* 1 *Inverse Sub Bytes*

Pada langkah *sub bytes* pada *round* ke-1 dilakukan substitusi hasil *array state* pada hasil inverse shift rows dengan S-*Box*. Hasil substitusi dari *sub bytes* terdapat pada Gambar 4.30.



94	1B	6E	35
A4	69	B7	B8
19	CC	C2	34
94	82	85	A2

E7	44	45	D9
1D	E4	20	9A
8E	27	A8	28
E7	11	67	1A

Gambar 4.30 Hasil Round 1 *Inverse Sub Bytes*

Nilai *hexadecimal* pada *array state* dijadikan sebagai nilai pada S-*Box*, kemudian dari nlai tersebut diambil sumbu X dan sumbu Y. Indeks yang merupakan sumbu X dan Y digabung, sehingga hasil substitusi bernilai XY.

4.5.2.6 *Round* 1 *Add Round Key*

Dalam proses penambahan *round key* pada *round* 1 dilakukan perhitungan XOR untuk *array state* hasil *inverse sub bytes* dengan *key schedule ke-13*. Hasil *add* round key pada *round* 1 dapat dilihat pada gambar 4.31.

4B	5D	6D	8E
6A	1E	F1	C6
E8	FB	81	28

14	30	26	74

Gambar 4.31 Hasil *Round* 1 *add Round Key*

4.5.2.7 Round 1 *Inverse Mix Columns*

Pada langkah *inverse mix columns* dilakukan perhitungan *multiplication* untuk *array state* hasil *add round key* dengan *circulant* matriks. Hasil *mix columns* dapat dilihat pada Gambar 4.32.

$$\begin{bmatrix} 14 & 11 & 13 & 09 \\ 09 & 14 & 11 & 13 \\ 13 & 09 & 14 & 11 \\ 11 & 13 & 09 & 14 \end{bmatrix} \begin{bmatrix} 4B & 5D & 6D & 8E \\ 6A & 1E & F1 & C6 \\ E8 & FB & 81 & 28 \\ 14 & 30 & 26 & 74 \end{bmatrix} = \begin{bmatrix} DC & D1 & 2B & 04 \\ F9 & D3 & 98 & 4B \\ DE & 7C & B1 & 8A \\ 26 & F6 & 39 & D1 \end{bmatrix}$$

Gambar 4.32 Hasil Round 1 Inverse Mix Columns

{14}.{4B}+{11}.{6A}+{13}.{E8}+{09}.{14}={DC}

{14}.{5D}+{11}.{1E}+{13}.{FB}+{09}.{30}={D1}

{14}.{6D}+{11}.{F1}+{13}.{81}+{09}.{26}={2B}

{14}.{8E}+{11}.{C6}+{13}.{28}+{09}.{74}={04}

Perhitungan dilakukan pada semua sampai pada *hexadecimal* terakhir sehingga diperoleh hasil sesuai pada Gambar 4.32.

4.5.2.8 Round 14

Pada perhitungan *round* ke-2 sampai dengan ke-14 dilakukan perhitungan yang sama untuk *inverse shift rows*, *inverse sub bytes*, tambah *round key*, dan *inverse mix columns*.

4.5.3 Proses Perhitungan PSNR(*Peak Signal to Noise Ratio*)

Dalam perhitungan PSNR kita akan menghitung perbedaan noise dari citra awal dan citra yang dihasilkan. Informasi dari kedua citra tersebut diperlihatkan dalam Table 4.6

Tabel 4.7 citra awal dan citra output

Nama	Citra	Dimensi	keterangan
Citra 10x10		10x10 pixel	Citra awal
Citra 10x10-LSB		10x10 pixel	Citra setelah disisipi

Sedangkan perbedaan intensitas warna antara citra awal dan citra output(‘) akan ditunjukkan oleh table 4.7

Tabel 4.8 perbedaan intensitas warna

Indeks pixel	R (dec)	G (dec)	B (dec)	R’ (dec)	G’ (dec)	B’ (dec)
(0,0)	200	23	150	200	22	151
(0,1)	221	27	174	222	28	174
(0,2)	231	35	192	230	34	192
(0,3)	234	35	193	234	34	192
(0,4)	228	11	175	228	10	175
(0,5)	225	0	159	225	0	158
(0,6)	230	13	185	231	13	186
(0,7)	223	20	176	224	20	175
(0,8)	222	45	181	222	46	180
(0,9)	221	39	174	220	37	175
(1,0)	208	0	155	207	0	155
(1,1)	208	13	160	207	12	160
(1,2)	217	27	169	217	26	169
(1,3)	232	23	186	231	23	186
(1,4)	224	0	154	224	0	154
(1,5)	224	0	140	224	0	141

(1,6)	226	1	170	225	0	171
(1,7)	219	18	169	210	18	168
(1,8)	222	46	178	222	45	178
(1,9)	223	45	180	223	45	180
(2,0)	221	1	170	220	2	171
(2,1)	222	15	172	222	14	172
(2,2)	219	19	167	219	18	167
(2,3)	224	12	166	224	11	165
(2,4)	223	0	119	223	0	120
(2,5)	212	0	97	212	0	97
(2,6)	216	0	121	216	0	120
(2,7)	206	0	134	206	0	135
(2,8)	222	36	175	222	36	175
(2,9)	221	30	177	221	30	177
(3,0)	230	5	171	231	6	171
(3,1)	229	0	158	229	0	158
(3,2)	224	0	130	224	0	131
(3,3)	215	0	120	215	0	121
(3,4)	166	29	55	165	27	54
(3,5)	173	37	60	173	37	61
(3,6)	192	0	76	191	1	76
(3,7)	202	0	109	203	1	109
(3,8)	212	3	128	212	3	128
(3,9)	219	1	159	218	2	158
(4,0)	236	40	190	236	41	189
(4,1)	230	0	143	231	0	142
(4,2)	224	12	117	223	11	116
(4,3)	182	63	84	183	64	85
(4,4)	170	144	52	169	145	53
(4,5)	168	129	71	167	130	72
(4,6)	176	81	87	176	80	86

(4,7)	184	0	57	185	0	57
(4,8)	200	0	84	201	0	85
(4,9)	217	0	131	217	0	131
(5,0)	236	40	190	236	40	189
(5,1)	236	33	188	236	34	189
(5,2)	235	0	162	236	0	163
(5,3)	217	176	139	217	175	138
(5,4)	215	174	139	215	174	139
(5,5)	200	161	135	200	160	136
(5,6)	200	47	110	201	46	110
(5,7)	198	0	80	198	0	80
(5,8)	209	0	120	208	1	121
(5,9)	220	1	157	220	1	156
(6,0)	233	43	189	222	44	190
(6,1)	234	37	183	233	36	184
(6,2)	232	0	150	231	1	153
(6,3)	235	0	141	236	1	140
(6,4)	233	0	148	232	1	147
(6,5)	234	0	149	234	1	148
(6,6)	220	0	112	220	0	112
(6,7)	201	0	91	201	0	91
(6,8)	205	0	127	205	0	127
(6,9)	217	26	162	217	26	162
(7,0)	235	29	193	235	29	193
(7,1)	233	2	182	233	2	182
(7,2)	234	0	158	234	0	158
(7,3)	232	0	158	232	0	158
(7,4)	231	0	153	231	0	153
(7,5)	230	0	128	230	0	128
(7,6)	222	0	136	222	0	136
(7,7)	215	0	120	215	0	120

(7,8)	211	6	142	211	6	142
(7,9)	213	25	160	213	25	160
(8,0)	236	25	193	236	25	193
(8,1)	232	0	175	232	0	175
(8,2)	230	0	171	230	0	171
(8,3)	232	0	174	232	0	174
(8,4)	230	8	171	230	8	171
(8,5)	224	0	156	224	0	156
(8,6)	223	0	154	223	0	154
(8,7)	219	0	144	219	0	144
(8,8)	221	1	154	221	1	154
(8,9)	223	14	154	223	14	154
(9,0)	234	0	179	234	0	179
(9,1)	232	0	164	232	0	164
(9,2)	233	0	182	233	0	182
(9,3)	232	0	184	232	0	184
(9,4)	227	0	174	227	0	174
(9,5)	221	0	163	221	0	163
(9,6)	221	0	161	221	0	161
(9,7)	219	0	158	219	0	158
(9,8)	222	0	161	222	0	161
(9,9)	220	5	172	220	5	172

*nilai RGB citra output ditandai dengan (‘)

Perhitungan PSNR :

$$\text{PSNR} = 20\text{xLog } 10 \left(\frac{255}{\sqrt{\frac{1}{(10x10)}[(200-200)+(23-22)+(150-151)+\cdots+(172-172)]^2}} \right) \text{db}$$

PSNR = 65.63144392872866 db

BAB V

IMPLEMENTASI

Pada bab ini akan dijelaskan implementasi dari seluruh proses yang sudah dirancang pada bab sebelumnya.

5.1 Lingkungan Implementasi

Implementasi merupakan proses transformasi representasi rancangan ke dalam bahasa pemrograman yang dapat dimengerti oleh komputer. Pada bab ini, lingkungan implementasi yang akan dijelaskan meliputi lingkungan implementasi perangkat keras dan perangkat lunak.

5.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan dan pengujian sistem kriptografi dan steganografi ini adalah sebuah notebook dengan spesifikasi sebagai berikut :

1. Monitor : 13,3”
2. CPU : Intel(R) Core(TM) i3 CPU M 370 @ 2.40GHz, 2399 Mhz, 2 Core(s), 4 Logical Processor(s)
3. Hard Disk : 320 GB
4. Memori : 2 GB
6. Perangkat Masukan : Keyboard, Mouse

Perangkat keras ini akan difungsikan sebagai tempat perangkat lunak untuk penelitian dijalankan.

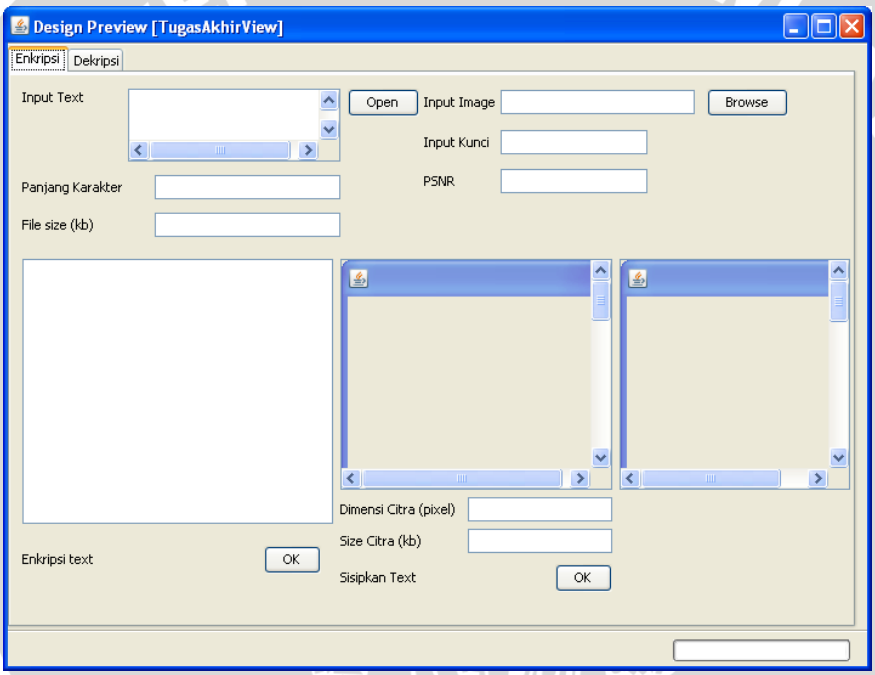
5.1.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan sistem kriptografi dan steganografi ini adalah :

1. Sistem operasi *Windows 7™ Ultimate 32-bit* sebagai tempat aplikasi dijalankan.
2. *Netbeans IDE 6,9* sebagai *programming software development* dalam pembuatan sistem kriptografi dan steganografi.

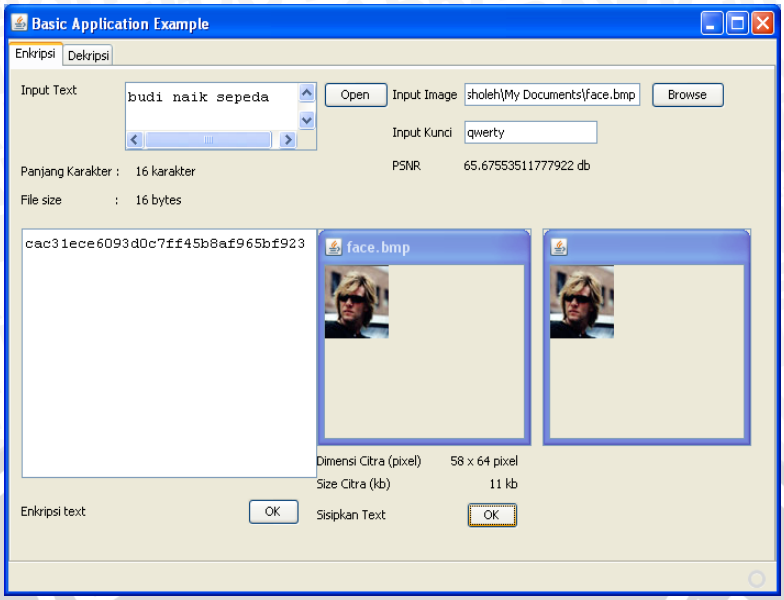
5.2 Implementasi Perangkat Lunak

Berdasarkan rancangan antar muka yang dikemukakan pada Bab IV maka dihasilkan antar muka yang ditunjukkan pada gambar 5.1



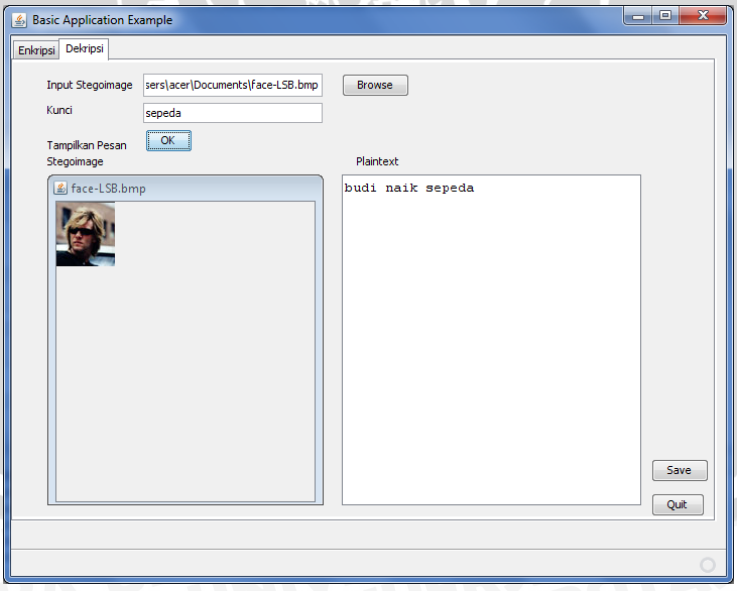
Gambar 5.1 Antar Muka

Untuk melakukan proses enkripsi *user* harus melakukan *input text* dengan langsung melakukan *input* pada *text area* yang disediakan atau melakukan klik pada *button open* untuk membuka *file text*. Setelah melakukan *input text user* melakukan *input* kunci pada *text area* yang disediakan. Kemudian *user* melakukan *input image* dengan melakukan klik pada *button browse* untuk memilih gambar. Setelah semua *input* dilakukan *user* melakukan klik pada *button* untuk enkripsi teks kemudian klik pada *button* untuk sisipkan teks. Jika semua proses tersebut berjalan maka akan tampak seperti gambar 5.2



Gambar 5.2 Antar Muka Enkripsi

Untuk proses dekripsi user harus memilih gambar hasil peyisipan ciphertext yang memiliki nama berakhiran –LSB dengan cara melakukan klik pada tombol browse, kemudian melakukan input kunci pada teks area yang disediakan. Untuk menjalankan dekripsi user melakukan klik pada tombol tampilkan pesan. Jika proses berjalan maka akan tampak seperti pada gambar 5.3.



Gambar 5.3 Antar Muka dekripsi

BAB VI

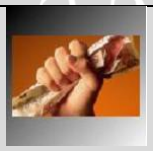
PENGUJIAN DAN ANALISIS

Pada bab ini akan dijelaskan strategi pengujian, hasil pengujian dan analisis terhadap hasil dari pengujian yang dilakukan.

6.1 Strategi Pengujian

Pengujian terhadap perangkat lunak dibagi menjadi tiga, yaitu pengujian fungsionalitas perangkat lunak, pengujian kinerja perangkat lunak dan pengujian ketahanan citra steganografi. Dalam pengujian, semua *file* citra mempunyai *format bitmap* dengan kedalaman 24 bit. Penjelasan lebih lengkap mengenai berkas citra yang akan di uji dapat dilihat pada Table 6.1.

Tabel 6.1 Daftar Berkas Citra *Bitmap* 24 bit

No	Nama File	Citra	Resolusi
1	Face.bmp		58x64
2	Jovi.bmp		123x103
3	Koper.bmp		128x128
4	Hand.bmp		139x138
5	Globe.bmp		144x144

Pada penelitian ini ada beberapa teks yang akan disisipkan ke dalam *file* citra. Setiap *file* citra akan di uji menggunakan tiga teks yang berbeda. Pada Tabel 6.2 ditunjukkan teks yang akan digunakan dalam penelitian kriptografi dan steganografi ini.

Tabel 6.2 Daftar Teks untuk Penelitian Kriptografi dan Steganografi

No	Nama File	Isi Pesan	Jumlah Karakter
1	Teks 1.txt	Abhimata Ar Rasyiid Mahasiswa Ilmu Komputer Angkatan 07	55
2	Teks 2.txt	Steganografi Ciphertext AES 256 Pada Citra Digital Menggunakan Metode Least Significant Bit (LSB)	97
3	Teks 3.txt	neovascular glaukoma adalah visi-dan mata-penyakit yang mengancam yang membawa prognosis yang buruk. Memahami penyebab neovascular glaukoma dengan diagnosis dini dan pengobatan adalah penting untuk positif hasilnya. Ada banyak pilihan pengobatan bedah yang tersedia, namun memiliki hasil terbaik, memutuskan teknik bedah untuk penting digunakan.	344
4	Teks 4.txt	Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka. Apabila ternyata didalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).	751
5	Teks 5.txt	Perkembangan teknologi yang begitu pesat memungkinkan manusia untuk dapat berkomunikasi dan saling bertukar data atau informasi secara jarak jauh. Seiring dengan kemudahan tersebut kebutuhan akan keamanan terhadap kerahasiaan informasi yang dipertukarkan semakin meningkat. Begitu banyak pengguna seperti departemen pertahanan, perusahaan atau individu–individu tidak ingin informasi yang disampaikannya diketahui oleh orang lain. Oleh karena itu dikembangkanlah cabang ilmu yang	1061

		mempelajari tentang cara–cara pengamanan data. Salah satu cara dalam melakukan pengamanan data adalah dengan menggunakan kriptografi. Istilah kriptografi sudah sangat dikenal dalam dunia pengamanan data. Kriptografi merupakan ilmu sekaligus seni untuk menjaga keamanan pesan [SCH-96]. Kriptografi berbasis pada algoritma pengkodean data informasi yang mendukung kebutuhan dari dua aspek keamanan informasi, yaitu *secrecy* (perlindungan terhadap kerahasiaan data informasi) dan *authenticity* (perlindungan terhadap pemalsuan dan pengubahan informasi yang tidak diinginkan) [BUD-10].	

6.2 Hasil Pengujian

Berdasarkan perancangan pengujian yang telah dijelaskan pada bab 4, maka hasil pengujian akan dijelaskan menjadi beberapa subbab pengujian, yaitu hasil pengujian fungsionalitas perangkat lunak dan hasil pengujian kinerja perangkat lunak.

6.2.1 Hasil Pengujian Fungsionalitas Perangkat Lunak

Pengujian fungsi proses steganografi terdapat dua proses yaitu fungsi penyisipan dan fungsi penguraian. Pengujian fungsi penyisipan dinyatakan berhasil apabila proses penyisipan teks ke dalam *file* citra tidak mengalami pesan kegagalan eksekusi dari perangkat lunak. Pengujian untuk fungsi penguraian kembali teks pesan dinyatakan berhasil jika pesan dapat diperoleh kembali, meskipun teks hasil penguraian belum tentu sama dengan teks pesan yang asli. Hasil pengujian fungsionalitas perangkat lunak untuk proses steganografi ditunjukkan tabel 6.3.

Tabel 6.3 Hasil Pengujian Fungsionalitas Perangkat Lunak Proses Steganografi

No	Proses Steganografi			
	File Citra	Pesan	Penyisipan	Penguraian
1	Face.bmp	Teks 1.txt	Berhasil	Berhasil
2	Face.bmp	Teks 2.txt	Berhasil	Berhasil
3	Face.bmp	Teks 3.txt	Berhasil	Berhasil
4	Face.bmp	Teks 4.txt	Berhasil	Berhasil
5	Face.bmp	Teks 5.txt	Berhasil	Berhasil

6	Jovi.bmp	Teks 1.txt	Berhasil	Berhasil
7	Jovi.bmp	Teks 2.txt	Berhasil	Berhasil
8	Jovi.bmp	Teks 3.txt	Berhasil	Berhasil
9	Jovi.bmp	Teks 4.txt	Berhasil	Berhasil
10	Jovi.bmp	Teks 5.txt	Berhasil	Berhasil
11	Koper.bmp	Teks 1.txt	Berhasil	Berhasil
12	Koper.bmp	Teks 2.txt	Berhasil	Berhasil
13	Koper.bmp	Teks 3.txt	Berhasil	Berhasil
14	Koper.bmp	Teks 4.txt	Berhasil	Berhasil
15	Koper.bmp	Teks 5.txt	Berhasil	Berhasil
16	Hand.bmp	Teks 1.txt	Berhasil	Berhasil
17	Hand.bmp	Teks 2.txt	Berhasil	Berhasil
18	Hand.bmp	Teks 3.txt	Berhasil	Berhasil
19	Hand.bmp	Teks 4.txt	Berhasil	Berhasil
20	Hand.bmp	Teks 5.txt	Berhasil	Berhasil
21	Globe.bmp	Teks 1.txt	Berhasil	Berhasil
22	Globe.bmp	Teks 2.txt	Berhasil	Berhasil
23	Globe.bmp	Teks 3.txt	Berhasil	Berhasil
24	Globe.bmp	Teks 4.txt	Berhasil	Berhasil
25	Globe.bmp	Teks 5.txt	Berhasil	Berhasil

Pengujian fungsionalitas proses kriptografi dilakukan dengan melakukan proses enkripsi pesan, fungsi enkripsi pesan dinyatakan berhasil apabila didapatkan *ciphertext* yang berupa karakter *hexadecimal* dan tidak terdapat pesan kegagalan eksekusi dari perangkat lunak. Kemudian dilakukan proses dekripsi dengan memberikan kunci yang benar dan kunci yang salah. Hal ini dilakukan untuk menguji jika diberikan kunci yang salah apakah pesan akan tetap kembali seperti semula. Pengujian untuk fungsi dekripsi pesan dengan perangkat lunak dinyatakan berhasil jika pesan dapat didekripsi menjadi karakter *ASCII*, walaupun belum tentu sama dengan pesan asli. Pada Tabel 6.4 merupakan hasil pengujian fungsionalitas perangkat lunak untuk proses kriptografi.

Tabel 6.4 Hasil Pengujian Fungsionalitas Perangkat Lunak Proses Kriptografi

No	Proses Kriptografi				
	Pesan Asli	Ciphertext	Key Untuk Enkripsi	Key Untuk Dekripsi	Hasil Dekripsi
1	Teks 1.txt	59f782da150c1e54776a6ca41bccbaa40f4da8241f615307c8007f0d27eb0d457280329ffeef6491e2b1488429288279ac1a4e47c0464e226590921c6bc43220	qwerty	qwerty	Abhimata Ar Rasyiid Mahasiswa Ilmu Komputer Angkatan 7
2	Teks 1.txt	59f782da150c1e54776a6ca41bccbaa40f4da8241f615307c8007f0d27eb0d457280329ffeef6491e2b1488429288279ac1a4e47c0464e226590921c6bc43220	qwerty	asd	îâ{¿s)FŽ]àRdÍì9 m¢O.o!¥ðí ¬uÔ5L}uÅÔ±íí¨ Á«<œª ÚÈ\ðôÝ §òÐË?5^Ý_
3	Teks 2.txt	aa5fcf2ea15b6490a5ea83f4b5618897f4c77e3f723406223e4bba6356ed1f88d6ee5551727f7b319b850e1c4c3cf7222ccf0cab3b2c79d185ddf50211b099f9db5036f0c5875a1b93ede5a2524305603564709e9dceafdc23244c4cb1d3e1b1cc89d4a9c4a2647154398542aab086cc	qwerty	qwerty	Steganografi Ciphertext AES 256 Pada Citra Digital Menggunakan Metode Least Significant Bit (LSB)
4	Teks 2.txt	aa5fcf2ea15b6490a5ea83f4b5618897f4c77e3f723406223e4bba6356ed1f88d6ee5551727f7b319b850e1c4c3cf7222ccf0cab3b2c79d185ddf50211b099f9db5036f0c5875a1b93ede5a2524305603564709e9dceafdc23244c4cb1d3e1b1cc89d4a9c4a2647154398542aab086cc	qwerty	abcdef	Ükv YMN|Š`½µ~g6 • N • æÃGþAW,ðç D^ì(" ö□ É‡ëÞw y¥Œ 9°§Öˆ6×Q>N-fef^"BŒ;A□ ÿîøŠO-Ðd¼ƒF‹:5‹²Ôº • ænøG›Ë1lU"û
5	Teks 3.txt	76b8ac34eb4278d8722ad7f7858bff88d85e9cbb08d28a1c987a489a989d20a45ee621d87e2017af9cfb2294355193dba93318988e8e61fb53d0e0cf90868f10115b6aec1bb5d2aa0432ce93444ced35	qwerty	qwerty	neovascular glaukoma adalah visi-dan mata-penyakit yang mengancam yang membawa prognosis yang buruk.

		74f7e188a67bfaadb9b6dde3c87f4a627e36eb9b41707c3f1bbe868c0c0a2aaa04319cf5be4dc8f0f5c43628833f5df18053f0cc62df81899a68e3d9a996655bc49d758595d4d72b1d1d42adcd3872571caf2e9d405e3e3fa1231793014797c2927f7e94458ca1c12b76cb222fce1e49a159cb3e36d8bc8eb28b000ea01df7c323bc178080c09bb2f823ed17ced11506ab2aeeab95dca512ef58d4c1cadc3c0a96dfab8fa2f16fa6b063307b1a19ac8a5246ce2485521de9f7eb86d18b96ae885d965366cd3a75396b217f89b83e81795ef581e53691503c2c10dfb00ad73a5434d6cafa2222f033987beb1f3bc53f305b8af01a5a050af65ba8643624c6a20448e66c2ed60fad12dc5758cdc458f948			Memahami penyebab neovascular glaukoma dengan diagnosis dini dan pengobatan adalah penting untuk positif hasilnya. Ada banyak pilihan pengobatan bedah yang tersedia, namun memiliki hasil terbaik, memutuskan teknik bedah untuk penting digunakan.
6	Teks 3.txt	76b8ac34eb4278d8722ad7f7858bff88d85e9cbb08d28a1c987a489a989d20a45ee621d87e2017af9cfb2294355193dba93318988e8e61fb53d0e0cf90868f10115b6aec1bb5d2aa0432ce93444ced3574f7e188a67bfaadb9b6dde3c87f4a627e36eb9b41707c3f1bbe868c0c0a2aaa04319cf5be4dc8f0f5c43628833f5df18053f0cc62df81899a68e3d9a996655bc49d758595d4d72b1d1d42adcd3872571caf2e9d405e3e3fa1231793014797c2927f7e94458ca1c12b76cb222fce1e49a159cb3e36d8bc8eb28b000ea01df7c323bc178080c09bb2f823ed17ced11506ab2aeeab95dca512ef58d4c1cadc3c0a96dfab8fa2f16fa6b0633	qwerty	Arcy	‘íŠ<ö›š¹®¼»B¹Ÿé°”i Â|ÿcMÿÓƒž Â2ê£‘G–Û3bVœWj-ZúÅó–KÕ«A^²Þ

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

		07b1a19ac8a5246ce2485521de9f7eb86d18b96ae885d965366cd3a75396b217f89b83e81795ef581e53691503c2c10dfb00ad73a5434d6cafa2222f033987beb1f3bc53f305b8af01a5a050af65ba8643624c6a20448e66c2ed60fad12dc5758cdc458f948			
7	Teks4.txt	f0a6097baf9bd55b74bbf649435b4753be654551b36513f16cb67d5824dfd7920cf722c9a3af9088a09dfd9d2a44c563c3c140ccb9186c39092f7a9b6d6c6757486bff30dbe091cc1bced3d38a03707b4cdf9a9d77df89c57a4d5a824bc94776bbf969de206beb2b2a77c46eee1ed29f0daa13ce8623175a32c630167f7465240480447ff0ff08b8f78ad12aaad530ddf71ba8b8f6b590340477fb60cbf314bfb5a5cb2772cb2fb39d2000cbf46efde4f1ff7f4c54efa385a5a34814c2f7ede08ad4d4bcece818171f6aff7fc6dc104be26cbaef86f6b8bee0d397daaad101b6f28faf8d0bf6fd7ed2ae298be94e192e9ea9ac56750c738b4d2a0a36c259f0565c3f11c219300ca74dc7cdcfee02683c604107ab2d6d322e60e27115330b0be9588dc3f94580cde5487e1f3989d249f24f79a38d78fde81dd1d096937486d9870a6285553f65dea206ab24b66ad0d2f7d42994ae446df6b7ff6c5ba8a2989cb9fd2109d75b9679f1158f5cc2c9b4edf666bff3efcc246e3cda7a5250776e493a48646519783c2988225c7aae81cf1b6a029dc8d9508f826fd39068c0fb6dce08f8838abc6d93153	qwerty	qwerty	Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka. Apabila ternyata didalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya

		3986fa534480419a42b20ec045a75e36b4b6ea93b47af8c9c74f28dea1b879010cca1f1f44f60db4c11b64a015b240a889414b280edd92608b267c76974059fdca4d83eafb7e8ec0e62370e194f2f02f9918668e81d2cd813da9426990234ea57945ae3561066a60b9f7c816bf6ecf20dbda0e5e65f0fa70b97cc2cba7993a5d6f1dc8ff88e38a58900260c41680a311b0656ff3f22443ae92de8e3d14d466d697ae951470eacade79284aa57a507cdec6ec2c0d23253b5bcb2e71d625fdb3883593eb4833f62bb9144f42e5c381b7bb541413f4b20a070b4f046e581133a85f3fc7727734662a801cc0dd2f95f96468d8a289bea0dc71d87ae187d1a5fd7e74aee12ad4d6c1bd92c3f2915b6fe1df679c433cfb558592121b6522defe87427215cdabbaf11fe4fc333ec56773e7c3288504500be61fd733939d2df36b4dded6b2783dc61bf730cc5			peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).
8	Teks4.txt	f0a6097baf9bd55b74bbf649435b4753be654551b36513f16cb67d5824dfd7920cf722c9a3af9088a09dfd9d2a44c563c3c140ccb9186c39092f7a9b6d6c6757486bff30dbe091cc1bced3d38a03707b4cdf9a9d77df89c57a4d5a824bc94776bbf969de206beb2b2a77c46eee1ed29f0daa13ce8623175a32c630167f7465240480447ff0ff08b8f78ad12aaad530ddf71ba8b8f6b590340477fb60cbf314bfb5a5cb2772cb2fb39d2000cbf46efde4f1ff7f4c54efa385a5a34814c2f7ede08ad4d4bcece818171f6af	qwerty	qwrt	å‡„¢PBœa¾éý"` ÞvvÞn`:9oSŸŸ_ œyg´¹}ñ!ù¯b Öd XV?ë‡ro¾-Æ„ÿW;u×AÑÎQz:×³@ô*8éíæn*—oyW# Œ5N8,š|K/-¦%W*–ˆÕ>ÜÆ±Âê□ªð □ Å´«1h¼ü³Š~¥¤vq£ÿÃ • ÚIw¸êßÿc–á«TÖs×ŸcØoâ¹aQXãwiÇÂPÂ‰;õcã\å7Fp"U • fûNê'L¿§kQàëîÞÒ*û®oîíÉYOäÇ□ ry‰¸–ùJ,m\@£Ûx"

		f7fc6dc104be26cbaef86 f6b8bee0d397daaad101 b6f28faf8d0bf6fd7ed2a e298be94e192e9ea9ac5 6750c738b4d2a0a36c25 9f0565c3f11c219300ca 74dc7cdcfee02683c604 107ab2d6d322e60e2711 5330b0be9588dc3f9458 0cde5487e1f3989d249f 24f79a38d78fde81dd1d 096937486d9870a6285 553f65dea206ab24b66a d0d2f7d42994ae446df6 b7ff6c5ba8a2989cb9fd2 109d75b9679f1158f5cc 2c9b4edf666bff3efcc24 6e3cda7a5250776e493a 48646519783c2988225 c7aae81cf1b6a029dc8d 9508f826fd39068c0fb6 dce08f8838abc6d93153 3986fa534480419a42b2 0ec045a75e36b4b6ea93 b47af8c9c74f28dea1b8 79010cca1f1f44f60db4c 11b64a015b240a88941 4b280edd92608b267c7 6974059fdca4d83eafb7 e8ec0e62370e194f2f02f 9918668e81d2cd813da9 426990234ea57945ae35 61066a60b9f7c816bf6e cf20dbda0e5e65f0fa70b 97cc2cba7993a5d6f1dc 8ff88e38a58900260c41 680a311b0656ff3f2244 3ae92de8e3d14d466d69 7ae951470eacade79284 aa57a507cdec6ec2c0d2 3253b5bcb2e71d625fdb 3883593eb4833f62bb91 44f42e5c381b7bb54141 3f4b20a070b4f046e581 133a85f3fc7727734662 a801cc0dd2f95f96468d 8a289bea0dc71d87ae18 7d1a5fd7e74aee12ad4d 6c1bd92c3f2915b6fe1df 679c433cfb558592121b 6522defe87427215cdab baf11fe4fc333ec56773e 7c3288504500be61fd73			WäX-K- â.Ü¿œ¨àjœØò8- “`U½(SWPáQ4¶ ^„á@/#Hl

		3939d2df36b4dded6b2783dc61bf730cc5			
9	Teks5.txt	9b857aaa536a91070907a83ada178f8ea7e8c9b5dfb3facccafa0a9b39b339f4f1e85590cf313bfbb2898f472742c87feb355460bd46598218a1671b80f83efd667562dad1abc9c73e5479d6d2b37919722736e03b482467ab3aface5141ac9d76f8d4b928e42ad254f80dc8b5ac47f52105c2191872438eb8a000dc6f7c8d716ade67b0f5bfc3db20ab1a126ce9093b5e520b310646fd40a34f7661888e55e49d52853fa4eba178c05c75604621af0bcf16aff5e1b685fd8c9b7598c3f7501fdd53911704926df314488da937b3384b13d1e7c35ebfc1f554746a5945a88df3b529b58c0f088631a0fc361b700701d0e69ba150dbb737b9fad82a74d46f4ebb7567f16f16f42df9269b59352398b5fa97307a67be26b441704989d2b1ac3f930c21fac03dc21d84db50fa6145ac22d0ce4e5e7f04ee4d4ae5bdba3edbf1aacf040c05ff4ba418244774cbd537c649c4f7aeca5811cce803147b84542922f342da21d2128022a8d708c45a7fb2bf06eed0dddf18d7f1f42916ed301d56e4da0587b0b326fd9eaf6c7b1d2ba696eea16e07471627cd1748eef48c50063298222d4a04c04e33338280d95afea9276d30195799b42056a5eebeb80cc87539d1e27c61a4a11dabcf9fb3df5e7c9bd05770b62bbe51d059813f1a917d158fda5e39b745d3048b0370cf2b09a86c989ea2769fc995a089af31cfd2b9b62de5d679	qwerty	qwerty	Perkembangan teknologi yang begitu pesat memungkinkan manusia untuk dapat berkomunikasi dan saling bertukar data atau informasi secara jarak jauh. Seiring dengan kemudahan tersebut kebutuhan akan keamanan terhadap kerahasiaan informasi yang dipertukarkan semakin meningkat. Begitu banyak pengguna seperti departemen pertahanan, perusahaan atau individu–individu tidak ingin informasi yang disampaikannya diketahui oleh orang lain. Oleh karena itu dikembangkanlah cabang ilmu yang mempelajari tentang cara–cara pengamanan data. Salah satu cara dalam melakukan pengamanan data adalah dengan menggunakan kriptografi. Istilah kriptografi sudah sangat dikenal dalam dunia pengamanan

		8672be7a5aa8f6a0a13a 769681cfc15b03034d49 c8bfb07a5d4f84fd75a0f 60b1b57d214ad010536 e7bb0cc1bddb7a60d9d2 9139fe6388fc20d88856 47a008246ea8b0c36273 c8153ea1ff3631b4183b 4bc9348faa42cf52bd3e 616f7deeb6a84db453b1 d1714358c81b4eacf52c da732fdb717daee3d199 2826519ee75ec71737c0 e89d49ca851a6e9d65b9 a89c15ed42b1ab19a3d9 293859623cde29ec5b50 bfa93965d383f3770740 05a55e57d932406136d 2140a6a23e11dd6a4a11 a2fc13d0cc8b343c66f4 38e7514e5e587cc8101b d5c37b9bc0b5a4b5bedc 22a8db13471906d444f9 2bed3b11d82befbbd3f2 f8081931a491e70a99a0 bc44e71cf159e3b902d8 1f9155a081112ed0e3a5 7cb9112a35d64461775 63e85484f7a35cf01029 b6ba4b37221de15f9ecd 54c6b533609a3f80bdf7 877c33ce057955426d99 de039f8cfb0f11126531 dcdeb78dac1fa74f8072 da6ebfec60385842d142 57297c66efbcde881449 da1be863131ca5bc0805 82b815e03a44b0cd6c88 02935c2ae6ef78b50d5e 1a3cdc68bbeb95f9aee2 ba6ca953472e531a8557 05493f3bd12704c43dcd 02d2f163f2a7c10fdc02e 782aac0cffd1299184f89 3bccfea7abd0c9a24096 a7c529e4e18ade5e3430 670d626b5476e878c2e0 128b3ef036907ec99444 4a464163c77e5c46123b fe29690d82635dbd5369 f9eb93cecf9936238c61 1ac568516c7f615b934e abfe9c4f88bce8ec517c6			data. Kriptografi merupakan ilmu sekaligus seni untuk menjaga keamanan pesan [SCH-96]. Kriptografi berbasis pada algoritma pengkodean data informasi yang mendukung kebutuhan dari dua aspek keamanan informasi, yaitu secrecy (perlindungan terhadap kerahasiaan data informasi) dan authenticity (perlindungan terhadap pemalsuan dan pengubahan informasi yang tidak diinginkan) [BUD-10].

		57a2942ddc300d923e70 002f2d6e525efaa3d8af7 394d4605d90bdc4c96e2 15			
10	Teks5.txt	9b857aaa536a91070907 a83ada178f8ea7e8c9b5 dfb3facccafa0a9b39b33 9f4f1e85590cf313bfbb2 898f472742c87feb3554 60bd46598218a1671b8 0f83efd667562dad1abc 9c73e5479d6d2b37919 722736e03b482467ab3a face5141ac9d76f8d4b9 28e42ad254f80dc8b5ac 47f52105c2191872438e b8a000dc6f7c8d716ade 67b0f5bfc3db20ab1a12 6ce9093b5e520b310646 fd40a34f7661888e55e4 9d52853fa4eba178c05c 75604621af0bcf16aff5e 1b685fd8c9b7598c3f75 01fdd53911704926df31 4488da937b3384b13d1 e7c35ebfc1f554746a59 45a88df3b529b58c0f08 8631a0fc361b700701d0 e69ba150dbb737b9fad8 2a74d46f4ebb7567f16f 16f42df9269b59352398 b5fa97307a67be26b441 704989d2b1ac3f930c21 fac03dc21d84db50fa61 45ac22d0ce4e5e7f04ee 4d4ae5bdba3edbf1aacf0 40c05ff4ba418244774c bd537c649c4f7aeca581 1cce803147b84542922f 342da21d2128022a8d7 08c45a7fb2bf06eed0dd df18d7f1f42916ed301d 56e4da0587b0b326fd9e af6c7b1d2ba696eea16e 07471627cd1748eef48c 50063298222d4a04c04e 33338280d95afea9276d 30195799b42056a5eebe b80cc87539d1e27c61a4 a11dabcf9fb3df5e7c9bd 05770b62bbe51d05981 3f1a917d158fda5e39b7 45d3048b0370cf2b09a8	qwerty	qwe	*Ð9‰R Q”Ç}aíóß $@„’bõ÷tµ"5ïÞ MÚ&“ hè¢o ýâ4A¯¨c} &>^Ü™ Ðk#c;™ •«KÐ×'ÈŠÎ¡hÆ(¦ œ¥AøcÜŒÓµ ÏØURÊ+?Í¢|-IŸÙ € }¸®\y9$4#“ª□¹ □ üµSf¥î7ÖCó/¡ ö °Dáß({äÙ r$}c¸tÏL•• ¸- ôé l$_©nÀÞï¾// n‰ïÊHe×¶®ËK ÓÄâ - U¢6x¾>U¹Uó,~u ú†íž • 7ZÑ.ZClc •+=¢ðæJk@gž m FåI% • i,¸dëèÑwc Ý+L›³>§×¨C|Cªfò ìÔs-\/- d†Cë:Kj¬ïÈ— Ø›]|N.üT¸^Fú1v XndO‘p&BIl— ¨4-{#=ýL\babýÚ` “cŒkIÕ°— ý»“¥QQ£9&‘êEq I„l• {PnœZ2Ðí.Ycò›˜ r°H{ð(‹Y…ÆW*ƒ* 4X©/'ktž ÇS– ©}ËDÛmos£nÕí <Æ3àš"Š;ð±åñÖ XÚôØ÷- QªTæÚïÉÍôÐãã~ ù • "µ»"ãÍŒT™ o9õœ^cù¡*×ÈGº Z*†¼a…±®aèûÎ $˜<<5øÌ|

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

		6c989ea2769fc995a089 af31cfd2b9b62de5d679 8672be7a5aa8f6a0a13a 769681cfc15b03034d49 c8bfb07a5d4f84fd75a0f 60b1b57d214ad010536 e7bb0cc1bddb7a60d9d2 9139fe6388fc20d88856 47a008246ea8b0c36273 c8153ea1ff3631b4183b 4bc9348faa42cf52bd3e 616f7deeb6a84db453b1 d1714358c81b4eacf52c da732fdb717daee3d199 2826519ee75ec71737c0 e89d49ca851a6e9d65b9 a89c15ed42b1ab19a3d9 293859623cde29ec5b50 bfa93965d383f3770740 05a55e57d932406136d 2140a6a23e11dd6a4a11 a2fc13d0cc8b343c66f4 38e7514e5e587cc8101b d5c37b9bc0b5a4b5bedc 22a8db13471906d444f9 2bed3b11d82befbbd3f2 f8081931a491e70a99a0 bc44e71cf159e3b902d8 1f9155a081112ed0e3a5 7cb9112a35d64461775 63e85484f7a35cf01029 b6ba4b37221de15f9ecd 54c6b533609a3f80bdf7 877c33ce057955426d99 de039f8cfb0f11126531 dcdeb78dac1fa74f8072 da6ebfec60385842d142 57297c66efbcde881449 da1be863131ca5bc0805 82b815e03a44b0cd6c88 02935c2ae6ef78b50d5e 1a3cdc68bbeb95f9aee2 ba6ca953472e531a8557 05493f3bd12704c43dcd 02d2f163f2a7c10fdc02e 782aac0cffd1299184f89 3bccfea7abd0c9a24096 a7c529e4e18ade5e3430 670d626b5476e878c2e0 128b3ef036907ec99444 4a464163c77e5c46123b fe29690d82635dbd5369 f9eb93cecf9936238c61			

		1ac568516c7f615b934e abfe9c4f88bce8ec517c6 57a2942ddc300d923e70 002f2d6e525efaa3d8af7 394d4605d90bdc4c96e2 15			

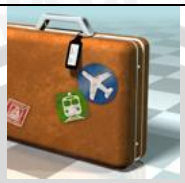
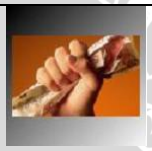
6.2.2 Hasil Pengujian Kinerja Perangkat Lunak

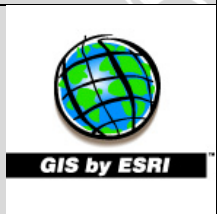
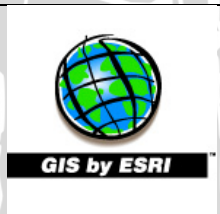
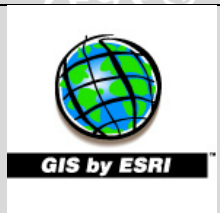
Pada Tabel 6.5 ditunjukkan hasil pengujian kinerja perangkat lunak dengan memasukkan *input* kombinasi pesan dan citra yang berbeda. Hal ini dilakukan untuk melihat berapa nilai PSNR yang dihasilkan.

Tabel 6.5 Hasil Pengujian Kinerja Perangkat Lunak Teks

No	Pesan	File Citra Asal	Citra Stegano	Nilai PSNR
1	Teks1.txt	 Face.bmp	 Face-LSB.bmp	57.38
2	Teks2.txt	 Face.bmp	 Face-LSB.bmp	56.08
3	Teks3.txt	 Face.bmp	 Face-LSB.bmp	53,52
4	Teks4.txt	 Face.bmp	 Face-LSB.bmp	52.04
5	Teks5.txt	 Face.bmp	 Face-LSB.bmp	51.50

6	Teks1.txt	 Jovi.bmp	 Jovi-LSB.bmp	61.69
7	Teks2.txt	 Jovi.bmp	 Jovi-LSB.bmp	61.31
8	Teks3.txt	 Jovi.bmp	 Jovi-LSB.bmp	58.44
9	Teks4.txt	 Jovi.bmp	 Jovi-LSB.bmp	57.56
10	Teks5.txt	 Jovi.bmp	 Jovi-LSB.bmp	56.99
11	Teks1.txt	 Koper.bmp	 Koper-LSB.bmp	71,76

12	Teks2.txt	 Koper.bmp	 Koper-LSB.bmp	69,61
13	Teks3.txt	 Koper.bmp	 Koper-LSB.bmp	65,08
14	Teks4.txt	 Koper.bmp	 Koper-LSB.bmp	60,93
15	Teks5.txt	 Koper.bmp	 Koper-LSB.bmp	59,12
16	Teks1.txt	 Hand.bmp	 Hand-LSB.bmp	67,73
17	Teks2.txt	 Hand.bmp	 Hand-LSB.bmp	66,01
18	Teks3.txt	 Hand.bmp	 Hand-LSB.bmp	61,06

19	Teks4.txt	 Hand.bmp	 Hand-LSB.bmp	57,96
20	Teks5.txt	 Hand.bmp	 Hand-LSB.bmp	56,74
21	Teks1.txt	 Globe.bmp	 Globe-LSB.bmp	67,96
22	Teks2.txt	 Globe.bmp	 Globe-LSB.bmp	65,49
23	Teks3.txt	 Globe.bmp	 Globe-LSB.bmp	61,24
24	Teks4.txt	 Globe.bmp	 Globe-LSB.bmp	57,96

25	Teks5.txt	 Globe.bmp	 Globe-LSB.bmp	56,75

Pada Tabel 6.6 ditunjukkan hasil pengujian kinerja perangkat lunak dengan kombinasi panjang kunci yang berbeda.

Tabel 6.6 Hasil Pengujian Kinerja Perangkat Lunak Kunci

No	Pesan	File Citra Asal	Ciphertext	Karakter	Kunci	Karakter	Nilai PSNR
1	Budi naik sepeda	 Face.bmp	9ec6c2d2177d7f161a364c48aeb38957	32	bhima	5	66.19
2	Budi naik sepeda	 Face.bmp	08be6740658eea0ed6b27d2a1ae9fa98	32	bhimailkom	10	65.99
3	Budi naik sepeda	 Face.bmp	1f0e75332cf07e940c0bfba376d4f992	32	ujianbhimailkom	15	65.74
4	Budi naik sepeda	 Face.bmp	f07ae0429608ed63ac3bafbe649cd8fd	32	ujianbhimailkombagus	20	65.41
5	Budi naik sepeda	 Face.bmp	c70541cc6f2ac50ebbc008ca438cc2	32	komputerabhimataararasyiid	25	66.19

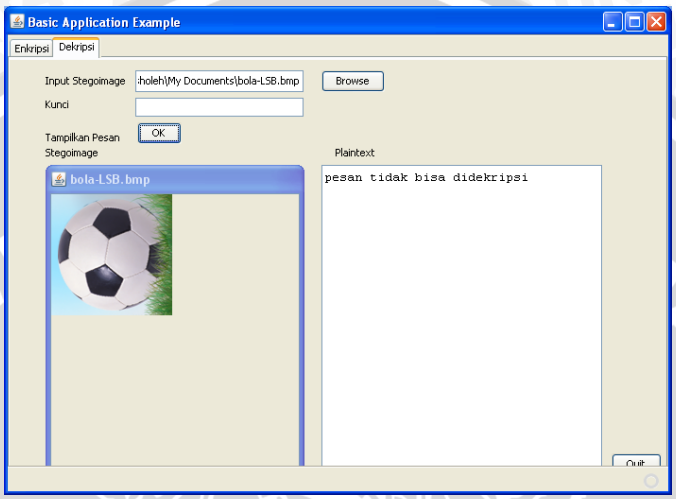
			a4				
6	Budi naik sepeda	 Face.bmp	4d437ceb39612337892fbbf581dd9555	32	abhimataarrasyiidnim0710963026	30	65.97
7	Budi naik sepeda	 Face.bmp	bc4ee14bce333099a8bc8f03e14a458d	32	programteknologiinformasidanilmu	32	66.02

Uji coba yang ditunjukkan oleh Tabel 6.5 adalah uji coba menyisipkan teks dengan panjang yang berbeda ke dalam gambar yang sama. Didapatkan jumlah karakter pesan yang disisipkan mempengaruhi nilai PSNR. Dapat dilihat pada Tabel 6.5 tejadi penurunan nilai PSNR seiring dengan pertambahan jumlah pesan yang disisipkan. Nilai PSNR diperoleh dengan membandingkan perbedaan intensitas warna antara citra asli dengan citra hasil steganografi. Metode LSB yang digunakan untuk menyisipkan pesan merubah bit terakhir pada nilai RGB citra dengan pesan yang akan disisipkan sehingga nilai PSNR disini dipengaruhi oleh panjang pesan dan bit terakhir pada citra yang dirubah. Sedangkan uji coba yang diperlihatkan oleh Table 6.6 yaitu dengan memasukkan kunci dengan panjang yang berbeda dalam pesan yang sama dan gambar yang sama didapatkan bahwa jumlah karakter kunci tidak mempengaruhi nilai PSNR. Kunci dengan jumlah karakter yang lebih sedikit tidak menjamin nilai PSNR lebih besar dari kunci dengan jumlah karakter yang lebih banyak. Hal ini dikarenakan jumlah karakter kunci tidak mempengaruhi jumlah karakter ciphertext yang dihasilkan tetapi mempengaruhi *ciphertext* yang dihasilkan. Perbedaan nilai PSNR ini disebabkan terdapat bit *ciphertext* yang sama dengan bit terakhir RGB citra sehingga bit terakhir RGB citra tidak berubah dan menyebabkan nilai PSNR menjadi lebih besar.

6.2.3 Hasil Pengujian Ketahanan Citra Steganografi

Dalam proses pengujian ketahanan citra steganografi terhadap manipulasi citra dipilih citra Face.bmp dan Teks3.txt seperti manipulasi yang akan dilakukan

yaitu *cropping* dari bawah sebesar 10 %, 50 %, 90%, rotasi 90 derajat searah jarum jam, rotasi 90 derajat berlawanan dengan jarum jam dan penambahan efek sephia. Parameter keberhasilan ini ditentukan dari pesan hasil dekripsi yang dihasilkan jika pesan yang dihasilkan sama dengan Teks3.txt maka besar error adalah 0% namun jika muncul pesan seperti pada Gambar 6.1 maka error adalah 100%.



Gambar 6.1 Error pesan tidak bias didekripsi

Hasil dari pengujian ini dapat dilihat pada tabel 6.6

Tabel 6.7 Hasil Pengujian Ketahanan Citra Steganografi

No	Jenis Serangan	Perubahan Citra	Hasil Dekripsi	Error
1	Cropping 10%		neovascular glaukoma adalah visi-dan mata-penyakit yang mengancam yang membawa prognosis yang buruk. Memahami penyebab neovascular glaukoma dengan diagnosis dini dan pengobatan adalah penting untuk positif hasilnya. Ada banyak pilihan pengobatan bedah yang tersedia, namun memiliki hasil terbaik, memutuskan teknik bedah untuk penting digunakan.	0%
2	Cropping 50%		neovascular glaukoma adalah visi-dan mata-penyakit yang mengancam yang membawa	0%

			prognosis yang buruk. Memahami penyebab neovascular glaukoma dengan diagnosis dini dan pengobatan adalah penting untuk positif hasilnya. Ada banyak pilihan pengobatan bedah yang tersedia, namun memiliki hasil terbaik, memutuskan teknik bedah untuk penting digunakan.	
3	Cropping 90%		Pesan tidak dapat didekripsi	100%
4	Rotasi 90 searah jarum jam		Pesan tidak dapat didekripsi	100%
5	Rotasi 90 berlawanan jarum jam		Pesan tidak dapat didekripsi	100%
6	Penambahan efek sephia		Pesan tidak dapat didekripsi	100%

Berdasarkan pengujian yang ditunjukkan oleh Tabel 6.6 manipulasi pada citra steganografi menyebabkan kegagalan dalam proses dekripsi pesan dikarenakan nilai rgb dari citra akan berubah dan menyebabkan tidak ditemukannya penanda. Pada proses *cropping* dari bawah citra tidak terjadi kegagalan pada cropping 10% dan 50% dikarenakan pesan yang disisipkan tidak meliputi semua lsb citra dan *cropping* tidak sampai memotong pesan yang disisipkan, namun hal ini tidak akan berlaku jika pesan yang disisipkan meliputi semua lsb citra kita dapat menghitung jumlah karakter pesan yang dapat disisipkan max_karakter_pesan = panjang_citra x lebar_citra x 3 .

6.3 Analisis Hasil Pengujian

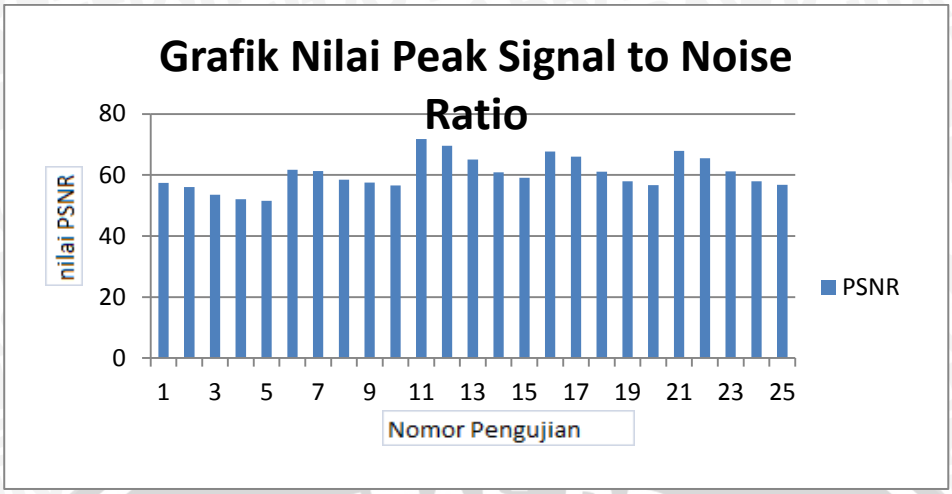
Hasil uji yang didapatkan akan di analisis lebih lanjut untuk melihat apakah hasilnya sesuai dengan dengan tujuan yang ingin dicapai dalam penelitian ini. Selain itu akan diambil kesimpulan terhadap hasil uji yang didapat.

6.3.1 Analisis Hasil Uji Fungsionalitas Perangkat Lunak

Hasil uji menunjukkan bahwa perangkat lunak pengamanan data kriptografi dan steganografi yang telah dibuat pada penelitian ini telah memenuhi kebutuhan perangkat lunak yang telah dipaparkan pada Bab IV. Hal ini dibuktian dengan keberhasilan perangkat lunak dalam melakukan proses enkripsi dan dekripsi pesan, serta proses penyisipan pesan dan penguraian pesan dari citra tanpa mengalami pesan kegagalan ataupun *error* pada saat eksekusi perangkat lunak. Walaupun pada kasus tertentu beberapa karakter pada teks hasil dekripsi berbeda dengan teks asli. Seperti pada Teks1.txt pada saat dilakukan proses dekripsi angka nol pada kalimat Abhimata Ar Rasyiid Mahasiswa Ilmu Komputer Angkatan 07 menghilang hal ini diakibatkan pada saat proses penghilangan padding semua angka nol akan dihilangkan sehingga jika kita akan menyisipkan pesan yang mengandung angka nol maka pada waktu pesan dikembalikan angka nol akan hilang.

6.3.2 Analisis Hasil Uji Kinerja Perangkat Lunak

Dalam analisis hasil uji kinerja perangkat lunak dapat disimpulkan bahwa ekstraksi pesan dari citra steganografi yang dihasilkan oleh perangkat lunak cukup baik, hal ini dibuktikan dengan nilai PSNR yang cukup tinggi yaitu diatas angka 50, dimana jika nilai PSNR semakin tinggi maka citra asli akan semakin mirip dengan citra steganografi. Nilai PSNR dipengaruhi oleh jumlah pesan yang akan disisipkan pada citra dan perubahan bit terakhir pada citra. Karena jika pesan yang disisipkan semakin banyak maka kemungkinan perubahan bit terakhir pada citra semakin besar namun tidak menutup kemungkinan bit pesan yang akan disisipkan sama dengan bit terakhir pada nilai RGB citra. Hasil uji coba dari tabel 6.5 ditampilkan melalui grafik pada Gambar 6.1 yang mana ditunjukkan terjadinya penurunan nilai PSNR seiring dengan pertambahan jumlah karakter pesan yang disisipkan.



Gambar 6.2 Grafik Nilai *Peak Signal to Noise Ratio*

Grafik rata-rata *Peak Signal Noise Ratio* (PSNR) pada Gambar 6.1 menunjukkan bahwa tingkat noise tertinggi terdapat pada nomor pengujian kelima yaitu pada citra Face.bmp dengan pesan Teks5.txt. Hal ini dikarenakan Teks5.txt memiliki pesan dengan jumlah karakter paling banyak jika dibandingkan dengan Teks1.txt, Teks2.txt, Teks3.txt dan Teks4.txt selain itu citra Face.bmp memiliki dimensi ukuran yang lebih kecil jika dibandingkan Jovi.bmp, Koper.bmp, Hand.bmp dan Globe.bmp sehingga hal ini menyebabkan perubahan hampir pada semua bit terakhir pada RGB citra Face.bmp.

Komposisi warna pada citra tidak berpengaruh terhadap nilai PSNR karena pesan akan melewati proses penyandian yang mana akan dirubah menjadi bentuk biner sebelum disisipkan ke dalam citra. Apabila terdapat kasus bit – bit *ciphertext* identik atau sama dengan bit – bit terakhir pada RGB citra yang bersesuaian maka dalam kasus ini citra hasil steganografi akan identik atau sama dengan citra asli.

6.3.3 Analisis Hasil Uji Ketahanan Citra Steganografi

Dari hasil uji ketahanan citra steganografi dapat disimpulkan, citra stegtanografi tidak tahan pada manipulasi citra. Proses croping, rotasi, penambahan efek sephia dapat menyebabkan pesan tidak dapat didekripsi. Namun dalam proses *cropping* terdapat pengecualian selama proses *cropping* tidak memotong pesan dan penanda maka pesan masih bisa didekripsi.

6.3.4 Analisis Umum Hasil Uji

Perangkat lunak yang dihasilkan dinilai telah bekerja relatif baik dan cukup memuaskan meskipun pada kasus tertentu terjadi perbedaan beberapa karakter antara pesan hasil penguraian dengan pesan yang asli. Citra steganografi tidak tahan jika dilakukan manipulasi citra.

Nilai PSNR dipengaruhi oleh jumlah bit terakhir yang dirubah dan bit – bit bit – bit *ciphertext* yang dihasilkan. Semakin banyak jumlah bit terakhir yang tidak dirubah dan semakin banyak kesamaan antara bit *ciphertext* dengan bit pada file citra maka nilai PSNR akan semakin besar.

BAB VII
KESIMPULAN DAN SARAN

7.1 Kesimpulan

Setelah melakukan skripsi ini maka disimpulkan :

1. Teknik pengamanan data steganografi melalui kriptografi dapat diimplementasikan pada sebuah perangkat lunak. Teknik kriptografi algoritma AES 256 diproses terlebih dahulu sehingga menghasilkan *ciphertext*, *ciphertext* kemudian disisipkan ke dalam citra penampung dengan teknik steganografi LSB.
2. Kualitas citra yang dihasilkan relatif sama dengan citra aslinya secara kasat mata dan hal ini juga ditunjukkan oleh grafik rata – rata PSNR yang memiliki nilai diatas 50 db. Faktor yang mempengaruhi kualitas citra yang dihasilkan adalah persentase perubahan jumlah bit terakhir pada citra, semakin sedikit bit yang dirubah maka citra hasil steganografi akan semakin mirip dengan citra aslinya.
3. Citra hasil steganografi tidak tahan jika dilakukan manipulasi citra seperti rotasi, penambahan efek sephia dan cropping yang melewati pesan yang tersisipi.

7.2 Saran

Saran yang dapat penulis berikan setelah mengerjakan skripsi ini adalah sebagai berikut :

1. Agar mendapatkan nilai PSNR lebih besar dapat digunakan citra penampung dengan ukuran yang lebih besar.

DAFTAR PUSTAKA

[ARI-09] Ariyus, D. 2009. *Keamanan Multimedia*. Andi Offset. Yogyakarta

[ARY-92] Arymurthy, Aniati Murni, dan Setiawan, Suryana. 1992. *Pengantar Pengolahan Citra*. PT Elex Media Komputindo. Jakarta.

[BUD-10] Budiawan, Dedy. 2010.*Perancangan Aplikasi Kriptography – Advanced Encryption Standar.* UPN. Surabaya

[COX-08] Cox, I. J., Miller, M. L., Bloom, J. A., Fridrich, J. dan Kalker, T. 2008. *Digital Watermarking and Steganography. 2nd Edition*. Morgan Kaufmann. Burlington.

[CVE-04] Cvejic, Nedeljko. 2004. *Algorithms for Audio Watermarking and Steganography*. Oulu University Press. Oulu

[ELV-10] Elvirman, M Zikki.2010. *Steganografi Ciphertext Pada Citra Digital Menggunakan LSB*.UB.Malang.

[FED-01] Federal Information Processing Standard No.197, 26 November 2001. *Encryption Standard*. UPN. Surabaya.

[KEK-08] Kekre, H. B., Athawale, A. dan Halarnkar, P. N. 2008. *Increased Capacity of Information Hiding in LSB's Method for Text and Image*. International Journal of Electrical, Computer, and Systems Engineering 2(4): 246 – 251.

[KRE-04] Krenn, J.R. 2004. *Steganography and Steganalysis*. http://www.krenn.nl/univ/ cry/steg/ article.pdf. Diakses tanggal 16 Mei 2009.

[KUR-04] Kurniawan, Yusuf. 2004. *Kriptografi Keamanan Internet dan Jaringan Komunikasi*. Informatika. Bandung.

[LAK-09] Laksono, S. 2009. Skripsi : *Kompresi Citra Digital Menggunakan Fuzzy Learning Vector Quantization*. Universitas Brawijaya, Malang.

[LES-10] Lestriandoko, N. H. 2006. *Pengacakan Pola Steganografi untuk Meningkatkan Keamanan Penyembunyian Data Digital*. http://journal.uii.ac.id/index.php/ Snati/article/view/1538/1313. Diakses tanggal 30 Mei 2010.

[MUN-04] Munir, R. 2004. *Kuliah IF5054 Kriptografi :Tipe dan Mode Algoritma Simetri*. Institut Teknologi Bandung. Bandung.

[MUN-06] Munir, R. 2006. *Kriptografi*. Informatika. Bandung.

[NOV-08] Novrina, Indah Kusuma W.2008. *Peningkatan Pengamanan Pesan Rahasia Dengan Teknik Penyisipan Pada Citra Digital Menggunakan Pendekatan Least Significant Bit(LSB)*.Universitas Gunadarma.Jakarta.

[PRI-12] Primadian MP, Shandy.2012. *Enkripsi Data Menggunakan Modifikasi Algoritma AES Rijndael dan MD5*. UB. Malang

[RAH-01] Raharjo. 2001. *Keamanan sistem informasi Berbasis Internet*. PT Insan Komunika/Infonesia. Bandung.

[SCH-96] Schneier, B. 1996. *Applied Cryptography, Second Edition: Protocols, Algorthms, and Source Code in C*. John Wiley & Sons, Inc.

[SEL-96] Sellars, Duncan, 1996, *An Introduction To Steganography*. John Wiley & Sons, Inc.

[YUN-09] Yuniati voni, dkk.2009 .*Enkripsi Dan Dekripsi Dengan Algoritma Aes 256 Untuk Semua Jenis File*.UKDW. Yogyakarta.