PENYEMBUNYIAN DATA TEKS PADA GAMBAR MENGGUNAKAN ALGORITMA *INTERNATIONAL DATA ENCRYPTION ALGORITHM* DAN *END OF FILE* BERBASIS ANDROID

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Teknik Pada
Jurusan Teknik Informatika

Oleh:

RIVALZA FAHLEVI

11351105131

UIN SUSKA RIAU

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU

PEKANBARU

2019

LEMBAR PERSETUJUAN

PENYEMBUNYIAN DATA TEKS PADA GAMBAR MENGGUNAKAN ALGORITMA *INTERNATIONAL DATA ENCRYPTION ALGORITHM* DAN *END OF FILE* BERBASIS ANDROID

TUGAS AKHIR

Oleh:

RIVALZA FAHLEVI
11351105131

Telah diperiksa dan disetujui sebagai laporan tugas akhir
di Pekanbaru pada tanggal 5 November 2019

Pembimbing,

Pizaini, S.T., M.Kom.
NIK. 130 517 107

LEMBAR PENGESAHAN

PENYEMBUNYIAN DATA TEKS PADA GAMBAR MENGGUNAKAN ALGORITMA *INTERNATIONAL DATA ENCRYPTION ALGORITHM* DAN *END OF FILE* BERBASIS ANDROID

TUGAS AKHIR

Oleh

RIVALZA FAHLEVI

11351105131

Telah dipertahankan di depan sidang dewan penguji

Sebagai salah satu syarat untuk memperoleh gelar sarjana Teknik Informatika

Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau

di Pekanbaru, pada tanggal 5 November 2019

Pekanbaru, 5 November 2019

Mengesahkan,

Dekan,	**Ketua Jurusan,**
Dr. Ahmad Darmawi, M.Ag.	**Dr. Elin Haerani, S.T., M.Kom.**
NIP. 19660604 199203 1 004	**NIP. 19810523 200710 2 003**

DEWAN PENGUJI

Ketua	: Dr. Elin Haerani, S.T., M.Kom.
Sekretaris	: Pizaini, S.T., M.Kom.
Penguji I	: Iwan Iskandar, M.T.
Penguji II	: Reski Mai Candra, S.T., M.Sc.

LEMBAR HAK ATAS KEKAYAAN INTELEKTUAL

Tugas Akhir yang tidak diterbitkan ini terdaftar dan tersedia di Perpustakaan Universitas Islam Negeri Sultan Syarif Kasim Riau adalah terbuka untuk umum dengan ketentuan bahwa hak cipta pada penulis. Referensi kepustakaan diperkenankan dicatat, tetapi pengutipan atau ringkasan hanya dapat dilakukan seizin penulis dan harus disertai dengan kebiasaan ilmiah untuk menyebutkan sumbernya.

Penggandaan atau penerbitan sebagian atau seluruh Tugas Akhir ini harus memperoleh izin dari Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau. Perpustakaan yang meminjamkan Tugas Akhir ini untuk anggotanya diharapkan untuk mengisi nama, tanda peminjaman dan tanggal pinjam.

LEMBAR PERNYATAAN

Dengan ini saya menyatakan bahwa dalam Tugas Akhir ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan didalam daftar pustaka.

Pekanbaru, 5 November 2019

Yang membuat pernyataan,

RIVALZA FAHLEVI

11351105131

LEMBAR PERSEMBAHAN

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

وَمَن يَتَّقِ اللَّهَ يَجْعَل لَّهُۥ مِنْ أَمْرِهِۦ يُسْرًا ﴿٤﴾

Dan barang -siapa yang bertakwa kepada Allah, niscaya Allah menjadikan baginya kemudahan dalam urusannya. – (Q.S At-Talaq: 4)

Alhamdulillah dengan Ridha-Mu ya Allah, pada akhirnya Tugas Akhir ini dapat terselesaikan dengan baik. Terima kasihku untukmu kupersembahkan untuk Ayah dan Ibu dan adikku tercinta yang tidak pernah henti-hentinya memberikan semangat, doa, nasehat dan serta pengorbanan yang tak tergantikan.

PENYEMBUNYIAN DATA TEKS PADA GAMBAR MENGGUNAKAN ALGORITMA *INTERNATIONAL DATA ENCRYPTION ALGORITHM* DAN *END OF FILE* BERBASIS ANDROID

RIVALZA FAHLEVI

11351105131

Tanggal Sidang: 5 November 2019

Jurusan Teknik Informatika

Fakultas Sains dan Teknologi

Universitas Islam Negeri Sultan Syarif Kasim Riau

ABSTRAK

Ilmu kriptografi dan steganografi banyak digunakan dalam menjaga keamanan pesan atau teks. pada penelitian ini akan menggabungkan Algoritma *IDEA* dan Algoritma *EOF* untuk memperkuat keamanan data teks. Pada proses kriptografi *IDEA* pesan akan dienkripsi dan pesan akan disisipkan kedalam gambar, Pada proses pengujian dan implementasi berbasis android maka di dapatkanlah hasil dari aplikasi ini kecepatan enkripsi dan dekripsi dengan rata-rata waktu enkripsi 0,829 detik dan dekripsi 0,090, kemudian mendapatkan nilai *MSE* 0,000003834 dan *PSNR* 390490863320.91 untuk mengetahui kualitas citra tersebut , dan terjadi perubahan kapasitas data gambar setelah disisipkan pesan, perubahan histogram juga terjadi pada pesan yang disisipkan pada gambar, pengujian brute force juga dilakukkan untuk menjaga ketahanan pesan, dengan melakukkan enkripsi dan dekripsi dengan algoritma *IDEA* dan *EOF* dan didukung dengan pengujian *MSE* dan *PSNR* akan menjaga keamanan data pesan.

Kata Kunci : Android, Dekripsi, Enkripsi, *EOF, IDEA*

HIDDEN DATA HIDDING IN IMAGES USING INTERNATIONAL DATA ENCRYPTION ALGORITHM AND END OF FILE BASED ON ANDROID

RIVALZA FAHLEVI

11351105131

Session Date: 5 November 2019

Informatics Engineering
Faculty of Science and Technology
Sultan Syarif Kasim State Islamic University Riau

ABSTRACT

Cryptography and steganography are widely used in message or text security. This research will launch IDEA Algorithm and EOF Algorithm for text data security agreement. In the IDEA cryptographic process the message will be encrypted and the message will be inserted into the image, the Android-based testing and implementation process is obtained from this application the encryption and decryption speed with an average delivery of encryption 0.829 seconds and 0.090 decryption, then get the MSE value of 0, 000003834 and PSNR 390490863320.91 to find out the quality of the image, and handle changes in image data after removing the message, histogram changes also occur in messages disseminated in the image, try brute force also done for messages, by using EOF slides and supported by MSE and PSNR testing request message data security.
Keywords : *Android, Decryption, Encryption, EOF, IDEA*

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Assalamu'alaikum wa rahmatullahi wa barakatuh.

Alhamdulillahi rabbil'alamin, Segala puji hanya bagi Allah Subhanahu wata'ala, karena atas rahmat dan karunia-Nya penulis mampu menyelesaikan Laporan Tugas Akhir ini dengan lancar. Tidak lupa ucapan shalawat beriring salam untuk Baginda Rasulullah Muhammad Shalallahu'alaihi wa sallam, yang telah mengajarkan kita untuk menjadi manusia yang beradab dan beliau juga telah menjadi inspirator lahirnya zaman yang penuh dengan ilmu pengetahuan seperti saat ini. Allahumma sholli'alaa sayyidina Muhammad wa'alaa ali sayyidina Muhammad.

Tugas Akhir ini disusun sebagai salah satu syarat untuk mendapatkan gelar kesarjanaan pada jurusan Teknik Informatika Universitas Islam Negeri Sultan Syarif Kasim Riau. Banyak saran, pengalaman, pengetahuan, bimbingan dan dukungan menuju kebaikan yang penulis terima dari berbagai pihak hingga penulisan laporan ini dapat diselesaikan. Maka dari itu pada kesempatan ini penulis ingin menyampaikan ucapan terima kasih kepada:

1. Bapak Prof. Dr. KH. Akhmad Mujahidin, M.Ag, selaku Rektor Universitas Islam Negeri Sultan Syarif Kasim Riau.
2. Bapak Dr. Ahmad Darmawi, M.Ag., selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau.
3. Ibu Dr. Elin Haerani, S.T, M.Kom, selaku Ketua Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau
4. Bapak Pizaini, S.T, M.Kom, selaku Pembimbing Tugas Akhir yang telah meluangkan waktu untuk membimbing penulis dan membagi ilmu, wawasan serta saran dan arahan sehingga penulis dapat memulai dan menyelesaikan penyusunan dan penulisan tugas akhir dengan baik.

5. Bapak Iwan Iskandar, M.T, selaku Penguji I Tugas Akhir dan Pembimbing Akademik yang telah memberikan masukan dan arahan kepada penulis dalam penyusunan tugas akhir serta membimbing penulis dalam tiap semester.
6. Bapak Reski Mai Candra, S.T, M.Sc, selaku Penguji II Tugas Akhir yang telah memberikan masukan dan arahan kepada penulis dalam penyusunan tugas akhir ini.
7. Seluruh Bapak dan Ibu yang mengajar di jurusan Teknik Informatika Universitas Islam Negeri Sultan Syarif Kasim Riau.
8. Kedua orang tua penulis, Ayahanda M. Reza Fahlevi dan Ibunda Nurbaiti dan adik Gilang Ramadhan Fahlevi yang selalu memberi semangat dan Do'a untuk penyelesaian Tugas Akhir.
9. Kepada Teman seperjuangan yang selama pengerjaan tugas akhir ini selalu bersama dalam suka maupun duka, Jeprianto, Aldi Wiratama, Yusuf Abdillah Putra, Syarif Hidayatullah, Jasriadi, Fakhrial Irsyadi, Panji yo. Walaupun kalian agak nakal tetapi kalian menemani dalam pengerjaan tugas akhir ini.
10. Teman seperjuangan TIF'13 E (TIF E WOLES) yang telah memberikan dukungan moral maupun materi selama ini.
11. Seluruh teman-teman dari jurusan Teknik Informatika angkatan 2013 yang memberikan semangat, serta senior dan junior yang tidak mungkin disebutkan satu persatu.
12. Semua pihak yang terlibat langsung maupun tidak langsung dalam pelaksanaan tugas akhir ini yang tidak dapat disebutkan satu persatu.

Penulis menyadari bahwa dalam penulisan laporan ini masih banyak Penulis menyadari bahwa dalam penulisan laporan ini masih banyak kesalahan dan kekurangan, oleh karena itu kritik dan saran yang sifatnya membangun sangat penulis harapkan untuk kesempurnaan laporan ini, yang dapat disampaikan ke alamat email penulis rivalza.fahlevi@students.uin-suska.ac.id Penulis berharap

semoga laporan ini dapat memberikan sesuatu yang bermanfaat bagi siapa saja yang membacanya. Amin.

Wassalamu'alaikum wa rahmatullahi wa barakatuh

Pekanbaru, 5 November 2019

Rivalza Fahlevi

DAFTAR ISI

DAFTAR GAMBAR

Gambar **Halaman**

DAFTAR TABEL

Tabel **Halaman**

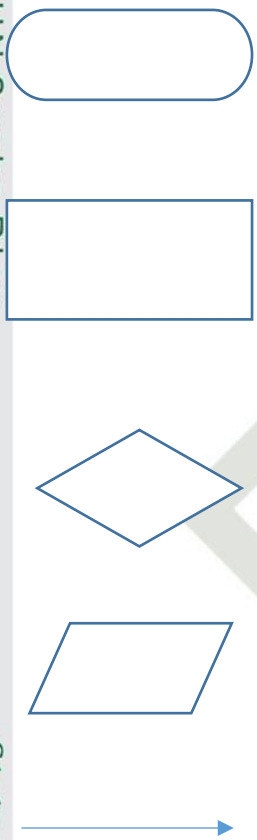
DAFTAR RUMUS

DAFTAR DIAGRAM

Diagram **Halaman**

DAFTAR SIMBOL

Flowchart

Gambar	Keterangan
	Terminator: Simbol terminator (mulai/selesai) merupakan tanda bahwa sistem akan dijalankan atau berkahir. Proses: Simbol yang digunakan untuk melakukan pemrosesan data baik oleh user maupun komputer (sistem). Verifikasi: Simbol yang digunakan untuk memutuskan apakah valid atu tidak validnya suatu kejadian. Data: Simbol yang digunakan utuk mendeskripsikan data input/ output yang digunakan. Arus Data: Simbol yang digunakan untuk menggambarkan arus data di dalam sistem.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Seiring semakin cepatnya perkembangan teknologi informasi saat ini didukung dengan pentingnya kebutuhan akan mendapatkan informasi. Keamanan suatu data ialah hal yang perlu diperhatikan dalam menjaga kerahasiaan informasi yang hanya boleh dilihat oleh orang-orang tertentu saja. Pengiriman data teks sudah tersebar luas didunia maya maupun nyata, dikarenakan data tersebut penting dan rahasia dan data teks tersebut dapat diketahui dengan mudah tanpa didukung dengan keamanan yang akan berdampak pada penyadapan informasi dan dapat diketahui oleh pihak-pihak yang tidak bertanggung jawab.

Untuk mengamankan data teks atau informasi ada beberapa metode keamanan data yang dapat digunakan untuk menjaga keamanan pesan dan dapat diterapkan pada berbagai media seperti teks, gambar, suara dan video, diantaranya dengan menggunakkan teknik Kriptografi IDEA, dan Steganografi EOF. Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya (Munir, 2006). Sedangkan steganografi adalah seni menyembunyikan pesan dalam data lain tanpa mengubah data yang ditumpanginya tersebut sehingga data yang ditumpanginya sebelum dan setelah proses penyembunyian hampir terlihat sama. algoritma kriptografi yang digunakan IDEA yaitu algoritma simetris yang beroperasi pada sebuah block pesan terbuka dengan lebar 64-bit dan metode steganografi end of file yang bertujuan merahasiakan keberadaan dari sebuah pesan tersembunyi atau sebuah informasi, sudah banyak penelitian yang dilakukkan pada kedua metode tersebut tetapi masih ada kelemahan pada enkripsi dan dekripsi. Oleh pada penelitian ini kedua metode tersebut akan digabungkan untuk memperkuat enkripsi dan dekripsi.

Pada penelitian yang telah dilakukan sebelumnya oleh (Tri Andriyanto, 2008) dengan penelitian perbandingan antara algoritma idea dan algoritma

blowfish pada penelitian ini menjelaskan tentang membandingkan kinerja algoritma IDEA dan Blowfish dalam hal kecepatan proses dan penggunaan memori pada saat proses enkripsi dan dekripsi suatu file. Terlihat dari penelitian tersebut bahwa algoritma IDEA lebih cepat dari algoritma Blowfish dan pemakaian memori kedua algoritma relatif sama. Tetapi penelitian tersebut tidak melihatkan kualitas data yang diperoleh.

Penelitian lainnya yang telah dilakukan sebelumnya oleh (Hendrawan, 2015) dengan penelitian metode pengamanan untuk metode pengamanan untuk file bertipe dokumen menggunakan kombinasi algoritma idea dan lsb . Pada penelitian ini membahas tentang penyisipan file ke dalam gambar dengan enkripsi file dengan idea dan disembunyikan ke gambar dengan metode lsb , tetapi jumlah data yang disisipkan terbatas.

Penelitian yang telah dilakukan sebelumnya oleh (Muslih & rachmawanto, 2016) dengan penelitian pengamanan file multimedia dengan metode steganografi end of file untuk menjaga kerahasian pesan, pada penelitian ini membahas tentang menyembunyikan file pada file induk tanpa ada perubahan yang signifikan terhadap file induk. Selain itu, file juga dapat diambil kembali untuk dilihat isi pesan atau informasi dengan jelas tanpa ada noise sedikitpun. Tetapi akan lebih baik digabungkan dengan metode kriptografi agar lebih aman.

Penelitian yang telah dilakukan sebelumnya oleh (Krisnawati, 2008) dengan penelitian metode least significant bit dan end of file untuk menyisipkan teks kedalam citra, pada penelitian ini menjelaskan tentang membandingkan tentang kelebihan dan kekurangan metode LSB dan EOF dalam penyisipkan pesan teks, Metode LSB pada saat penyisipan pesan jumlah karakter pesan sangat terbatas harus menyesuaikan besar citra dan besar pesan dikirim tetapi kelebihan ukuran citra tidak berubah. Sedangkan Metode EOF akan meletakkan pesan di akhir citra sehingga ukuran file akan bertambah besar, oleh karena itu pesan teks yang disisipkan tidak terbatas jumlahnya.

Untuk menjaga Keamanan informasi lebih terjamin keamanannya tentunya perlu suatu cara agar enkripsi file tidak mudah untuk dipecahkan oleh orang lain

serta hasil dari stego image tidak menimbulkan kecurigaan. Sehingga penulis melakukan penelitian untuk menjaga keamanan pesan agar tidak dibaca oleh pihak yang diinginkan maka perlu kriptografi untuk mengacak pesan dan steganografi sebagai penyembunyian pesan pada media yang sudah ditetapkan.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka didapatlah rumusan masalah yaitu, “Bagaimana Membangun Aplikasi Penyembunyian Data teks Pada Gambar Menggunakan Algoritma Kriptografi IDEA dan Steganografi End Of File Berbasis android”

1.3 Batasan Masalah

Dalam penulisan skripsi ini, yang menjadi ruang lingkup untuk batasan masalah adalah sebagai berikut:

1. Media yang digunakan berupa gambar dengan format .JPG .PNG
2. Data yang di enkripsi hanya berupa teks karakter

1.4 Tujuan Penelitian

Tujuan penelitian ini untuk untuk merancang serta membangun sistem atau aplikasi pengamanan data teks menggunakan teknik kriptografi algoritma *International Data Encryption Algorithm* (IDEA) dan teknik Steganografi dengan metode *End Of File* (EOF) berbasis android dalam melakukan enkripsi, proses histogram, proses MSE dan PSNR dan Dekripsi

1.5 Sistematika Penulisan

Berikut merupakan susunan sistematika penulisan laporan tugas akhir yang akan disusun sebagai berikut :

BAB I PENDAHULUAN

Berisi dasar-dasar dari penulisan atau deskripsi umum pada tugas akhir yang meliputi latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian dan sistematika penulisan

BAB II LANDASAN TEORI

Berisi penjelasan mengenai dasar-dasar teori, rujukan dan metode yang digunakan sebagai dasar dalam menerapkan metode *International Data Encryption Algorithm* (IDEA) dan algoritma *End Of File* (EOF), dan penelitian sebelumnya.

BAB III METODOLOGI PENELITAN

Menjelaskan metode pengerjaan skripsi, studi pustaka, perumusan masalah, analisis, perancangan, implementasi, pengujian dan kesimpulan

BAB IV ANALISA DAN PERANCANGAN

Bab ini menjelaskan tentang analisa dan perancangan dimulai dari analisa komponen system, analisa tampilan system, serta bagaimana algoritma *International Data Encryption Algorithm* (IDEA) metode *End Of File* (EOF) dan bekerja dalam system

BAB V IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi penjelasan tentang pengertian dan tujuan implementasi sistem, batasan implementasi, lingkungan implementasi, impelmentasi sistem dan pengujian.

BAB VI PENUTUP

Berisi tentang kesimpulan serta saran dari penelitian yang telah dilakukan.

BAB II
LANDASAN TEORI

2.1 Keamanan Data

Keamanan data pada saat ini sangat penting dimasa saat kini. Pertukaran data yang sering terjadi secara *offline* maupun *online* serta sangat rentan akan ancaman pencurian data. Hal ini merupakan pengaruh dari perkembangan yang sangat signifikan dalam teknologi informasi. Oleh karena itu terdapat beberapa cara untuk menjaga rahasia pada data yaitu kriptografi dan steganografi.

Keamanan data ialah perlindungan terhadap data di dalam suatu sistem untuk melawan terhadap otorisasi tidak sah, modifikasi, atau perusakan dan perlindungan sistem komputer terhadap penggunaan tidak sah atau modifikasi.

Kriptografi bertujuan memberikan layanan keamanan ada empat aspek utama dalam keamanan data dan informasi yaitu (Rachmawati & Candra, 2015):

1. Kerahasiaan (*Confidentiality*) yaitu Informasi dirahasiakan dari semua pihak yang tidak berwenang.
2. Keutuhan Data (*Integrity*) yaitu Pesan tidak berubah dalam proses pengiriman hingga pesan diterima oleh penerima.
3. Autentifikasi (*Authentication*) yaitu Kepastian terhadap identitas setiap entitas yang terlibat dan keaslian sumber data.
4. Nirpenyangkalan (*Nonrepudiation)* yaitu Setiap entitas yang berkumunikasu tidak dapat menolak atau menyangkal atas data yang telah dikirim atau diterima.

2.2 Kriptografi

Kriptografi berasal dari kata kripto dan grafi. Kripto berarti menyembunyikan, dan grafi yaitu ilmu. Kriptografi *(cryptography)* adalah suatu ilmu yang mempelajari suatu sistem penyandian untuk menjamin kerahasiaan dan keamanan data. Orang yang melakukan disebut Criptographer. Awal tahun 400 SM, bangsa Spartan di Yunani memanfaatkan kriptografi di bidang militer dengan menggunakan alat yang disebut scytale, yakni pita panjang berbahan daun

papyrus yang dibaca dengan cara digulungkan ke sebatang silinder. Sedangkan peradaban Cina dan Jepang menemukan kriptografi pada abad 15 M. Dan pada zaman Romawi Kuno, Julius Caesar ingin mengirimkan pesan rahasia pada seorang jendral di medan perang. Pesan tersebut harus dikirimkan melalui seorang prajurit, tetapi karena pesan tersebut mengandung rahasia, Julius Caesar tidak ingin pesan tersebut terbuka di tengah jalan. Julius Caesar mengacak isi pesan tersebut menjadi suatu pesan yang tidak dapat dipahami oleh siapapun, kecuali jendralnya.

Kriptografi adalah bidang ilmu pengetahuan yang mempelajari pemakaian persamaan matematika untuk melakukan proses penyandian data (Onno W.P., 2000). Kriptografi bertujuan untuk mengamankan isi data atau menjaga kerahasiaan informasi dari orang yang tidak berhak untuk mengetahui isi data tersebut.

2.2.1 Terminologi Kriptografi

Berikut beberapa istilah terminologi dalam kriptografi yang dikutip dari

1. Pesan, Plaintext, dan Ciphertext

   Pesan adalah data ataupun suatu informasi yang dapat dibaca dan dimengerti maknanya. Dan nama lain untuk pesan ialah plaintext, atau teks jelas. Ciphertext adalah suatu bentuk pesan yang bersandi. Disandikannya suatu pesan adalah agar pesan tersebut tidak dapat dimengerti oleh pihak lainnya. (Munir, 2006)

2. Pengirim dan Penerima

   Suatu aktivitas komunikasi data, akan melibatkan pertukaran antara dua entitas, yakni pengirim dan penerima. Pengirim adalah entitas yang mengirim pesan kepada entitas lainnya. Sedangkan penerima adalah entitas yang menerima pesan.Suatu pengiriman pesan, pengirim tentu menginginkan pesan dapat dikirim secara aman. Untuk mengamankannya, pengirim biasanya akan menyandikan pesan yang dikirimkan tersebut. (Munir, 2006)

3. Enkripsi dan Dekripsi

   Suatu proses untuk menyandikan plaintext menjadi ciphertext disebut enkripsi (encryption). Sedangkan proses pengembalian dari ciphertext manjadi plaintext dinamakan dekripsi (decription). Enkripsi dan dekripsi merupakan suatu pesan yang memetakan elemen-elemen antara kedua himpunan tersebut. Misalkan P adalah himpunan plaintext, dan C adalah himpunan ciphertext, maka fungsi enkripsi E memetakan P ke C, ditulis E(P) = C. Dan fungsi dekripsi D memetakan C ke P, ditulis D(C) = P. (Munir, 2006)

4. Cipher dan kunci

   Cipher bisa disebut juga algoritma kriptografi yaitu aturan untuk melakukan enchipering dan deciphering atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa cipher memerlukan algoritma yang berbeda untuk melaukan *enciphering* dan *deciphering*. Konsep metematis yang mendasari algoritma kriptografi adalah relasi anatara dua buah himpunan yaitu himpunan yang berisi elemen-elemen plainteks dan himpunan yang berisi *ciphertéksí*. (Munir, 2006)

5. Sistem kriptografi

   Kriptografi membentuk sebuah sistem yang dinamakan sistem kriptografi. Sistem kriptografi adalah kumpulan yang terdiri dari algoritma kriptografi, semua *plainteks* dan *ciperteks* yang mungkin dan kunci. Didalam sistem kriptografi *cipher* hanyalah salah satu komponen saja. (Munir, 2006)

6. Penyadap

   Penyadap adalah orang yang mencoba menangkap pesan selama ditransmisikan. (Munir, 2006)

7. Kriptanalis dan kriptologi

   Kriptanalis adalah ilmu dan seni untuk memecahkan *ciperteks* menjadi plainteks tanpa mengetahui kunci yang digunakan. Pelakunya disebut kriptanalis. Jika seorang kriptografer mentransformasikan plainteks menjadi *cipherteks* dengan suatu algoritma dan kunci maka sebaliknya

seseorang kriptanalis berusaha untuk memecahkan *cipherteks* untuk menemukan *plainteks.* (Munir, 2006)

2.2.2 Data Teks

Telah disebutkan dalam buku (Eriyanto., 2001), Teks hampir sama dengan wacana, bedanya kalau teks hanya bisa disampaikan dalam bentuk tulisan saja, sedangkan wacana bisa disampaikan dalam bentuk lisan maupun tertulis. sekumpulan karakter terdiri dari huruf-huruf,angka-angka(A-Z,a-z,0-9) dan simbol-simbol lainya seperti %,&,^,=,@,$,!,* dan lain-lain,dengan menggunakan kode ASCII setiap karakter dari text berjumlah 8-bit atau 1 byte.

Adapun Jenis Jenis Teks yang ada antara lain:

1. Teks Anekdot

   Teks Anekdot merupakan sebuah teks yang berisi peristiwa-peristiwa lucu, konyol, atau menjengkelkan sebagai akibat dari krisis yan ditanggapi dengan reaksi.

2. Teks Deskripsi

   Jenis teks yang menggambarkan keadaan (sifat, bentuk, ukuran, warna, dsb) sesuatu (manusia atau benda) secara individual dan unik.

3. Teks Diskusi

   Teks Diskusi merupakan sebuah yang berisi tinjauan terhadap sebuah isu dari dua sudut pandang yang berbeda, yaitu sisi yang mendukung dan menentang isu tersebut.

4. Teks Editorial

   Teks Editorial merupakan sebuah teks yang ada pada koran atau majalah yang merupakan ungkapan wawasan atau gagasan terhadap sesuatu yang mewakili koran atau majalah tersebut. Teks Editorial biasanya disebut tajuk rencana.

5. Teks Eksemplum

   Teks Eksemplum adalah jenis teks rekaan yang berisi insiden yang menurut partisipannya tidak perlu terjadi.

6. Teks Eksplanasi

   Teks Eksplanasi merupakan sebuah teks yang menjelaskan hubungan logis dari beberapa peristiwa.

7. Teks Eksposisi

   Teks Eksposisi merupakan sebuah teks yang berfungsi untuk mengungkapkan gagasan atau megusulkan sesuatu berdasarkan argumentasi yang kuat.

8. Teks Naratif

   Naratif merupakan sebuah teks rekaan yang berisi komplikasi yang menimbulkan masalah yang memerlukan waktu untuk melakukan evaluasi agar dapat memecahkan masalah tersebut.

9. Teks Negosiasi

   Sesuai dengan namanya Teks Negosiasi merupakan sebuah teks yang berisi tentang proses tawar-menawar dng jalan berunding guna mencapai kesepakatan bersama antara satu pihak (kelompok atau organisasi) dan pihak (kelompok atau organisasi) yg lain.

10. Teks Recount

    Teks Recount merupakan sebuah teks yang berisi pengungkapan pengalaman atau peristiwa yang dilakukan pada masa lampau

11. Teks Prosedure (teks prosedural)

    Teks prosedure merupakan sebuah teks yang menerangkan langkah-langkah untuk membuat sesuatu atau mencapai suatu tujuan. Teks prosedur juga dapat berupa sebuah protokol.

2.2.3 Algoritma IDEA

Algoritma *International Data Encryption Algorithm* (IDEA) merupakan algoritma simetris yang muncul pertama kali pada tahun 1990 yang dikembangkan oleh ilmuan Xueijia Lai dan James L Massey. algoritma simetris yang beroperasi pada sebuah blok pesan terbuka dengan lebar 64 bit dan panjang kunci berukuran 128 bit. Algoritma IDEA menggunakan algoritma yang sama untuk proses enkripsi dan dekripsi. Dan pesan rahasia yang dihasilkan oleh

algoritma ini berupa blok pesan rahasia dengan lebar atau ukuran 64 bit. Algoritma IDEA menggunakan operasi campuran dari tiga operasi aljabar yang berbeda, yaitu: Operasi XOR, Operasi penjumlahan modulo 216 dan Operasi perkalian modulo 216 +1. Semua operasi ini dilakukan pada subblok 16 bit. Algoritma IDEA melakukan iterasi sebanyak 8 iterasi dan terdapat transformasi keluaran setelah melakukan 8 iterasi. (Tri Andriyanto, 2008)

Algoritma utama dari sistem kriptografi IDEA adalah sebagai berikut:

1. Proses enkripsi : ek(M)= C
2. Proses dekripsi : dk(C)= M

Dimana:

E = adalah fungsi enkripsi

D = adalah fungsi dekripsi

M = adalah pesan terbuka

C = adalah pesan rahasia

K = adalah kunci enkripsi atau dekripsi

2.2.4 Proses Enkripsi IDEA

Pada proses enkripsi, algoritma IDEA ini terdapat tiga operasi yang berbeda untuk pasangan sub-blok 16-bit yang digunakan, sebagai berikut:

1. XOR dua sub-blok 16-bit bir per bit
2. Penjumlahan integer modulo (216 + 1) dua sub-blok 16-bit , dimana kedua sub-blok itu dianggap sebagai representasi biner dari integer biasa.

Perkalian modulo (216 + 1) dua sub-blok 16-bit, dimana kedua sub-blok 16-bit itu dianggap sebagai representasi biner dari integer biasa kecuali sub-blok nol dianggap mewakili integer 216

Blok pesan terbuka dengan lebar 64-bit , X, dibagi menjadi 4 sub-blok 16-bit, X1, X2, X3, X4, sehingga X = (X1, X2, X3, X4). Keempat sub-blok 16-bit itu ditransformasikan menjadi sub-blok 16-bit, Y2, Y2, Y3, Y4, sebagai pesan rahasia 64-bit Y = (Y1, Y2, Y3, Y4) yang berada dibawah kendali 52 sub-blok kunci 16-bit yang dibentuk dari dari blok kunci 128 bit.

Keempat sub-blok 16-bit, X1, X2, X3, X4, digunakan sebagai masukn untuk putaran pertama dari algoritma IDEA. Dalam setiap putaran dilakukan operasi XOR, penjumlahan, perkalian antara dua sub-blok 16-bit dan diikuti pertukaran antara sub-blok 16-bit putaran kedua dan ketiga. Keluaran putaran sebelumnya menjadi masukan putaran berikutnya. Setelah putaran kedelapan dilakukan transformasi keluara yang dikendalikan oleh 4 sub-blok unci 16-bit

Pada setiap putaran dilakukan operasi-operasi sebagai berikut :

1. Perkalian X1 dengan sub-kunci pertama mod (216 + 1)
2. Penjumlahan X2 dengan sub-kunci kedua mod 216
3. Pejumlahan X3 dengan sub kunci ketiga mod 216
4. Perkalian X4 dengan sub kunci keempat
5. Operasi XOR hasil langkah 1) dan 3)
6. Operasi XOR hasil angkah 2) dan 4)
7. Perkalian hasil langkah 5) dengan sub-kunci kelima mod (216 + 1)
8. Penjumlahan hasil langkah 6) dengan langkah 7) mod 216
9. Perkalian hasil langkah 8) dengan sub-kunci keenam mod (216 + 1)
10. Penjumlahan hasil langah 7) dengan 9) mod 216
11. Operasi XOR hasil langkah 1) dan 9)
12. Operasi XOR hasil langkah 3) dan 9)
13. Operasi XOR hasil langkah 2) dan 10)
14. Operasi XOR hasil langkah 4) dan 10)

Keluaran setiap putaran adalah 4 sub-blok yang dihasilkan pada langkah 11, 12, 13, dan 14 dan menjadi masukan putaran berikutnya.

Setelah putaran kedelapan terdapat transformasi keluaran, yaitu :

1. Perkalian X1 dengan sub-kunci pertama mod (216 + 1)
2. Penjumlahan X2 dengan sub-kuci ketiga mod 216
3. Penjumlahan X3 dengan sub-kunci kedua mod 216
4. Perkalian X4 dengan sub-kunci keempat mod (216 + 1)

Terahir, keempat sub-blok 16-bit 16-bit yang merupakan hasil operasi 1, 2, 3, dan 4 ii digabung kembali menjadi blok pesan rahasia 64-bit.

2.2.5 Proses Dekripsi IDEA

Proses dekripsi menggunakan algoritma yang sama dengan proses enkripsi tetapi 52 buah sub-blok kunci yang digunakan masing-masing merup akan hasil turunan 52 buah sub-blok kunci enkripsi.

Sub-blok Kunci Enkripsi

Putaran ke-1	K_1 K_2 K_3 K_4 K_5 K_6
Putaran ke-2	K_7 K_8 K_9 K_{10} K_{11} K_{12}
Putaran ke-3	K_{13} K_{14} K_{15} K_{16} K_{17} K_{18}
Putaran ke-4	K_{19} K_{20} K_{21} K_{22} K_{23} K_{24}
Putaran ke-5	K_{25} K_{26} K_{27} K_{28} K_{29} K_{30}
Putaran ke-6	K_{31} K_{32} K_{33} K_{34} K_{35} K_{36}
Putaran ke-7	K_{37} K_{38} K_{39} K_{40} K_{41} K_{42}
Putaran ke-8	K_{43} K_{44} K_{45} K_{46} K_{47} K_{48}
Transformasi Output	K_{49} K_{50} K_{51} K_{52}

Sub-blok Kunci Dekripsi

Putaran ke-1	$(K_{49})^{-1}$ –K_{50} –K_{51} $(K_{52})^{-1}$ K_{47} K_{48}
Putaran ke-2	$(K_{43})^{-1}$ –K_{45} –K_{44} $(K_{46})^{-1}$ K_{41} K_{42}
Putaran ke-3	$(K_{37})^{-1}$ –K_{39} –K_{38} $(K_{40})^{-1}$ K_{35} K_{36}
Putaran ke-4	$(K_{31})^{-1}$ –K_{33} –K_{32} $(K_{34})^{-1}$ K_{29} K_{30}
Putaran ke-5	$(K_{25})^{-1}$ –K_{27} –K_{26} $(K_{28})^{-1}$ K_{23} K_{24}
Putaran ke-6	$(K_{19})^{-1}$ –K_{21} –K_{20} $(K_{22})^{-1}$ K_{17} K_{18}
Putaran ke-7	$(K_{13})^{-1}$ –K_{15} –K_{14} $(K_{16})^{-1}$ K_{11} K_{12}
Putaran ke-8	$(K_{7})^{-1}$ –K_{9} –K_{8} $(K_{10})^{-1}$ K_{5} K_{6}
Transformasi Output	$(K_{1})^{-1}$ –K_{2} –K_{3} $(K_{4})^{-1}$

Keterangan :

a. K-1 merupakan invers perkalian modulo 216+1 dari K, dimana K K-1 = 1
b. K merupakan invers penjumlahan modulo 216 dri K, dimana K K-1 = 0

Pembentukan sub-kunci

Sebanyak 52 sub-blok kunci 16-bit untuk proses enkripsi diperolah dari sebuah kunci 128-bit pilihan pemakai. Blok kunci 128-bit dipartisi menjadi 8 sub-blok kunci 16-bit yang lansung dipakai sebagai 8 sub-blok kunci pertama. Kemudian blok kunci 128-bit dirotasi dari kiri 25 poisi untk dipartisi lagi menjad 8 sub-blok kunci 16-bit berikutnya. Proses rotasi dan pertisi itu diulangi lagi smpai diperoleh 52 sub-blok kunci 16-bit

2.3 Steganografi

Kata steganografi berasal dari bahasa Yunani, yaitu dari kata Stegos (covered/tersembunyi) dan Graptos (writing/tulisan). Steganografi di dunia modern biasanya mengacu pada informasi atau suatu arsip yang telah disembunyikan ke dalam suatu arsip citra digital, audio, atau video. Teknik Steganografi ini telah banyak digunakan dalam strategi peperangan dan pengiriman sandi rahasia sejak jaman dahulu kala. Dalam perang Dunia II, teknik steganografi umum digunakan oleh tentara Jerman dalam mengirimkan pesan rahasia dari atau menuju Jerman. Steganografi adalah ilmu atau seni menyembunyikan (embedded) informasi dengan cara menyisipkan pesan rahasian di dalam sesuatu media tertentu.

Menurut (Munir, 2006) terdapat beberapa istilah yang berkaitan dengan steganografi yaitu :

1. *Carrier file* : *file* yang berisi pesan rahasia tersebut.
2. *Steganalysis* : proses untuk mendeteksi keberadaan pesan rahasia dalam suatu *file.*
3. *Stego-medium* : media yang digunakan untuk membawa pesan rahasia.
4. *Redundant bits* : sebagian informasi yang terdapat di dalam *file* yang jika dihilangkan tidak akan menimbulkan kerusakan yang signifikan (setidaknya indera manusia).
5. *Payload* : informasi yang akan disembunyikan

Steganografi mempunyai tiga aspek berbeda di dalam sistem penyembunyian informasi bertentangan dengan satu sama lain yaitu (Wijaya & Prayudi, 2004):

1. Kapasitas (capacity)

   Kapasitas mengarah kepada jumlah informasi yang dapat disembunyikan dalam medium cover. Keamanan merupakan ketidakmampuan pengamat dalam mendeteksi pesan yang disembunyikan dan ketahanan adalah jumlah modifikasi medium stego yang dapat bertahan sebelum musuh merusak pesan rahasia yang disembunyikan dalam steganografi.

2. Keamanan (security)

   Keamanan dalam sistem steganografi klasik menggambarkan rahasia sistem encodding-nya. Teori informasi sangat mungkin untuk lebih spesifik daripada apa yang dimaksudkan dengan suatu sistem yang benar-benar aman.

3. Ketahanan (robustness)

   Ketahanan mengarah kepada data citra penampang (seperti dirubahnya kontras, penajaman, rotasi, perbesaran gambar, pemotongan dan sebagainya). Jika pada citra dijalankan operasi pengolahan citra, maka data yang tersembunyi tidak mengalami kerusakan.

Didalam steganografi ada beberapa teknik steganografi yang dipakai ada tujuh teknik steganografi diantaranya:

1. Injection

   Adalah sebuah teknik penanaman pesan rahasian dengan langsung ke dalam sebuah media. Salah satu masalah dari teknik ini yaitu ukuran media yang bisa diinjeksi menjadi lebih besar dari ukuran normalnya, menjadikan mudah terdeteksi. Teknik ini biasa disebut dengan embedding.

2. Subtitusi

   Adalah data normal yang diganti dengan data rahasia. Seringkali hasil dari teknik ini tidak begitu merubah ukuran data asli, namun tergantung dari file media yan data yang ingin disembunyikan. Teknik substitusi dapat menurunkan kualitas media yang disisipinya.

3. Transformasi Domain

   Adalah teknik yang sangat efektif. Pada dasarnya, transformasi domain membuat data tersembunyi dalam transforspace.

4. SpreadSpectrum

   Adalah teknik pentransmisi memakai pseudo-noise code, yang independen data informasi sebagai modulator berupa gelombang dalam penyebaran energi sinyal dalam suatu jalur komunikasi (bandwith) yang lebih besar

daripada sinyal jalur komunikasi informasi. Oleh penerima, sinyal disatukan kembali memkai replika pseudo-noise code yang disinkronkan.

5. Statistical Method

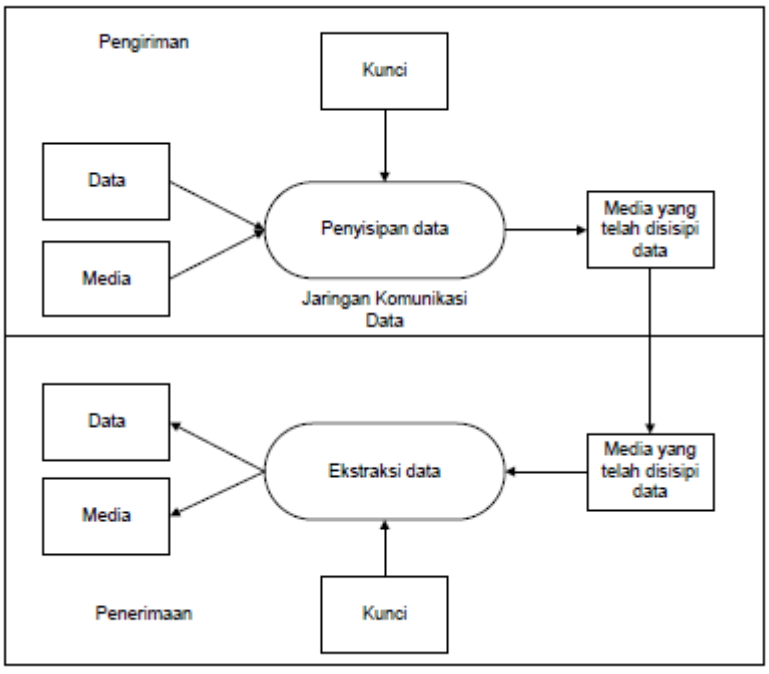
   Adalah teknik yang disebut juga skema steganographic 1 bit. Skema itu menanamkan satu bit informasi di media tumpangan dan merubah statistik meskipun hanya 1 bity. Perubahan statistik ditampilkan dengan indikasi 1 dan apabila tidak ada perubahan, akan diketahui indikasi 0. Sisten ini bekerja menurut kemampuan penerima dalam membedakan antara informasi yang dimodifikasi dengan yang belum.

6. Distortion

   Metode ini membuat perubahan terhadap benda yang ditumpangi oleh data rahasia.

7. Cover Generation

   Adalah metode yang unik dibanding dengan metode lainnya sebab cover object dipilih untuk membuat pesan tersembunyi.



Gambar 2.1 Ilustrasi Dasar Konsep Steganografi (Arini, 2012)

Steganografi adalah seni dan ilmu menulis pesan tersembunyi dengan suatu cara sehingga selain si penerima yang dimaksud tak ada satupun orang yang mengetahui atau menyadari bahwa suatu pesam tersimpan. Sebaliknya, kriptografi adalah menyamarkan arti suatu pesan tanpa menyembunyikan keberadaanya dan membuat siapapun menyadari bahwa ada sesuatu yang mencurigakan dari pesan tersebut.

2.3.1 Metode Steganografi

Untuk media gambar metode steganografi memiliki beberapa meode yang semakin hari semakin berkembang karena adanya kelemahan disetiap metode tersebut. Contoh beberapa metode yang sudah diketahui :

1. *Least Significant Bits* (LSB)

   Metode yang digunakan untuk menyembunyikan pesan pada media digital tersebut berbeda-beda. Contohnya, pada berkas image pesan dapat disembunyikan dengan menggunakan cara menyisipkannya pada bit rendah atau bit yang paling kanan (LSB) pada data pixel yang menyusun file tersebut. Pada berkas bitmap 24 bit, setiap pixel (titik) pada gambar tersebut terdiri dari susunan tiga warna merah, hijau dan biru (RGB) yang masing-masing disusun oleh bilangan 8 bit (byte) dari 0 sampai 255 atau dengan format biner 00000000 sampai 11111111. Dengan demikian, pada setiap pixel berkas bitmap 24 bit kita dapat menyisipkan 3 bit data. Kekurangan dari LSB Insertion: Dapat diambil kesimpulan dari contoh 8 bit pixel, menggunakan LSB Insertion dapat secara drastis mengubah unsur pokok warna dari pixel. Ini dapat menunjukkan perbedaan yang nyata dari cover image menjadi stego image, sehingga tanda tersebut menunjukkan keadaan dari steganografi. Variasi warna kurang jelas dengan 24 bit image, bagaimanapun file tersebut sangatlah besar. Antara 8 bit dan 24 bit image mudah diserang dalam pemrosesan image, seperti cropping (kegagalan) dan compression (pemampatan). Keuntungan dari

LSB Insertion : Keuntungan yang paling besar dari algoritma LSB ini adalah cepat dan mudah. Dan juga algoritma tersebut memiliki software steganografi yang mendukung dengan bekerja di antara unsur pokok warna LSB melalui manipulasi pallete (lukisan).

2. Metode *End Of Fle* (EOF)

Teknik EOF atau End Of File merupakan salah satu teknik yang digunakan dalam steganografi. Teknik ini menggunakan cara dengan menyisipkan data pada akhir file. Teknik ini dapat digunakan untuk menyisipkan data yang ukurannya sesuai dengan kebutuhan. Ukuran file yang telah disisipkan data sama dengan ukuran file sebelum disisipkan data ditambah dengan ukuran data yang disisipkan ke dalam file tersebut.(Utami, 2007). Untuk teknik ini dapat menambahkan data atau file yang akan disembunyikan lebih dari ukuran file image. Data yang disembunyikan tersebut akan disisipkan pada akhir file sehingga file image akan terlihat sedikit berbeda dengan aslinya. Ada penanda khusus yang terlihat dari file image di paling bawah seperti garis-garis.

2.3.2 Proses penyisipan pesan metode EOF

Adapun langkah-langkah *encoding* menggunakan metode *End of File* adalah sebagai berikut (Sembiring, 2013):

1. Proses encoding dimulai dengan pesan yang akan disisipkan. Pesan diubah kedalam bentuk biner dengan representasi 1 atau 0.
2. Kemudian disisipkan angka 1 didepan rangkaian biner tersebut.langkah selanjutnya rangkaian biner tersebut dikonversikan menjadi bilangan decimal dan menghasilkan sebuah bilangan yang dinamakan dengan *m*
3. Menghitung jumlah warna yang terdapat pada berkas RGB yang menjadi objek steganografi.dan akan menghasilkan sebuah bilangan.Bilangan tesebut dinamakan dengan n, maka apabila m>n!-1 maka pesan yang akan disisipkan berukuran terlalu besar sehingga proses penyisipan tidak dapat dilakukan.

4. Warna dalam palet warna diurutkan sesuai dengan urutan yang natural. Setiap warna dengan *format* RGB dikonversikan kedalam bilangan integer dengan aturan (Merah *65536 + Hijau * 256 + Biru). Kemudian diurutkan berdasarkan besar bilangan integer yang mewakili warna tersebut.
5. Setelah itu lakukan proses iterasi terhadap variable *i* dengan nilai *i* adalah dari 1 sampai n. Setiap warna dengan urutan *n – i* dipindahkan ke posisi baru yaitu *m mod i*, kemudian m dibagi dengan *i*.
6. Kemudian palet warna yang baru hasil iterasi pada langkah ke – 4 dimasukkan kedalm palet warna berkas RGB. Apabila ada tempat yang diisi oleh dua buah warna, maka warna sebelumnya yang menempati tempat tesebut akan digeser satu tempat ke samping.
7. Apabila ternyata besar dari palet warna yang baru lebih kecil dari 256 maka palet warna akan diisi dengan warna terakhir dari palet warna sebelumnya.
8. Kemudian berkas RGB akan dikompresi ulang dengan palet warna yang baru, untuk menghasilkan berkas yang baru dengan ukuran dan gambar yang sama, namun telah disisipi pesan.

Adapun langkah- langkah proses *decoding* atau mengekstrak pesan dari citra RGB yang telah disisipi pesan dengan metode *End Of File* adalah sebagai berikut:

1. Masukkan nomor sesuai dengan posisi setiap warna pada palet warna citra RGB yang telah disisipkan pesan
2. Warna diurutkan berdasarkan konversi RGB ke nilai integer dengan rumus:(Merah * 65536 + Hijau * 256 + Biru).
3. *m* diberi nilai 0
4. Iterasi variabel *i* dari *i+1* sampai *n-1*. m=m*(n-1) + posisi warna ke *i* iterasi variabel *j* dari *i* +1 sampai n-1 jika posisi warna ke *j* > nilai posisi warna ke-i, maka posisi warna ke *i* dikurangkan 1
5. Setelah nilai *m* diperoleh, maka nilai *m* dikonversikan kebilangan binary untuk memperoleh pesan asli kembali.

2.3.3 Pengujian kelayakan steganografi

Sudah banyak penelitian terhadap keamanan data khususnya steganografi, yang mana kualitas dari sebuah objek file input dan ouput akan menjadi aplikasi steganografi. Media penampung adalah Data dapat berupa file text ataupun file dokumen. Semua informasi atau data pada komputer disimpan serta dimanipulasi dalam format biner yaitu 0 dan 1 dan sering disebut dengan bit (binary digit), BMP adalah representasi dari citra grafis yang terdiri dari susunan titik yang tersimpan di memori komputer yang tidak terkompresi. Sebenarnya banyak jenis citra lain yang bisa dijadikan media untuk steganografi seperti PNG, JPEG dan lain-lain.. oleh karena itu disini saya juga akan menggunakan format gambar juga sebagai media kompresi.

MSE (Mean Squared Error) merupakan nilai error kuadrat rata - rata antara citra asli dengan dengan citra manipulasi dalam beberapa contoh steganografi mse ialah nilai error kuadrat rata - rata antara citra asli dan citra sisipan.

PSNR (Peak Signal to Noise Ratio) merupakan perbandingan antara nilai maksimum dari sinyal yang diukur dengan besarnya derau yang berpengaruh pada sinyal tersebut. PSNR biasanya digunakan untuk perbandingan kualitas antara citra original dengan citra yang sudah disisipkan pesan.(Ghazali Moenandar Male, Wirawan, 2012)

Rumus MSE dan PSNR

$$\text{MSE} = \frac{1}{M.N} \sum_{x=1}^{M} \sum_{y=1}^{N} (Sxy - Cxy)^2 \quad (2.1)$$

$$\text{PSNR} = 10\ log_{10}\ (\frac{Cmax^2}{MSE}) \quad (2.2)$$

Keterangan dimana:

1. C_{max} merupakan nilai pixel terbesar pada keseluruhan citra
2. x dan y merupakan koordinat pada suatu titik pada citra
3. M dan N adalah dimensi dari citra
4. S adalah citra tersisipi
5. C adalah citra asli

2.4 Histogram

Histogram citra merupakan grafik yang menggambarkan penyebaran nilai-nilai intensitas pixel dari suatu citra atau bagian tertentu didalam citra. Dari sebuah histogram dapat diketahui frekuensi kemunculan relative dari intensitas pada citra tersebut (Munir, 2004)

Dalam statistic histrogram berupa grafis untuk menampilkan distribusi data berupa visual atau beberapa sering suatu nilai yang berbeda terjadi dalam sekumpulan data.

Dengan demikian histogram chiper-image seharusnya datar atau secara statistik distribusi relative. Sehingga adanya sebuah indikasi bahwa algoritma enkripsi citra memiliki tingkat keamanan yang bagus (Munir, 2012)

2.5 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, middleuare dan aplikasi. Android menyediakan platfurm terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. (Safaat, 2012) Android ini memiliki beberapa kelebihan antara lain:

1. Open Source

   Sistem operasi Android ini memang merupakan sistem operasi yang bersifat Open Source yang dapat dikembangakan oleh siapapun. Semua aplikasi yang disediakan di Google Play merupakan pengembangan dari semua orang (programmer) di dunia.
2. Multi Tasking

   Multi Tasking artinya mampu mengoperasikan lebih dari satu aplikasi sekaligus. Seperi menjalankan aplikasi social media dan pada saat itu juga menjalankan aplikasi pemutar musik.
3. Widget Widget

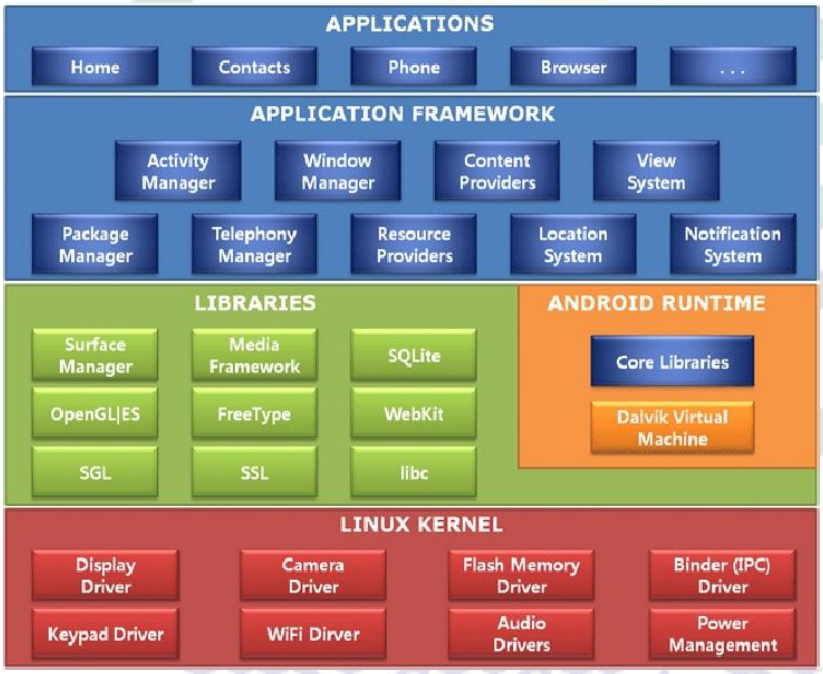
   merupakan salah satu aplikasi yang dapat membantu pengguna dalam menjalankan aplikasi dengan jalan pintas.

4. Syncronisation Dengan syncronisation pengguna dapat mengintregasikan e-mail, akun social media, gmail dan lainnya dengan sistem operasi Android. Sehingga pengguna akan mengetahui informasi terbaru dan pesan yang masuk pada e-mail atau akun social media dengan cepat.

Sedangkan kekurangan dari Android antara lain:

1. Haus Data Internet Sistem operasi Android memang menjadi OS yang haus akan data internet. Beberapa aplikasi yang disediakan hanya dapat diakses dengan menggunakan data internet.
2. Boros Baterai Konsumsi daya baterai yang digunakan Android memang terbilang boros, terlebih lagi bila pengguna menuikmari signal 3G.

Arsitektur android terdiri dari bebrapa layer yang membentuk sistem android dapat dilihat dari gambar berikut:



Gambar 2.2 Arsitektur Android
(sumber: https://sites.google.com/a/student.unsika.ac.id/bongkar-os-linux/struktur-sistem-operasi/struktur-sistem-operasi-android)

secara garis besar Arsitektur Android dapat dijelaskan dari gambar diatas sebagai berikut (Safaat, 2012):

1. Applications dan Widgets ini adalah layer di mana kita berhubungan dengan aplikasi saja, di mana biasanya kita download aplikasi kemudian kita lakukan instlasi dan jalankan aplikasi tersebut. Di layer terdapat aplikasi inti atermasuk klien email, program SMS, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi ditulis menggunakan bahasa pemrograman Java.
2. Applications Frameworks Android adalah "Open Development Platform" yaitu Android menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang bagus dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi resources, menjalankan seruice background, mengatur alarm, dan menambahkan status notifi.cations, dan sebagainya. pengembang memiliki akses penuh menuju API framework seperti yang dilakukan oleh aplikasi yang kategori inti. Arsitektur aplikasi dirancang supaya kita dengan mudah dapat menggunakan kembali komponen yang sudah digunakan (reuse).
3. Libraries ini adalah layer di mana fitur-fitur Android berada, biasanya para pembuat aplikasi mengakses libraries untuk menjalankan aplikasinya.
4. Android RunTime Layer yang membuat aplikasi Android dapat dijalankan di mana dalam prosesnya menggunakan Implementasi Linux. Daluik Virtual Machine (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi Android.
5. Linux Kernel adalah layer di mana inti dari operating sistem dari Android itu berada. Berisi file--1ile system yang mengatur sistem processing, memorA, resource, driuers, dan sistem-sistem operasi android lainnya.

2.6 Penelitian yang terkait

Penelitian yang telah dilakukan sebelumnya oleh (Tri Andriyanto, 2008) dengan penelitian perbandingan antara algoritma idea dan algoritma blowfish pada penelitian ini menjelaskan tentang membandingkan kinerja algoritma IDEA dan Blowfish dalam hal kecepatan proses dan penggunaan memori pada saat proses enkripsi dan dekripsi suatu file. Terlihat dari penelitian tersebut bahwa algoritma IDEA lebih cepat dari algoritma Blowfish dan pemakaian memori kedua algoritma relatif sama. Tetapi penelitian tersebut tidak melihatkan kualitas data yang diperoleh.

Penelitian yang telah dilakukan sebelumnya oleh (Hendrawan, 2015) dengan penelitian metode pengamanan untuk metode pengamanan untuk file bertipe dokumen menggunakan kombinasi algoritma idea dan lsb . Pada penelitian ini membahas tentang penyisipan file ke dalam gambar dengan enkripsi file dengan idea dan disembunyikan ke gambar dengan metode lsb , tetapi jumlah data yang disisipkan terbatas .

Penelitian yang telah dilakukan sebelumnya oleh (Muslih & rachmawanto, 2016) dengan penelitian pengamanan file multimedia dengan metode steganografi end of file untuk menjaga kerahasian pesan, pada penelitian ini membahas tentang menyembunyikan file pada file induk tanpa ada perubahan yang signifikan terhadap file induk. Selain itu, file juga dapat diambil kembali untuk dilihat isi pesan atau informasi dengan jelas tanpa ada noise sedikitpun. Tetapi akan lebih baik digabungkan dengan metode kriptografi agar lebih aman.

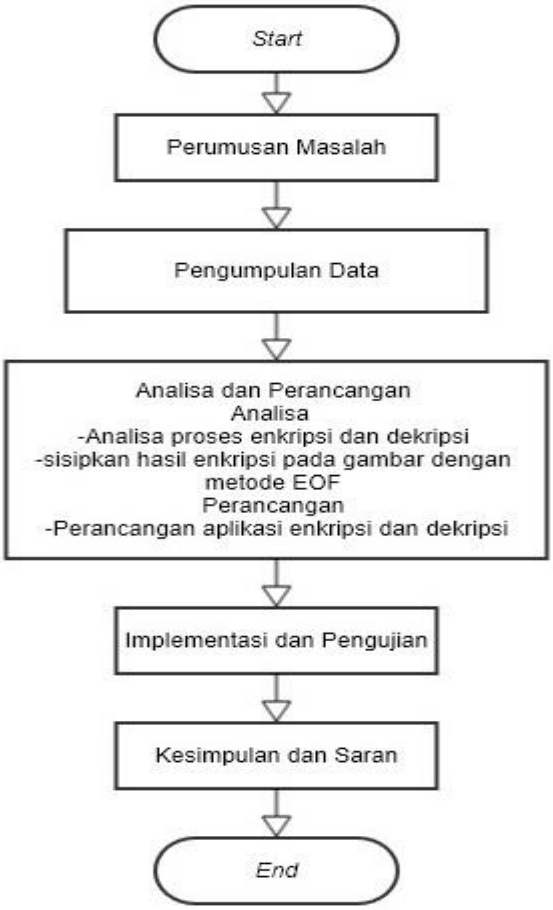
Penelitian yang telah dilakukan sebelumnya oleh (Krisnawati, 2008) dengan penelitian metode least significant bit dan end of file untuk menyisipkan teks kedalam citr, pada penelitian ini menjelaskan tentang membandingkan tentang kelebihan dan kekurangan metode LSB dan EOF dalam penyisipkan pesan teks, Metode LSB pada saat penyisipan pesan jumlah karakter pesan sangat terbatas harus menyesuaikan besar citra dan besar pesan dikirim tetapi kelebihan ukuran citra tidak berubah . Sedangkan Metode EOF akan meletakkan pesan di

akhir citra sehingga ukuran file akan bertambah besar, oleh karena itu pesan teks yang disisipkan tidak terbatas jumlahnya.

Penelitian yang telah dilakukan sebelumnya oleh (Ukkas, Andrea, Baretto, & Anggen, n.d.) pada penelitian ini menjelaskan tentang Teknik Pengamanan Data Dengan Menggunakan Steganografi Metode End of File dan Kriptografi Vernam Cipher ini dapat bekerja menyisipkan sebuah atau beberapa plain file didalam sebuah file spoof dengan format bitmap (.bmp) dengan mengimplementasikan steganografi metode end of file, sehingga plain file tersebut tersembunyi didalam file spoof. tetapi pengujiannya hanya format gambar bitmap (.bmp) yang digunakan untuk enkripsi dan membuat kurangnya keamanannya.

BAB III
METODOLOGI PENELITIAN

Metodologi penelitian ialah acuan atau tahapan-tahapan yang akan dilakukan selama penelitian agar penelitian dapat berjalan sesuai prosedur dan mencapai hasil yang maksimal. Metodologi penelitian pada tugas akhir ini dapat dilihat pada Gambar 3.1 berikut:



Gambar 3.1 Metodologi Penelitian

3.1 Perumusan Masalah

Pada tahapan perumusan masalah, dilakukan perumusan terhadap permasalahan yang akan diteliti tentang mengamankan data teks, ketepatan huruf yang akan di enkripsi dan dekripsi, cepat atau lambat waktu enkripsi data teks, mehihat perbandingan histogram, melihat kualitas gambar cover dan gambar yang telah disisipkan dan melihat perubahan ukuran data yang dapat dienkripsi dan didekripsi oleh algoritma *International Data Encryption Algorithm* (IDEA) dan teknik Steganografi dengan metode *End Of File* (EOF) jika digabungkan.

3.2 Pengumpulan Data

Tahapan pengumpulan data ini yang dilakukkan adalah pengumpulan data dan informasi dari penelitian terkait berupa referensi berupa jurnal-jurnal, buku yang berupa *text book* maupun *e-book*, artikel dari internet dan referensi-referensi terkait lainnya dengan penelitian yaitu penyembunyian data teks pada gambar menggunakan algoritma kriptografi *International Data Encryption Algorithm* (IDEA) dan teknik Steganografi dengan metode *End Of File* (EOF)

3.3 Analisa dan Perancangan

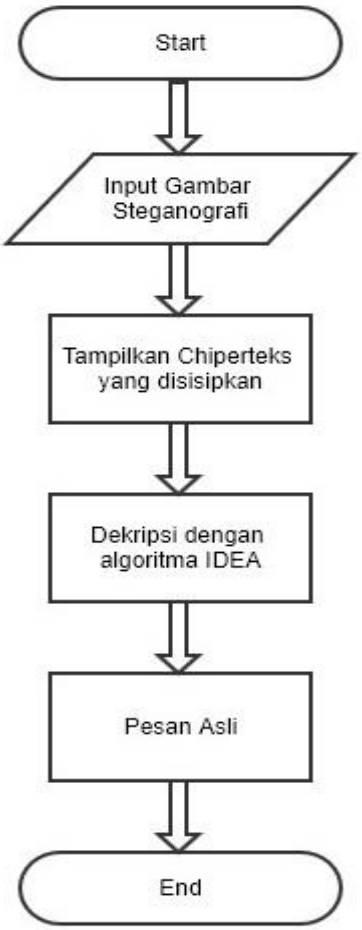
3.3.1 Analisa

Selesai mengidentifikasi masalah dan setelah melakukan pengumpulan data, maka selanjutnya ketahapan analisa. Analisa merupakan sebuah proses untuk memecahkan sesuatu yang saling berkaitan yang di dalamnya mempelajari serta mengevaluasi suatu yang terjadi yang bertujuan untuk mengetahui gambaran jelas mengenai penelitian yang dilakukan, analisa terdapat beberapa tahapan dan flowchart yang dilakukan pada penelitian ini yaitu:

Gambar 3.2 Flowchart Sistem Enkripsi Pesan

Penjelasan Flowchart Sistem Enkripsi

1. Masukkan Planteks atau teks yang akan dienkripsi dan masukkan kunci public dari IDEA
2. Enkripsi data teks dengan algoritma Idea
3. Setelah chiperteks didapatkan kemudian chiperteks tersebut disisipkan kedalam gambar dengan algoritma EOF
4. Melihat perbedaan pesan disispkan dengan histogram
5. Kemudian menghitung kualitas citra dengan MSE dan PSNR
6. Melihat kecepatan waktu enkripsi



Gambar 3.3 Flowchart Sistem Dekripsi pesan

Penjelasan Flowchart Sistem Dekripsi Pesan

1. Tampilkan chiperteks pada gambar dengan EOF
2. Kemudian masukan kunci public kemudian dekripsi chiperteks tersebut dengan algoritma IDEA
3. Pesan asli akan tampil

3.3.2 Perancangan

Pada tahapan ini peracangan aplikasi dirancang berdasarkan hasil analisa yang telah dilakukan sebelumnya untuk mengimplementasi enkripsi dan dekripsi algoritma IDEA dan metode end of file berkerja pada aplikasi terdapat 3 tahapan yang dilakukan yakni:

1. Perancangan basis data

   Pada tahapan ini merupakan tahap perancangan tabel dan atribut yang dibutuhkan oleh sistem.

2. Coding

   Pada tahapan ini dilakukan tahapan pengcodingan aplikasi dengan menggunakan aplikasi pendukung.

3. Perancangan *interface* (antar muka)

   Pada tahapan Aplikasi Perancangan *interface* aplikasi bertujuan untuk memperlihatkan bagaimana tampilan atau konsep awal desain antarmuka dan aplikasi yang akan dibangun.

3.4 Implementasi dan Pengujian

Pada tahapan implementasi ini, dilakukan tahapan yang telah dikembangkan berdasarkan hasil analisa dan perancangan yang telah dilakukan sebelumnya, aplikasi yang telah dibangun akan dicoba langsung di *smartphone* untuk mengetahui tingkat keberhasilannya dan kelemahan aplikasi tersebut sehingga akan diketahui apakah aplikasi akan berjalan dengan tujuan yang diinginkan.

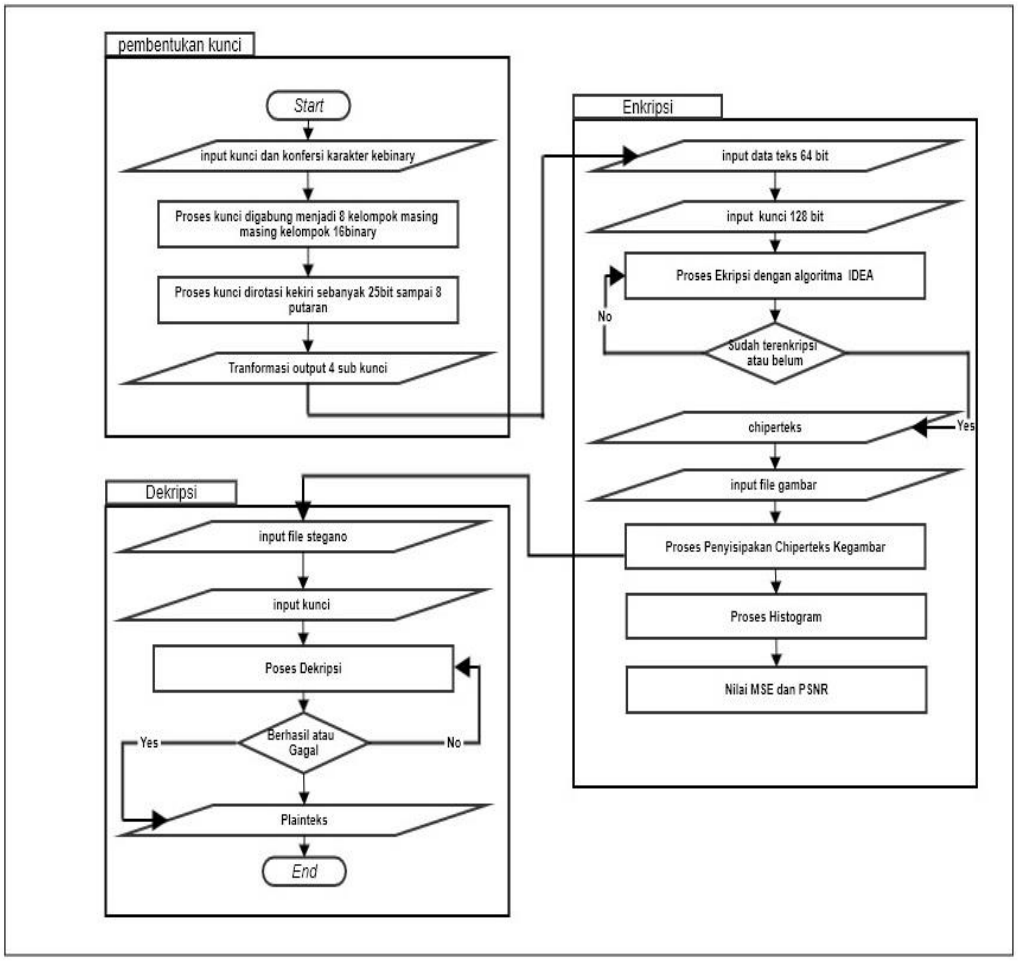
Pada tahapan pengujian dilakukkan pengujian dengan kriptografi dan steganografi melihat keamanan atau ketahanan metode, pengujian terhadap ketepatan huruf yang akan di enkripsi dan dekripsi, cepat atau lambat waktu enkripsi data teks, dan melihat perubahan ukuran data yang dapat dienkripsi dan didekripsi, melihat perubahan pesan dihistogram dan melihat kualitas citra dengan MSE dan PSNR.

3.5 Kesimpulan dan Saran

Pada tahapan ini tahap penentuan kesimpulan terhadap hasil pengujian yang telah dilakukan. Selain itu juga diberikan saran untuk penyempurnaan serta pengembangan hasil penelitian ini untuk selanjutnya.

BAB IV
ANALISA DAN PERANCANGAN

Pada bab ini membahas mengenai analisis dan perancangan sistem dari algoritma *International Data Encryption Algorithm* (IDEA) dan Steganografi *End Of File* (EOF) dan ada beberapa tahapan-tahapan skema pada penelitian ini dapat dilihat gambar 4.1



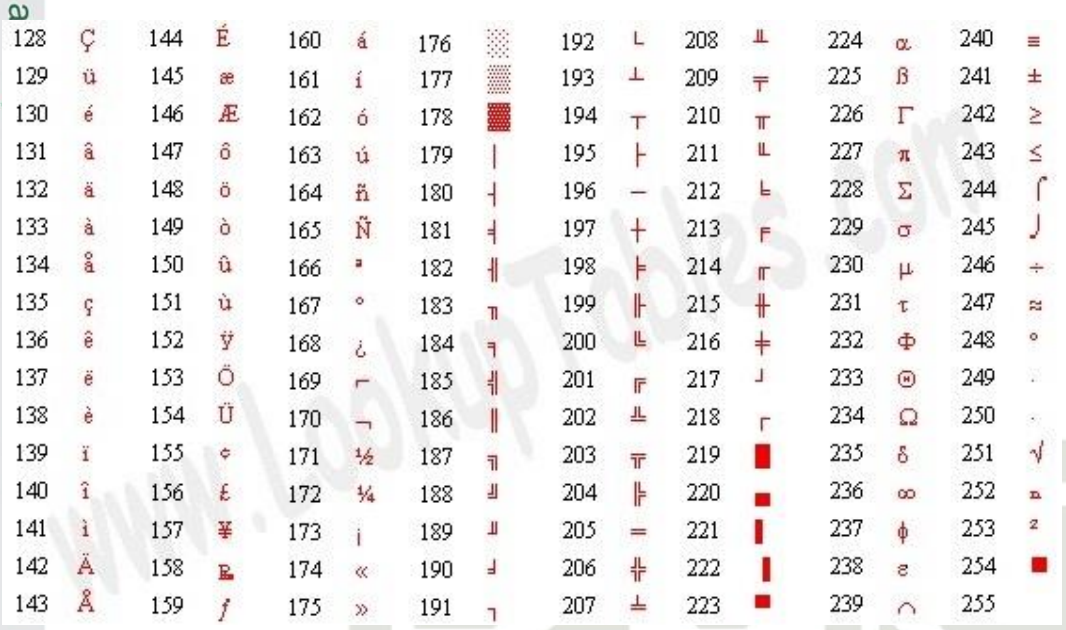
Gambar 4.1 Flowchart Analisa Metode

4.1 Analisis Metode

Pada tahapan analisi metode merupakan sebuah proses untuk memecahkan sesuatu yang saling berkaitan yang di dalamnya mempelajari serta mengevaluasi suatu yang terjadi yang bertujuan melihat awal proses kerja sistem ini dan mengapa sistem ini diperlukan. Pada saat melakukkan perhitungan manual algoritma idea dan end of file dimulai dari pembentukkan kunci , proses enkripsi, proses histogram, proses menentukan nilai MSE dan PSNR dan proses dekripsi. Sebelum melakukan proses perhitungan manual diperlukan tabel ASCII untuk sebagai melihat hasil enkripsi dan dekripsi. ASCII merupakan kode standar amerika untuk pertukaran informasi atau sebuah standar internasional dalam pengkodean huruf dan simbol seperti Unicode dan hex tetapi ASCII lebih bersifat universal dan terdiri 256 karater dapat lihat pada gambar dibawah ini:

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Gambar 4.2 Tabel ASCII (Sumber : http://www.asciitable.com/)



128	Ç	144	É	160	á	176	▒	192	└	208	╨	224	α	240	≡
129	ü	145	æ	161	í	177	▒	193	┴	209	╤	225	ß	241	±
130	é	146	Æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥
131	â	147	ô	163	ú	179	│	195	├	211	╙	227	π	243	≤
132	ä	148	ö	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠
133	à	149	ò	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡
134	å	150	û	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
135	ç	151	ù	167	º	183	╖	199	╟	215	╫	231	τ	247	≈
136	ê	152	ÿ	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°
137	ë	153	Ö	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	∙
138	è	154	Ü	170	¬	186	║	202	╩	218	┌	234	Ω	250	·
139	ï	155	¢	171	½	187	╗	203	╦	219	█	235	δ	251	√
140	î	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	n
141	ì	157	¥	173	¡	189	╜	205	═	221	▌	237	ϕ	253	²
142	Ä	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■
143	Å	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255	

Gambar 4.3 Tabel ASCII (Sumber : http://www.asciitable.com/)

4.1.1 Pembentukan kunci

Tahap pembentukan subkunci idea terdiri dari 52 subkunci 16 bit untuk proses enkripsi diperoleh dari kunci 128 bit. Pada sebuah kunci 128 bit dipisahkan menjadi 8 subkunci yang masing – masing terdiri 16 bit dan langsung digunakan pada putaran pertama yang terdiri 6 subkunci dan untuk putaran ke 2 terdapat kekurangan 4 subkunci, kemudian kunci diawal 128 bit tersebut digeser kekiri sebenyak 25 bit untuk memperoleh 8 subkunci berikutnya untuk mengisi sub kunci yang kurang .dan begitu seterusnya sampai putaran ke 8 dan memperoleh 52 subkunci yang baru.

Berikut contoh perhitungan secara manual pembentukan kunci idea contoh kunci kriptostegano435

1. konversikan karakter ke binary

Tabel 4.1 konversi karakter ke binary

Char	Decimal	Binary
k	107	01101011
r	114	01110010
i	105	01101001

p	112	01110000
t	116	01110100
o	111	01101111
s	115	01110011
t	116	01110100
e	101	01100101
g	103	01100111
a	97	01100001
n	110	01101110
o	111	01101111
3	51	00110011
4	52	00110100
5	53	00110101

2. kemudian kunci digabung semua
011010110111001001101001011010010111010001101111011100110111
010001100101011001110110000101101110011011110011001100101000
00110101

3. kemudian kunci dipecah menjadi 8 kelompok masing – masing 16 bit dan dirotasi 25 bit kekiri untuk setiap putaran , 1 putaran terdiri dari 6 sub kunci.

Tabel 4.2 Tabel pembentukan subkunci

Putaran 1	0110101101110010	0110100101110000	0111010001101111
	K1	K2	K3
	0111001101110100	0110010101100111	0110000101101110
	K4	K5	K6
Putaran 2	0110111100110011	0011010000110101	1110000011101000
	K7	K8	K9
	1101111011100110	1110100011001010	1100111011000010

	K10	K11	K12
Putaran 3	1101111011100110	0110011001101000	0110101011010110
	K13	K14	K15
	1110010011010010	1100110111010001	1001010110011101
	K16	K17	K18
Putaran 4	1000010110111001	1011110011001100	1101000011010101
	K19	K20	K21
	1010110111001001	1010010111000001	1101000110111101
	K22	K23	K24
Putaran 5	0011101100001011	011100110111001	1001100110100001
	K25	K26	K27
	1010101101011011	1001001101001011	1000001110100011
	K28	K29	K30
Putaran 6	0111101110011011	1010001100101011	1111001100110011
	K31	K32	K33
	0100001101010110	1011011100100110	1001011100000111
	K34	K35	K36
Putaran 7	0100011011110111	0011011101000110	0101011001110110
	K37	K38	K39
	0001011011100110	1010110101101110	0100110100101110
	K40	K41	K42
Putaran 8	0000111010001101	1110111001101110	1000110010101100
	K43	K44	K45
	1110110000101101	1100110111100110	0110011010000110
	K46	K47	K58
Transformasi output	0101110000011101	0001101111011100	1101110100011001
	K49	K50	K51
	0101100111011000		
	K52		

4. hanya empat kunci terakhir yang akan digunakan pada transformasi output enkripsi dan dekripsi yaitu

   K49 = 0101110000011101

   K50 = 0001101111011100

   K51 = 1101110100011001

   K52 = 0101100111011000

4.1.2 Proses Enkripsi

Enkripsi merupakan suatu metode yang digunakan untuk menyandikan plaintext menjadi ciphertext sehingga keamanan informasinya terjaga. Proses enkripsi idea memiliki 8 putaran setiap putaran memiliki 14 proses .untuk lebih lebih jelasnya lihat contoh perhitungan manual dibawah ini:

Contoh plainteks dan kunci:

Plainteks : uinsuska

Kunci : kriptostegano345

Plainteks diubah terlebih dahulu ke ascii

0111010101101001 = ui

0110111001110011 = ns

0111010101110011 = us

0110101101100001 = ka

Kemudian dimasukkan pada proses putaran 1 untuk X1,X2,X3,X4

Putaran 1

P#1 (X1 * K1) mod (2^16 + 1)

= 0111010101101001 * 0110101101110010 mod (2^16+1)

= 1111101001111100

P#2 (X2 + K2) mod (2^16)

= 0110111001110011 + 0110100101110000 mod (2^16)

= 1101011111100011

P#3 (X3 + K3) mod (2^16)

= 0111010101110011 + 0111010001101111mod (2^16)

= 1110100111100010

P#4 (X4 * K4) mod (2^16 + 1)

= 0110101101100001 * 0111001101110100 mod (2^16+1)

= 0000101010000111

P#5 P#1 XOR P#3

= 1111101001111100 XOR 1110100111100010 = 0001001110011110

P#6 P#2 XOR P#4

= 1101011111100011 XOR 0000101010000111 = 1101110101100100

P#7 (P#5 * K5) mod (2^16+1)

= 0001001110011110 * 0110010101100111 mod (2^16+1)

= 0011001011001101

P#8 (P#6 + P#7) mod (2^16)

= 0001001110011110 * 0110010101100111 mod (2^16+1)

= 0011001011001101

P#9 (P#8 * K6) mod (2^16+1)

= 0001000000110001 * 0110000101101110 mod (2^16+1)

= 0111111111100101

P#10 (P#7 + P#9) mod (2^16)

= 0011001011001101 + 0111111111100101 mod (2^16)

= 1011001010110010

P#11 P#1 XOR P#9

= 1111101001111100 XOR 0111111111100101 = 1000010110011001

P#12 P#3 XOR P#9

= 1110100111100010 XOR 0111111111100101 = 1001011000000111

P#13 P#2 XOR P#10

= 1101011111100011 XOR 1011001010110010 = 0110010101010001

P#14 P#4 XOR P#10

= 0000101010000111 XOR 1011001010110010 = 1011100000110101

Selanjutnya empat langkah terakhir P#11,P#12,P#13,P#14 yang akan digunakan pada proses kedua dan seterusnya yang akan menjadi menjadi

X1,X2,X3,X4 untuk proses putaran selanjutnya hingga mejadi 8 putaran dan ditransfomasi output untuk mendapatkan chiperteksnya.

Putaran 2

P#1 (X1 * K1) mod (2^16 + 1)
= 1000010110011001 * 0110111100110011 mod (2^16+1)
= 1011101001110100

P#2 (X2 + K2) mod (2^16)
= 1001011000000111 + 0011010000110101 mod (2^16)
= 1100101000111100

P#3 (X3 + K3) mod (2^16)
= 0110010101010001 + 1110000011101000 mod (2^16)
= 0100011000111100

P#4 (X4 * K4) mod (2^16 + 1)
= 1011100000110101 * 1101111011100110
= 1101010100111100

P#5 P#1 XOR P#3
= 1011101001110100 XOR 0100011000111100 = 1111110001001101

P#6 P#2 XOR P#4
= 1100101000111100 XOR 1101010100111100 = 0001111100000000

P#7 (P#5 * K5) mod (2^16+1)
= 1111110001001101 * 1110100011001010 mod (2^16+1)
= 1111011101010111

P#8 (P#6 + P#7) mod (2^16)
= 0001111100000000 + 1111011101010111 mod (2^16)
= 0001011001010111

P#9 (P#8 * K6) mod (2^16+1)
= 0001011001010111 * 1100111011000010 mod (2^16+1)
= 1101110111100100

P#10 (P#7 + P#9) mod (2^16)
= 1111011101010111 + 1101110111100100 mod (2^16)

= 1101010100111011

P#11 P#1 XOR P#9

= 1011101001110100 XOR 1101110111100100 = 0110011110010000

P#12 P#3 XOR P#9

= 0100011000111100 XOR 1101110111100100 = 1001101111011000

P#13 P#2 XOR P#10

= 1100101000111100 XOR 1101010100111011 = 0001111100000111

P#14 P#4 XOR P#10

= 1101010100111100 XOR 1101010100111011 = 0000000000000111

Putaran 3

P#1 (X1 * K1) mod (2^16 + 1)

= 0110011110010000 * 11011100110111110 mod (2^16 + 1)

= 0011010110000111

P#2 (X2 + K2) mod (2^16)

= 1001101111011000 + 0110011001101000 mod (2^16)

= 0000001001000101

P#3 (X3 + K3) mod (2^16)

= 0001111100000111 + 0110101011010110 mod (2^16)

= 1000100111011101

P#4 (X4 * K4) mod (2^16 + 1)

= 0000000000000111 * 1110010011010010 mod (2^16 + 1)

= 0100000110111000

P#5 P#1 XOR P#3

= 0011010110000111 XOR 1000100111011101 = 1011110001011010

P#6 P#2 XOR P#4

= 0000001001000101 XOR 0100000110111000 = 0100001111111101

P#7 (P#5 * K5) mod (2^16+1)

= 1011110001011010 * 1100110111010001 mod (2^16+1) =

= 0100000000001101

P#8 (P#6 + P#7) mod (2^16)
= 0100001111111101 + 0100000000001101 mod (2^16)
= 1000010000001010

P#9 (P#8 * K6) mod (2^16+1)
= 1000010000001010 * 1001010110011101 mod (2^16+1)
= 0111111011111000

P#10 (P#7 + P#9) mod (2^16)
= 0100000000001101 + 0111111011111000 mod (2^16)
= 1011111100000101

P#11 P#1 XOR P#9
= 0011010110000111 XOR 0111111011111000 = 0100101101111111

P#12 P#3 XOR P#9
= 1000100111011101 XOR 0111111011111000 = 1111011100100101

P#13 P#2 XOR P#10
= 0000001001000101 XOR 1011111100000101 = 1011110101000000

P#14 P#4 XOR P#10
= 0100000110111000 XOR 1011111100000101 = 1111111010111101

Putaran 4

P#1 (X1 * K1) mod (2^16 + 1)
= 0100101101111111 * 1000010110111001 mod (2^16 + 1)
= 0110001001011000

P#2 (X2 + K2) mod (2^16)
= 1111011100100101 + 1011110011001100 mod (2^16)
= 1011001111110001

P#3 (X3 + K3) mod (2^16)
= 1011110101000000 + 1101000011010101 mod (2^16)
= 1000111000010101

P#4 (X4 * K4) mod (2^16 + 1)
= 1111111010111101 * 1010110111001001 mod (2^16 + 1)
= 0000111001111000

P#5 P#1 XOR P#3
= 0110001001011000 XOR 1000111000010101 = 1110110001001101

P#6 P#2 XOR P#4
= 1011001111110001 XOR 0000111001111000 = 1011110110001001

P#7 (P#5 * K5) mod (2^16+1)
= 1110110001001101 * 1010010111000001 mod (2^16+1)
= 0010111000001110

P#8 (P#6 + P#7) mod (2^16)
= 1011110110001001 + 0010111000001110 mod (2^16)
= 1110101110010111

P#9 (P#8 * K6) mod (2^16+1)
= 1110101110010111 * 1010000110111101 mod (2^16+1)
= 0111010001111000

P#10 (P#7 + P#9) mod (2^16)
= 0010111000001110 + 0111010001111000 mod (2^16)
= 1010001010000110

P#11 P#1 XOR P#9
= 0110001001011000 XOR 0111010001111000 = 0001011000100000

P#12 P#3 XOR P#9
= 1000111000010101 XOR 0111010001111000 = 1111101001101101

P#13 P#2 XOR P#10
= 1011001111110001 XOR 1010001010000110 = 0001000101110111

P#14 P#4 XOR P#10
= 0000111001111000 XOR 1010001010000110 = 1010110011111110

Putaran 5

P#1 (X1 * K1) mod (2^16 + 1)
= 0001011000100000 * 0011101100001011 mod (2^16 + 1)
= 0100111001000110

P#2 (X2 + K2) mod (2^16)

= 1111101001101101 + 0111001101111001 mod (2^16)

= 0110110111100110

P#3 (X3 + K3) mod (2^16)

= 0001000101110111 + 1001100110100001 mod (2^16)

= 1010101100011000

P#4 (X4 * K4) mod (2^16 + 1)

= 1010110011111110 * 1010101101011011 mod (2^16 + 1)

= 1011010010000000

P#5 P#1 XOR P#3

= 0100111001000110 XOR 1010101100011000 = 1110010101011110

P#6 P#2 XOR P#4

= 0110110111100110 XOR 1011010010000000 = 1101100101100110

P#7 (P#5 * K5) mod (2^16+1)

= 1110010101011110 * 1001001101001011 mod (2^16+1)

= 1010100010010011

P#8 (P#6 + P#7) mod (2^16)

= 1101100101100110 + 1010100010010011 mod (2^16)

= 1000000111111001

P#9 (P#8 * K6) mod (2^16+1)

= 1000000111111001 * 1000001110100011 mod (2^16+1)

= 1110100110110111

P#10 (P#7 + P#9) mod (2^16)

= 1010100010010011 + 1110100110110111 mod (2^16)

= 1001001001001010

P#11 P#1 XOR P#9

= 0100111001000110 XOR 1110100110110111 = 1010011111110001

P#12 P#3 XOR P#9

= 1010101100011000 XOR 1110100110110111 = 0100001010101111

P#13 P#2 XOR P#10

= 0110110111100110 XOR 1001001001001010 = 1111111110101100

P#14 P#4 XOR P#10
= 1011010010000000 XOR 1001001001001010 = 0010011011001010

Putaran 6

P#1 (X1 * K1) mod (2^16 + 1)
= 1010011111110001 * 0111101110011011 mod (2^16 + 1)
= 0010100011010101

P#2 (X2 + K2) mod (2^16)
= 0100001010101111 + 1010001100101011 (2^16)
= 1110010111011010

P#3 (X3 + K3) mod (2^16)
= 1111111110101100 + 1111001100110011 mod (2^16)
= 1111001011011111

P#4 (X4 * K4) mod (2^16 + 1)
= 0010011011001010 * 0100001101010110 mod (2^16 + 1)
= 1101101110101001

P#5 P#1 XOR P#3
= 0010100011010101 XOR 1111001011011111 = 1101101000001010

P#6 P#2 XOR P#4
= 1110010111011010 XOR 1101101110101001 = 0011111001110011

P#7 (P#5 * K5) mod (2^16+1)
= 1101101000001010 * 1011011100100110 mod (2^16+1)
= 1110011110000000

P#8 (P#6 + P#7) mod (2^16)
= 0011111001110011 + 1110011110000000 mod (2^16)
= 0010010111110011

P#9 (P#8 * K6) mod (2^16+1)
= 0010010111110011 * 1001011100000111 mod (2^16+1)
= 0100100001000010

P#10 (P#7 + P#9) mod (2^16)

= 1110011110000000 + 0100100001000010 mod (2^16)

= 0010111111000010

P#11 P#1 XOR P#9

= 0010100011010101 XOR 0100100001000010 = 0110000010010111

P#12 P#3 XOR P#9

= 1111001011011111 XOR 0100100001000010 = 1011101010011101

P#13 P#2 XOR P#10

= 1110010111011010 XOR 0010111111000010 = 1100101000011000

P#14 P#4 XOR P#10

= 1101101110101001 XOR 0010111111000010 = 1111010001101011

Putaran 7

P#1 (X1 * K1) mod (2^16 + 1)

= 0110000010010111 * 0100011011110111 mod (2^16 + 1)

= 0110000011101011

P#2 (X2 + K2) mod (2^16)

= 1011101010011101 + 0011011101000110 mod (2^16)

= 1111000111100011

P#3 (X3 + K3) mod (2^16)

= 1100101000011000 + 0101011001110110 mod (2^16)

= 0010000010001110

P#4 (X4 * K4) mod (2^16 + 1)

= 1111010001101011 * 0001011011100110 mod (2^16 + 1)

= 1011010001000110

P#5 P#1 XOR P#3

= 0110000011101011 XOR 0010000010001110 = 0100000001100101

P#6 P#2 XOR P#4

= 1111000111100011 XOR 1011010001000110 = 0100010110100101

P#7 (P#5 * K5) mod (2^16+1)

= 0100000001100101 * 1010110101101110 mod (2^16+1)

= 1100000011000111

P#8 (P#6 + P#7) mod (2^16)

= 0100010110100101 + 1100000011000111 mod (2^16)

= 0000011001101100

P#9 (P#8 * K6) mod (2^16+1)

= 0000011001101100 * 0100110100101110 mod (2^16+1)

= 1010000101111001

P#10 (P#7 + P#9) mod (2^16)

= 1100000011000111 + 1010000101111001 mod (2^16)

= 0110001001000000

P#11 P#1 XOR P#9

= 0110000011101011 XOR 1010000101111001 = 1100000110010010

P#12 P#3 XOR P#9

= 0010000010001110 XOR 1010000101111001 = 1000000111110111

P#13 P#2 XOR P#10

= 1111000111100011 XOR 0110001001000000 = 1001001110100011

P#14 P#4 XOR P#10

= 1011010001000110 XOR 0110001001000000 = 1101011000000110

Putaran 8

P#1 (X1 * K1) mod (2^16 + 1)

= 1100000110010010 * 0000111010001101 mod (2^16 + 1)

= 1000111001101010

P#2 (X2 + K2) mod (2^16)

= 1000000111110111 + 1110111001101110 mod (2^16)

= 0111000001100101

P#3 (X3 + K3) mod (2^16)

= 1001001110100011 + 1000110010101100 mod (2^16)

= 0010000001001111

P#4 (X4 * K4) mod (2^16 + 1)

= 1101011000000110 * 1110110000101101 mod (2^16 + 1)

= 0110000110011100

P#5 P#1 XOR P#3
= 1000111001101010 XOR 0010000001001111 = 1010111000100101

P#6 P#2 XOR P#4
= 0111000001100101 XOR 0110000110011100 = 0001000111111001

P#7 (P#5 * K5) mod (2^16+1)
= 1010111000100101 * 1100110111100110 mod (2^16+1)
= 1000101000101111

P#8 (P#6 + P#7) mod (2^16)
= 0001000111111001 + 1000101000101111 mod (2^16)
= 1001110000101000

P#9 (P#8 * K6) mod (2^16+1)
= 1001110000101000 * 0110011010000110 mod (2^16+1)
= 0110111001100111

P#10 (P#7 + P#9) mod (2^16)
= 1000101000101111 + 0110111001100111 mod (2^16)
= 1111100010010110

P#11 P#1 XOR P#9
= 1000111001101010 XOR 0110111001100111 = 1110000000001101

P#12 P#3 XOR P#9
= 0010000001001111 XOR 0110111001100111 = 0100111000101000

P#13 P#2 XOR P#10
= 0111000001100101 XOR 1111100010010110 = 1000100011110011

P#14 P#4 XOR P#10
= 0110000110011100 XOR 1111100010010110 = 1001100100001010

TRANSFORMASI OUTPUT

P#1 (X1 * K1) mod (2^16 +1)
= 1110000000001101 * 0101110000011101 mod (2^16 +1)
= 1011110011011100

P#2 (X2 + K2) mod (2^16)
= 1000100011110011 + 0001101111011100 mod (2^16)

= 1010010011001111

P#3 (X3 + K3) mod (2^16)

= 0100111000101000 + 1101110100011001 mod (2^16)

= 0010101101000001

P#4 (X4 * K4) mod (2^16 +1)

= 1001100100001010 * 0101100111011000 mod (2^16 +1)

= 0110010010111011

Setelah melakukkan 8 putaran dan 1 transformasi output tahap enkripsi selesai dan didapatkan chiperteks :

1011110011011100 = ¼ü

1010010011001111 = ¤Ï

0010101101000001 = +A

0110010010111011 = d»

Maka didapatkanlah chiperteks dari plainteks uinsuska adalah ¼ü ¤Ï+Ad»

4.1.3 Penyembunyian pesan

Setelah melakukkan proses enkripsi dan mendapatkan chiperteks dengan metode idea yaitu ¼ü ¤Ï+Ad» dan kemudian hasil chiperteks tersebut diubah menjadi bilangan decimal menjadi 188 220 164 207 43 65 100 187 untuk disembunyikan kedalam gambar dengan metode steganografi end of file

1. Menentukan citra digital

   Pada proses pemilihan gambar atau citra digital ini yang dipilih adalah gambar yang berformat JPG/PNG yang akan diproses, kemudian citra digital berformat JPG dan PNG tersebut akan diubah grayscale dengan ukuran 400x400 pixel.

Gambar 4.4 Citra Digital Yang Akan Disisipkan

2. Membaca nilai citra digital

Proses penyembunyian pesan pada citra digital dengan metode end of file , pada proses ini berguna untuk mencari nilai pixel dan jumlah pixel yang akan disembunyikan kedalam citra digital tersebut

xy	1	2	3	4	5	.....	394	395	396	397	398	399	400
1	15	19	42	29	5	.....	157	155	154	152	151	150	151
2	19	18	15	20	15	.....	157	156	155	153	152	152	154
3	31	12	9	24	24	.....	158	158	156	152	149	150	153
4	18	15	15	9	26	.....	161	160	158	154	150	151	153
5	96	32	27	5	12	.....	155	157	159	160	158	157	157
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....
394	192	137	200	153	182	.....	150	157	141	79	82	126	92
395	179	186	69	177	164	.....	164	148	124	100	102	104	97
396	108	172	175	198	160	.....	72	121	156	138	159	153	80
397	121	211	178	177	142	.....	73	144	110	142	156	82	87
398	199	193	138	171	214	.....	126	139	77	92	112	100	83
399	170	119	128	193	124	.....	196	186	209	117	102	81	56
400	142	201	165	149	164	.....	176	99	150	90	73	70	112

Gambar 4.5 Matrix Pixel 400x400

3. Penyembunyian pesan kecitra digital

Setelah hasil chiperteks diubah menjadi decimal selanjutnya pesan akan disembunyikan pada akhir pixel 400x400 dengan nilai decimalnya 188 220 164 207 43 65 100 187

Xy	392	393	394	395	396	397	398	400
401	188	220	164	207	43	65	100	187

Gambar 4.6 Penyispan chiperteks

4. Pesan tersimpan

Proses penyimpan akan menyimpan pesan yang telah disembunyikan pada akhir pixel tersebut.

4.1.4 Analisis MSE dan PSNR

perhitungan nilai MSE dan PSNR

Nilai citra asli dengan ukuran 400x400 pixel yang telah digrayscale AG(asli grayscale) . proses mendapatkan nilai citra dilakukkan dengan melihat hasil dari *console* dari sistem atau aplikasi tersebut.

AG(0)[244]AG(1)[240]AG(2)[218]AG(3)[231]AG(4)[252]AG(5)[253]AG(6)[251]AG(7)[229]AG(8)[240]AG(9)[229]AG(10)[229]AG(11)[225]AG(12)[236]AG(13)[234]AG(14)[243]AG(15)[245]AG(16)[225]AG(17)[248]AG(18)[240]AG(19)[227]AG(20)[219]AG(21)[226]AG(22)[237]AG(23)[236]AG(24)[213]AG(25)[227]AG(26)[221]AG(27)[228]AG(28)[190]AG(29)[116]AG(30)[245]AG(31)[213]AG(32)[240]AG(33)[173]AG(34)[230]AG(35)[243]AG(36)[243]AG(37)[227]AG(38)[228]AG(39)[227]AG(40)[193]AG(41)[211]AG(42)[242]AG(43)[218]AG(44)[221]AG(45)[178]AG(46)[200]AG(47)[182]AG(48)[221]AG(49)[221]AG(50)[226]AG(51)[229]AG(52)[224]AG(53)[203]AG(54)[222]AG(55)[45]AG(56)[20]AG(57)[33]AG(58)[33]AG(59)[30]AG(60)[28]AG(61)[31]AG(62)[29]AG(63)[35]AG(64)[33]AG(65)[33]AG(66)[32]AG(67)[30]AG(68)[31]AG(69)[31]AG(70)[32]AG(71)[32]AG(72)[30]AG(73)[31]AG(74)[31]AG(75)[30]AG(76)[30]AG(77)[31]AG(78)[32]AG(79)[32]AG(80)[31]AG(81)[31]AG(82)[31]AG(83)[32]AG(84)[31]AG(85)[31]AG(86)[34]AG(87)[37]AG(88)[37]AG(89)[36]AG(90)[33]AG(

91)[33]AG(92)[33]AG(93)[35]AG(94)[35]AG(95)[36]AG(96)[35]AG(97)[35]A
G(98)[35]AG(99)[36]AG(100)[38]AG(101)[39]AG(102)[38]AG(103)[38]AG(10
4)[36]AG(105)[37]AG(106)[37]AG(107)[37]AG(108)[39]AG(109)[39]AG(110)[
39]AG(111)[39]AG(112)[38]AG(113)[38]AG(114)[39]AG(115)[39]AG(116)[38]
AG(117)[38]AG(118)[38]AG(119)[37]AG(120)[40]AG(121)[41]AG(122)[40]A
G(123)[39]AG(124)[39]AG(125)[41]AG(126)[43]AG(127)[41]AG(128)[41]AG(
129)[41]AG(130)[40]AG(131)[40]AG(132)[42]AG(133)[44]AG(134)[43]AG(13
5)[42]AG(136)[41]AG(137)[43]AG(138)[44]AG(139)[44]AG(140)[43]AG(141)[
44]AG(142)[44]AG(143)[44]AG(144)[44]AG(145)[43]AG(146)[43]AG(147)[45]
AG(148)[46]AG(149)[46]AG(150)[46]AG(151)[46]AG(152)[46]AG(153)[47]A
G(154)[46]AG(155)[45]AG(156)[46]AG(157)[47]AG(158)[47]AG(159)[50]AG(
160)[47]AG(161)[48]AG(162)

Kemudian citra 400x400 pixel disisipkan pesan dienkripsi dan didapatkan perbedaan nilai citra tersebut EG(enkripsi grayscale)

[245]EG(16)[225]EG(17)[249]EG(18)[240]EG(19)[227]EG(20)[219]EG(21)[226
]EG(22)[237]EG(23)[236]EG(24)[213]EG(25)[226]EG(26)[221]EG(27)[228]EG(
28)[190]EG(29)[116]EG(30)[245]EG(31)[213]EG(32)[240]EG(33)[173]EG(34)[
231]EG(35)[243]EG(36)[243]EG(37)[227]EG(38)[228]EG(39)[227]EG(40)[193]
EG(41)[211]EG(42)[242]EG(43)[218]EG(44)[221]EG(45)[178]EG(46)[200]EG(
47)[182]EG(48)[221]EG(49)[221]EG(50)[226]EG(51)[229]EG(52)[224]EG(53)[
203]EG(54)[222]EG(55)[45]EG(56)[20]EG(57)[33]EG(58)[33]EG(59)[30]EG(60
)[28]EG(61)[31]EG(62)[29]EG(63)[35]EG(64)[33]EG(65)[33]EG(66)[32]EG(67)
[30]EG(68)[31]EG(69)[31]EG(70)[32]EG(71)[32]EG(72)[30]EG(73)[31]EG(74)[
31]EG(75)[30]EG(76)[30]EG(77)[31]EG(78)[32]EG(79)[32]EG(80)[31]EG(81)[
31]EG(82)[31]EG(83)[32]EG(84)[31]EG(85)[31]EG(86)[34]EG(87)[37]EG(88)[
37]EG(89)[36]EG(90)[33]EG(91)[33]EG(92)[33]EG(93)[35]EG(94)[35]EG(95)[
36]EG(96)[35]EG(97)[35]EG(98)[35]EG(99)[36]EG(100)[38]EG(101)[39]EG(10
2)[38]EG(103)[38]EG(104)[36]EG(105)[37]EG(106)[37]EG(107)[37]EG(108)[3
9]EG(109)[39]EG(110)[39]EG(111)[39]EG(112)[38]EG(113)[38]EG(114)[38]E

G(115)[38]EG(116)[38]EG(117)[38]EG(118)[38]EG(119)[37]EG(120)[40]EG(121)[41]EG(122)[40]EG(123)[39]EG(124)[39]EG(125)[41]EG(126)[43]EG(127)[41]EG(128)[41]EG(129)[41]EG(130)[40]EG(131)[40]EG(132)[42]EG(133)[44]EG(134)[43]EG(135)[42]EG(136)[41]EG(137)[43]EG(138)[44]EG(139)[44]EG(140)[43]EG(141)[44]EG(142)[44]EG(143)[44]EG(144)[44]EG(145)[43]EG(146)[43]EG(147)[45]EG(148)[46]EG(149)[46]EG(150)[46]EG(151)[46]EG(152)[45]EG(153)[47]EG(154)[46]EG(155)[45]EG(156)[46]EG(157)[46]EG(158)[47]EG(159)[50]EG(160)[47]EG(161)[48]EG(162)[48]EG(163)[49]EG(164)[51]EG(165)[52]EG(166)[51]EG(167)[51]EG(168)[51]EG(169)[51]EG(170)[51]EG(171)[52]EG(172)[53]EG(173)[54]EG(174)[54]EG(175)[55]EG(176)[55]EG(177)[53]EG(178)[50]EG(179)[49]EG(180)[53]EG(181)[58]EG(182)[57]EG(183)[54]EG(184)[53]EG(185)[53]EG(186)[54]EG(187)[55]EG(188)[56]EG(189)[57]EG(190)[58]EG(191)[59]EG(192)[58]EG(193)[58]EG(194)[58]EG(195)[58]EG(196)[58]EG(197)[58]EG(198)[57]EG(199)[57]EG(200)[56]EG(201)[58]EG(202)[59]EG(203)[59]EG(204)[59]EG(205)[60]EG(206)[63]EG(207)[64]EG(208)[60]EG(209)[59]EG(210)

perhitungan nilai MSE dan PNSR

contoh sampel nilai citra asli dengan ukuran 3x3 pixel sedangkan mencari manual dengan pixel 400x400 perhitungan sangat panjang .

contoh citra asli 3x3

244	240	218
229	240	229
234	243	245

Nilai pixel citra akhir 4x3 pixel

contoh citra akhir 3x3

245	225	249
237	236	213
190	116	245

Pertama Hitung nilai MSE terlebih dahulu

$$MSE=\frac{(245-244)^2(225-240)^2(249-218)^2(237-229)^2(236-240^2(213-229)^2(190-234)^2(116-243)^2(245-245)^2}{3x3}$$

$$MSE = \frac{1+15+31+8+4+16+44+127+0}{9}$$

MSE = 27.33

Kemudian menghitung PNSR :

PNSR = 10 log_{10} $(\frac{Cmax^2}{MSE})$

PNSR = 10 log_{10} $(\frac{249^2}{MSE})$

PNSR = 19,11

Maka dari itu untuk mencari nilai pixel 400x400 dilihat langsung dari console sistem.

MSE = 4.111111111111108 / (400*400)=0.00002569444444444424

PSNR 10*log(10)*(pow(255,2)/0.00002569444444444424)=58271583180.4299

4.1.5 Proses Dekripsi

Proses kebalikan dari enkripsi dimana proses ini mengubah hasil chipertext menjadi plainteks dengan menggunakan algoritma kriptografi idea untuk mendapatkan plainteks tersebut dibutuhkan chiperteks dan dibutuhkan kunci public yang sebelumnya yang digunakan pada enkripsi yang berbeda hanya pada invers pekalian dan invers penjumlahan, pada setiap langkah putaran dekripsi yang terdiri 8 putaran proses dekripsi adalah sebegai berikut:

Contoh chiperteks dan kunci:

Chiperteks : ¼ü ¤Ï+Ad»

Kunci : kriptostegano345

Chiperteks diubah terlebih dahulu ke ascii

X1 = 1011110011011100 = ¼ü

X2 = 1010010011001111 = ¤Ï

X3 = 0010101101000001 = +A

X4 = 0110010010111011 = d»

Kemudian pada proses dekripsi untuk putaran 1 untuk X1,X2,X3,X4 yang dimaksud X1,X2,X3,X4 iyalah chiperteks hasil enkripsi dan sedangkan K1,K2,K3,K4,K5,K6 adalah sub kunci dekripsi yang sudah ditentukan.

Putaran 1

P#1 (X1 * K1) mod (2^16 + 1)
= 1011110011011100 * 10001000010100 mod (2^16+1)
= 1110000000001101

P#2 (X2 + K2) mod (2^16)
= 1010010011001111 + 1110010000100100 mod (2^16)
= 1000100011110011

P#3 (X3 + K3) mod (2^16)
= 0010101101000001 + 0010001011100111 mod (2^16)
= 0100111000101000

P#4 (X4 * K4) mod (2^16 + 1)
= 0110010010111011 * 00111010100101100 mod (2^16+1)
= 1001100100001010

P#5 P#1 XOR P#3
= 1110000000001101 XOR 0100111000101000 = 1010111000100101

P#6 P#2 XOR P#4
= 1000100011110011 XOR 1001100100001010 = 0001000111111001

P#7 (P#5 * K5) mod (2^16+1)
= 1010111000100101 * 1100110111100110 mod (2^16+1)
= 1000101000101111

P#8 (P#6 + P#7) mod (2^16)
= 0001000111111001 + 1000101000101111 mod (2^16)
= 1001110000101000

P#9 (P#8 * K6) mod (2^16+1)
= 1001110000101000 * 0110011010000110 mod (2^16+1)
= 0110111001100111

P#10 (P#7 + P#9) mod (2^16)
= 1000101000101111 + 0110111001100111 mod (2^16)
= 1111100010010110

P#11 P#1 XOR P#9
= 1110000000001101 XOR 0110111001100111 = 1000111001101010

P#12 P#3 XOR P#9
= 0100111000101000 XOR 0110111001100111 = 0010000001001111

P#13 P#2 XOR P#10
= 1000100011110011 XOR 1111100010010110 = 0111000001100101

P#14 P#4 XOR P#10
= 1001100100001010 XOR 1111100010010110 = 0110000110011100

Putaran 2

P#1 (X1 * K1) mod (2^16 + 1)
= 1000111001101010 * 0111000110001001 mod (2^16+1)
= 1100000110010010

P#2 (X2 + K2) mod (2^16)
= 0010000001001111 + 0001000110010010 mod (2^16)
= 0011000111100001

P#3 (X3 + K3) mod (2^16)
= 0111000001100101 + 0111001101010100 mod (2^16)
= 1110001110111001

P#4 (X4 * K4) mod (2^16 + 1)
= 0110000110011100 * 1000111001111010 mod (2^16+1)
= 1101011000000110

P#5 P#1 XOR P#3
= 1100000110010010 XOR 1110001110111001 = 0010001000101011

P#6 P#2 XOR P#4

= 0011000111100001 XOR 1101011000000110 = 1110011111100111

P#7 (P#5 * K5) mod (2^16+1)

= 0010001000101011 * 1010110101101110 mod (2^16+1)

= 1010011001010101

P#8 (P#6 + P#7) mod (2^16)

= 1110011111100111 + 1010011001010101 mod (2^16)

= 1000111000111100

P#9 (P#8 * K6) mod (2^16+1)

= 1000111000111100 * 0100110100101110 mod (2^16+1)

= 0110111111100111

P#10 (P#7 + P#9) mod (2^16)

= 1010011001010101 + 0110111111100111 mod (2^16)

= 0001011000111100

P#11 P#1 XOR P#9

= 1100000110010010 XOR 0110111111100111 = 1010111001110101

P#12 P#3 XOR P#9

= 1110001110111001 XOR 0110111111100111 = 1000110001011110

P#13 P#2 XOR P#10

= 0011000111100001 XOR 0001011000111100 = 0010011111011101

P#14 P#4 XOR P#10

= 1101011000000110 XOR 0001011000111100 = 1100000000111010

Putaran 3

P#1 (X1 * K1) mod (2^16 + 1)

= 1010111001110101 * 1010101100101100 mod (2^16+1)

= 1010111001110111

P#2 (X2 + K2) mod (2^16)

= 1000110001011110 + 1100100010111010 mod (2^16)

= 0101010100011000

P#3 (X3 + K3) mod (2^16)

= 0010011111011101 + 1010100110001010 mod (2^16)

= 1101000101100111

P#4 (X4 * K4) mod (2^16 + 1)

= 1100000000111010 * 0010000001000110 mod (2^16+1)

= 1011011110100001

P#5 P#1 XOR P#3

= 1010111001110111 XOR 1101000101100111 = 0111111100010000

P#6 P#2 XOR P#4

= 0101010100011000 XOR 1011011110100001 = 1110001010111001

P#7 (P#5 * K5) mod (2^16+1)

= 0111111100010000 * 1011011100100110 mod (2^16+1)

= 1111000101111010

P#8 (P#6 + P#7) mod (2^16)

= 1110001010111001 + 1111000101111010 mod (2^16)

= 1101010000110011

P#9 (P#8 * K6) mod (2^16+1)

= 1101010000110011 * 1001011100000111 mod (2^16+1)

= 0110010100110110

P#10 (P#7 + P#9) mod (2^16)

= 1111000101111010 + 0110010100110110 mod (2^16)

= 0101011010110000

P#11 P#1 XOR P#9

= 1010111001110111 XOR 0110010100110110 = 1100101101000001

P#12 P#3 XOR P#9

= 1101000101100111 XOR 0110010100110110 = 1011010001010001

P#13 P#2 XOR P#10

= 0101010100011000 XOR 0101011010110000 = 0000001110101000

P#14 P#4 XOR P#10

= 1011011110100001 XOR 0101011010110000 = 1110000100010001

Putaran 4

P#1 (X1 * K1) mod (2^16 + 1)
= 1100101101000001 * 0010011110110001 mod (2^16+1)
= 0100111101101110

P#2 (X2 + K2) mod (2^16)
= 1011010001010001 + 0101110011010101 mod (2^16)
= 0001000100100110

P#3 (X3 + K3) mod (2^16)
= 0000001110101000 + 0000110011001101 mod (2^16)
= 0001000001110101

P#4 (X4 * K4) mod (2^16 + 1)
= 1110000100010001 * 0101111111001110 mod (2^16+1)
= 0001011001110100

P#5 P#1 XOR P#3
= 0100111101101110 XOR 0001000001110101 = 0101111100011011

P#6 P#2 XOR P#4
= 0001000100100110 XOR 0001011001110100 = 0000011101010010

P#7 (P#5 * K5) mod (2^16+1)
= 0101111100011011 * 1001001101001011 mod (2^16+1)
= 0010011100110001

P#8 (P#6 + P#7) mod (2^16)
= 0000011101010010 + 0010011100110001 mod (2^16)
= 0010111010000011

P#9 (P#8 * K6) mod (2^16+1)
= 0010111010000011 * 1000001110100011 mod (2^16+1)
= 1000111001111111

P#10 (P#7 + P#9) mod (2^16)
= 0010011100110001 + 1000111001111111 mod (2^16)
= 1011010110110000

P#11 P#1 XOR P#9
= 0100111101101110 XOR 1000111001111111 = 1100000100010001

P#12 P#3 XOR P#9
= 0001000001110101 XOR 1000111001111111 = 1001111000001010
P#13 P#2 XOR P#10
= 0001000100100110 XOR 1011010110110000 = 1010010010010110
P#14 P#4 XOR P#10
= 0001011001110100 XOR 1011010110110000 = 1010001111000100

Putaran 5
P#1 (X1 * K1) mod (2^16 + 1)
= 1100000100010001 * 1010000100011011 mod (2^16+1)
= 1001010001001100
P#2 (X2 + K2) mod (2^16)
= 1001111000001010 + 1000110010000111 mod (2^16)
= 0010101010010001
P#3 (X3 + K3) mod (2^16)
= 1010010010010110 + 0110011001011111 mod (2^16)
= 0000101011110101
P#4 (X4 * K4) mod (2^16 + 1)
= 1010001111000100 * mod (2^16+1)
= 1100110111101110
P#5 P#1 XOR P#3
= 1001010001001100 XOR 0000101011110101 = 1001111010111001
P#6 P#2 XOR P#4
= 0010101010010001 XOR 1100110111101110 = 1110011101111111
P#7 (P#5 * K5) mod (2^16+1)
= 1001111010111001 * 1010010111000001 mod (2^16+1)
= 0111111110110101
P#8 (P#6 + P#7) mod (2^16)
= 1110011101111111 + 0111111110110101 mod (2^16)
= 0110011100110100
P#9 (P#8 * K6) mod (2^16+1)

= 0110011100110100 * 1101000110111101 mod (2^16+1)

= 0101000011010111

P#10 (P#7 + P#9) mod (2^16)

= 0111111110110101 + 0101000011010111 mod (2^16)

= 1101000011010111

P#11 P#1 XOR P#9

= 1001010001001100 XOR 0101000011010111 = 1100010010011011

P#12 P#3 XOR P#9

= 0000101011110101 XOR 0101000011010111 = 0101101000100010

P#13 P#2 XOR P#10

= 0010101010010001 XOR 1101000011010111 = 1111101000011101

P#14 P#4 XOR P#10

= 1100110111101110 XOR 1101000011010111 = 0001110101100010

Putaran 6

P#1 (X1 * K1) mod (2^16 + 1)

= 1100010010011011 * 1100110010000110 mod (2^16+1)

= 1101000000010001

P#2 (X2 + K2) mod (2^16)

= 0101101000100010 + 0100001100110100 mod (2^16)

= 1001110101010110

P#3 (X3 + K3) mod (2^16)

= 1111101000011101 + 0010111100101011 mod (2^16)

= 0010100101001000

P#4 (X4 * K4) mod (2^16 + 1)

= 0001110101100010 * 1110011101100110 mod (2^16+1)

= 0000100001111101

P#5 P#1 XOR P#3

= 1101000000010001 XOR 0010100101001000 = 1111100101011001

P#6 P#2 XOR P#4

= 1001110101010110 XOR 0000100001111101 = 1001010100101011

P#7 (P#5 * K5) mod (2^16+1)
= 1111100101011001 * 1100110111010001 mod (2^16+1)
= 0000111000110010

P#8 (P#6 + P#7) mod (2^16)
= 1001010100101011 + 0000111000110010 mod (2^16)
= 1010001101011101

P#9 (P#8 * K6) mod (2^16+1)
= 1010001101011101 * 1001010110011101 mod (2^16+1)
= 1111000110010001

P#10 (P#7 + P#9) mod (2^16)
= 0000111000110010 + 1111000110010001 mod (2^16)
= 1111111111000011

P#11 P#1 XOR P#9
= 1101000000010001 XOR 1111000110010001 = 0010000110000000

P#12 P#3 XOR P#9
= 0010100101001000 XOR 1111000110010001 = 1101100011011001

P#13 P#2 XOR P#10
= 1001110101010110 XOR 1111111111000011 = 0110001010010101

P#14 P#4 XOR P#10
= 0000100001111101 XOR 1111111111000011 = 1111011110111110

Putaran 7

P#1 (X1 * K1) mod (2^16 + 1)
= 0010000110000000 * 1100100100101001 mod (2^16+1)
= 1100001100101110

P#2 (X2 + K2) mod (2^16)
= 1101100011011001 + 1001100110011000 mod (2^16)
= 0111001001110001

P#3 (X3 + K3) mod (2^16)
= 0110001010010101 + 1001010100101010 mod (2^16)
= 1111011110111111

P#4 (X4 * K4) mod (2^16 + 1)

= 1111011110111110 * 1100000011010111 mod (2^16+1)

= 1101010111110101

P#5 P#1 XOR P#3

= 1100001100101110 XOR 1111011110111111 = 0011010010010001

P#6 P#2 XOR P#4

= 0111001001110001 XOR 1101010111110101 = 1010011110000100

P#7 (P#5 * K5) mod (2^16+1)

= 0011010010010001 * 1110100011001010 mod (2^16+1)

= 1011001010011110

P#8 (P#6 + P#7) mod (2^16)

= 1010011110000100 + 1011001010011110 mod (2^16)

= 0101101000100010

P#9 (P#8 * K6) mod (2^16+1)

= 0101101000100010 * 1100111011000010 mod (2^16+1)

= 0110000011111001

P#10 (P#7 + P#9) mod (2^16)

= 1011001010011110 + 0110000011111001 mod (2^16)

= 0001001110010111

P#11 P#1 XOR P#9

= 1100001100101110 XOR 0110000011111001 = 1010001111010111

P#12 P#3 XOR P#9

= 1111011110111111 XOR 0110000011111001 = 1001011101000110

P#13 P#2 XOR P#10

= 0111001001110001 XOR 0001001110010111 = 0110000111100110

P#14 P#4 XOR P#10

= 1101010111110101 XOR 0001001110010111 = 1100011001100010

Putaran 8

P#1 (X1 * K1) mod (2^16 + 1)
= 1010001111010111 * 1111001101100011 mod (2^16+1)
= 1101010101100010

P#2 (X2 + K2) mod (2^16)
= 1001011101000110 + 1100101111001011 mod (2^16)
= 0110001100010001

P#3 (X3 + K3) mod (2^16)
= 0110000111100110 + 0001111100011000 mod (2^16)
= 1000000011111110

P#4 (X4 * K4) mod (2^16 + 1)
= 1100011001100010 * 1001111101001000 mod (2^16+1)
= 0010111000100010

P#5 P#1 XOR P#3
= 1101010101100010 XOR 1000000011111110 = 0101010110011100

P#6 P#2 XOR P#4
= 0110001100010001 XOR 0010111000100010 = 0100110100110011

P#7 (P#5 * K5) mod (2^16+1)
= 0101010110011100 * 0110010101100111 mod (2^16+1)
= 1101101111011100

P#8 (P#6 + P#7) mod (2^16)
= 0100110100110011 + 1101101111011100 mod (2^16)
= 0010100100001111

P#9 (P#8 * K6) mod (2^16+1)
= 0010100100001111 * 0100001111010010 mod (2^16+1)
= 0100001111010010

P#10 (P#7 + P#9) mod (2^16)
= 1101101111011100 + 0100001111010010 mod (2^16)
= 0001111110101110

P#11 P#1 XOR P#9
= 1101010101100010 XOR 0100001111010010 = 1001011010110000

P#12 P#3 XOR P#9

= 1000000011111110 XOR 0100001111010010 = 1100001100101100

P#13 P#2 XOR P#10

= 0110001100010001 XOR 0001111110101110 = 0111110010111111

P#14 P#4 XOR P#10

= 0010111000100010 XOR 0001111110101110 = 0011000110001100

TRANSFORMASI OUTPUT

P#1 (X1 * K1) mod (2^16 +1)

= 1111101001111100 * 0010111000011110 mod (2^16 +1)

= 0111010101101001

P#2 (X2 + K2) mod (2^16)

= 1101011111100011 + 1001011010010000 mod (2^16)

= 0110111001110011

P#3 (X3 + K3) mod (2^16)

= 1110100111100010 + 1000101110010001 mod (2^16)

= 0111010101110011

P#4 (X4 * K4) mod (2^16 +1)

= 0000101010000111 * 0110011001110001 mod (2^16 +1)

= 0110101101100001

Setelah melakukkan 8 putaran dan 1 transformasi output tahap enkripsi selesai dan didapatkan chiperteks :

0111010101101001 = ui

0110111001110011 = ns

0111010101110011 = us

0110101101100001 = ka

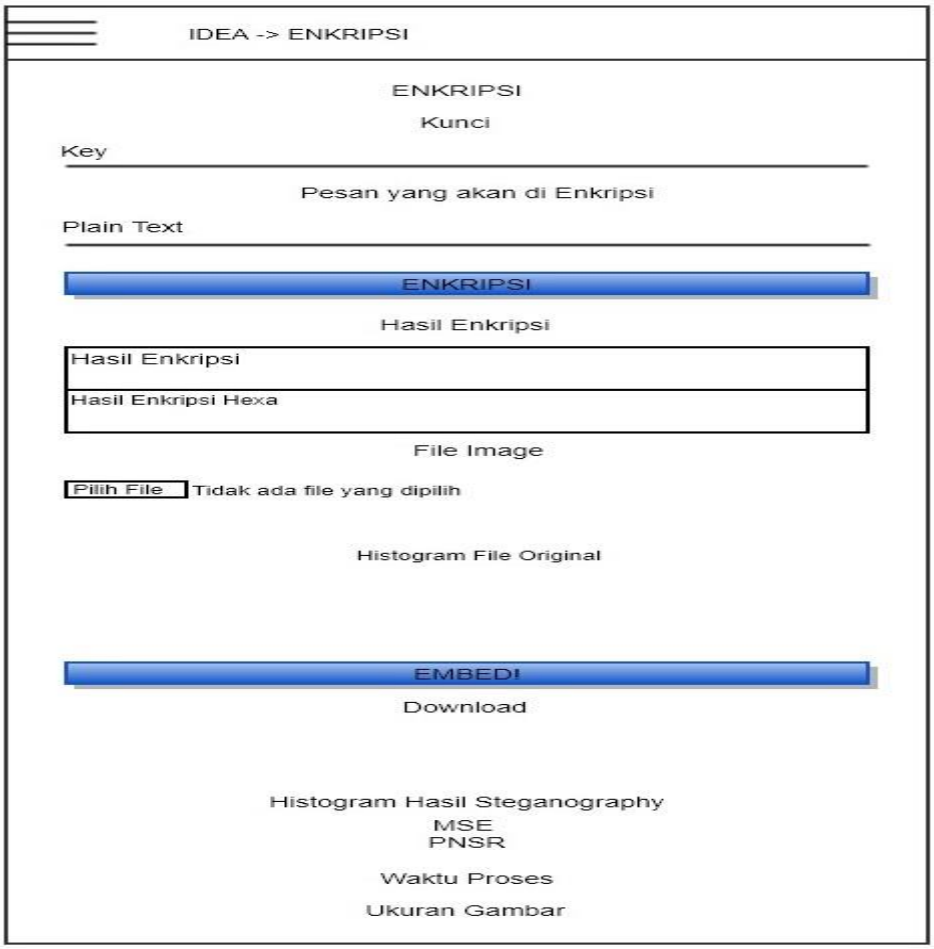
Maka didapatkanlah palinteks dari chiperteks ¼ü ¤Ï+Ad» adalah uinsuska

4.2 Perancangan Aplikasi

Perancangan aplikasi merupakan gambaran atau sketsa yang akan dibuat untuk merancang konsep aplikasi tersebut

4.2.1 Perancangan Antarmuka Enkripsi

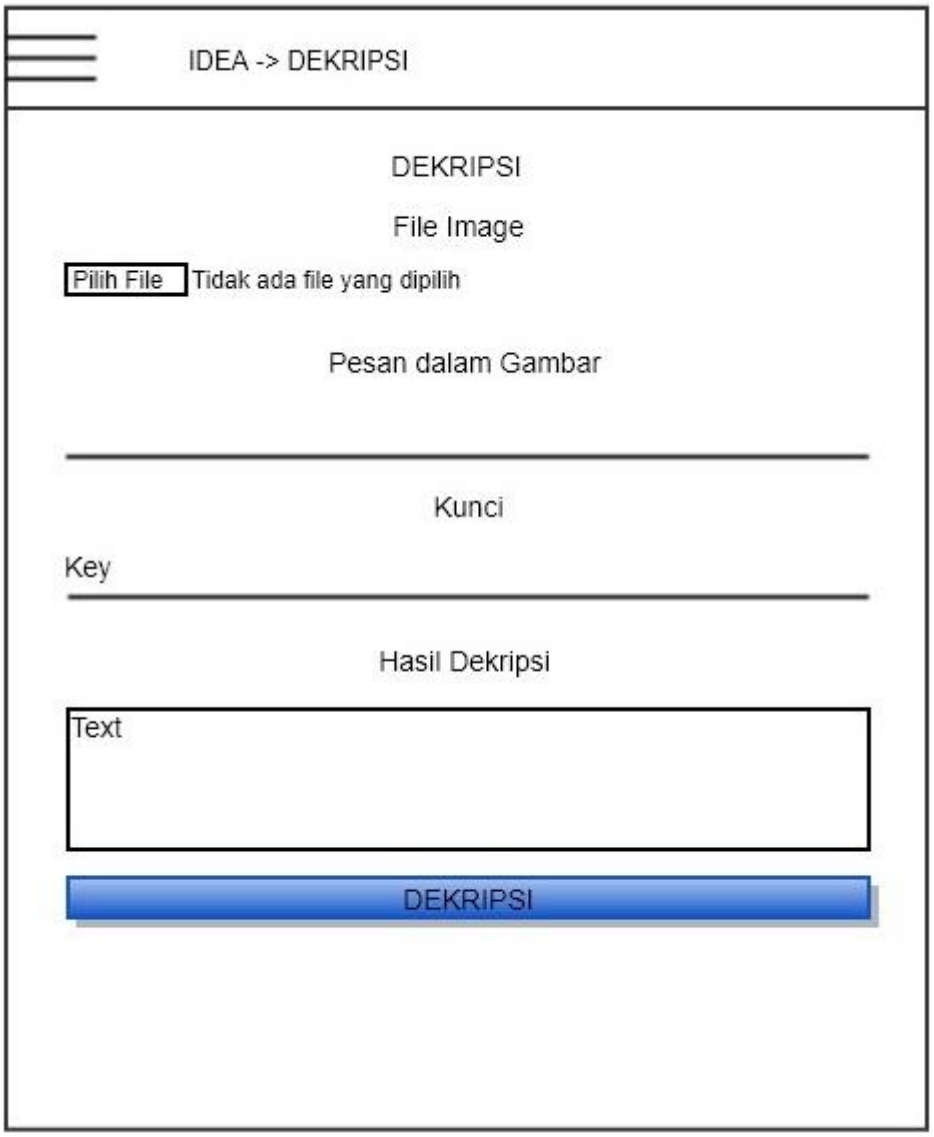
Perancangan menu antarmuka enkripsi ini berisikan tentang rancangan memasukan kunci public dan memasukan plainteks yang akan diubah menjadi chiperteks, memasukan gambar berformat .jpg atau .png yang akan dipergunakan untuk proses penyembunyian pesan, dan hasilnya gambar yang telah disembunyikan pesan enkripsi dan juga membandingkan histogram asli dengan histogram yang telah disembunyikan pesan dan menghitung ukuran pixel gambar , waktu proses penyembunyian pesan dan terakhir menghitung MSE dan PSNR gambar tersebut.



Gambar 4.7 Perancangan Antarmuka Enkripsi

4.2.2 Perancangan Antarmuka Dekripsi

Pada tahap perancangan antarmuka dekripsi ini berisi tentang rancangan memaskan hasil gambar citra yang telah disembunyikan pesan kemudian pesan digambar akan melakukan ektaksi pesan didapatkan pesan chiperteksnya dan memasukan kunci public untuk memulai dekripsi selanjutnya terdapat tombol dekripsi untuk menampilkan pesan asli.



Gambar 4.8 Perancangan Antarmuka Dekripsi

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukkan maka didapatkanlah kesimpulan sebagai berikut:

1. Aplikasi sudah berjalan dengan baik sesuai dengan yang sudah dilakukan pengujian dan di implementasikan sebelumnya.
2. Enkripsi dan dekripsi dengan algoritma *Idea* menggunakan kunci 128bit dan plainteks 64 bit menghasilkan chiperteks yang tidak bisa dibaca dan dirusak pesannya dan hanya dapat diketahui dengan kunci public *Idea*.
3. Proses mendapatkan nilai MSE dan PSNR untuk mengetahui gambar atau citra yang akan disisipkan tersebut layak digunakan dengan algoritma *Eof.*

6.2 Saran

Adapun saran yang dapat diberikan untuk pengembangan penelitian ini adalah Pengembangan aplikasi pengamanan teks dapat dilakukkan dengan file dokumen yang lain seperti *.doc, *.pdf dan lain-lain. pengujian lebih banyak lagi seperti pengujian steganografi robustness yaitu *cropping,resize, blur* dan lain-lain

DAFTAR PUSTAKA

Arini, G. M. (2012). Pengamanan Pesan Steganografi dengan Metode LSB Berlapis Enkripsi dalam PHP.

Dharwiyanti, S. (2003). Pengantar Unified Modeling Language (UML), 1–13.

Eriyanto. (2001). *Analisis Wacana, Pengantar Analisis Teks Media*. Yogyakarta: LKiS.

Ghazali Moenandar Male, Wirawan, E. S. (2012). Analisa Kualitas Citra Pada Steganografi Untuk Aplikasi e-Government, 1–9.

Hendrawan, R. (2015). Metode Pengamanan Untuk File Bertipe Dokumen Menggunakan Kombinasi Algoritma Kriptografi International Data Encriyption Algorithm (IDEA) Dan Steganografi Least Significant Bit (LSB).

Krisnawati. (2008). Metode Least Significant Bit (LSB) Dan End Of File (EOF) Untuk Menyisipkan Teks Ke dalam Citra Grayscale, *2008*(semnasIF), 39–44.

Munir, R. (2004). *Pengolahan Citra Digital*. Bandung: Informatika.

Munir, R. (2006). *Kriptografi*. Bandung: Informatika.

Munir, R. (2012). Algoritma Enkripsi Selektif Citra Digital dalam Ranah Frekuensi Berbasis Permutasi Chaos, *10*(2), 66–72.

Muslih, E. H., & rachmawanto. (2016). Pengamanan File Multimedia Dengan Metode Steganografi End Of File Untuk Menjaga Kerahasiaan Pesan, *15*(1), 1–6.

Onno W.P., A. A. W. (2000). *Mengenal eCommerce*. Jakarta: Elex Media Komputindo.

Rachmawati, D., & Candra, A. (2015). Implementasi Kombinasi Caesar dan Affine Cipher untuk Keamanan Data Teks, *1*(2).

Safaat, N. (2012). *Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android.* Pasar Buku Palasari No. 82, Bandung: Informatika.

Sembiring, S. (2013). Perancang Aplikasi Steganografi Untuk Menyisipkan Pesan Teks Pada Gambar Dengan Metode End Of File, 45–51.

Tri Andriyanto, D. L. C. P. (2008). Studi dan perbandingan algoritma idea dan

algoritma blowfish, (Kommit), 20–21.

Ukkas, M. I., Andrea, R., Baretto, A., & Anggen, P. (n.d.). Teknik Pengamanan Data Dengan Seganografi Metode End Of File (EOF) Dan Kriptografi Vernam Chiper, 20–26.

Utami, E. (2007). Implementasi Steganografi teknik Eof Dengan Gabungan Enkripsi Rijndael, Shift Chiper Dan Fungsi Hash MD5, *2007*(November), 1–16.

Wijaya, E. S., & Prayudi, Y. (2004). Konsep Hidden Message Menggunakan Teknik Steganografi Dynamic Cell Spreading, *2*(1), 23–38.

DAFTAR RIWAYAT HIDUP

Nama Lengkap	:	Rivalza Fahlevi
Tempat/Tanggal Lahir	:	Pekanbaru / 02 Oktober1994
Nama Ayah	:	M.Reza Fahlevi
Nama Ibu	:	Nurbaiti
Anak ke	:	1
Jumlah Sdr.	:	2

Nama Sdr. : Gilang Ramadhan Fahlevi

Alamat : Jl. Melati Gg.Aster No.12 Pekanbaru

E-mail : rivalza.fahlevi@students.uin-suska.ac.id

PENDIDIKAN

- ✓ Tahun 1999-2000 : TK Hikmah Pekanbaru
- ✓ Tahun 2000-2006 : SD Negeri 009 Pekanbaru
- ✓ Tahun 2006-2009 : SMP Negeri 2 Pekanbaru
- ✓ Tahun 2009-2012 : SMA Negeri 2 Pekanbaru
- ✓ Tahun 2013-2019 : Universitas Islam Negeri Sultan Syarif Kasim Riau, Jurusan Teknik Informatika.