ANALISA METODE *END OF FILE* DALAM PENYISIPAN PESAN TEKS TERENKRIPSI DENGAN *HILL CIPHER* PADA GAMBAR

Disusun dan Diajukan untuk Memenuhi Persyaratan Ujian Akhir Memperoleh
Gelar Sarjana Komputer pada Fakultas Sains dan Teknologi
Universitas Pembangunan Panca Budi
Medan

SKRIPSI

OLEH

NAMA	**: MHD. ANSHOR HARAHAP**
NPM	**: 1724370457**
PROGRAM STUDI	**: SISTEM KOMPUTER**

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS PEMBANGUNAN PANCA BUDI
MEDAN
2019

ABSTRAK

Menggabungkan antara kriptografi dan steganografi dapat dilakukan untuk lebih meningkatkan pengamanan pada pesan yang akan disisipkan. Pesan teks yang akan disisipkan, dienkripsi terlebih dahulu dengan teknik kriptografi, setelah *ciphertext* dihasilkan, kemudian disisipkan pada *file* gambar dengan teknik steganografi. Hal ini dilakukan agar pesan teks yang telah dienkripsi tidak menimbulkan kecurigaan pada orang banyak saat melihat *ciphertext* dari pesan teks yang dihasilkan. Kecurigaan yang dimaksud adalah susunan huruf yang tak memiliki arti pada *ciphertext* akan mengundang orang untuk berfikir bahwa ada pesan yang disisipkan/ rahasia dibalik susunan huruf tersebut.

***Kata kunci** : Penyandian Pesan Teks, End Of File, Hill Cipher.*

DAFTAR GAMBAR

Halaman

DAFTAR ISI

Halaman

DAFTAR ISTILAH

STEGANOGRAPHY **:** Steganografi (*Steganography*) adalah ilmu dan seni menyembunyikan pesan rahasia (*hiding message*) sedemikian sehingga keberadaan (eksitensi) pesan tidak terdeteksi oleh indera manusia. Kata steganografi berasal dari bahasa *Yunani* yang berarti "tulisan tersembunyi" (*covered writing*). Steganografi membutuhkan dua properti yaitu wadah penampung dan data rahasia yang akan disembunyikan.

UML **:** ***Unified Modelling Language*** (UML) adalah sebuah bahasa untuk menentukan, visualisasi, mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak. *Artifact* dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya.

PNG **:** ***Portable Network Graphics*** (PNG) adalah salah satu format penyimpanan citra yang menggunakan metode pemadatan yang tidak menghilangkan bagian dari citra tersebut (*lossless compression*). PNG dibaca "ping", namun biasanya dieja apa adanya - untuk menghindari kerancuan dengan istilah "ping" pada jaringan komputer.

DAFTAR LAMPIRAN

Halaman

DAFTAR TABEL

Halaman

KATA PENGANTAR

Puji sukur kehadirat Allah SWT Tuhan yang Maha Esa karena dengan berkat dan kasih anugrahnya-Nya penulis masih diberikan kesehatan sehingga akhirnya dapat menyelesaikan skripsi ini.

Tugas akhir disusun berdasarkan hasil penelitian yang dilaksanakan pada Dinas Kesehatan Provinsi Sumatera Utara dengan judul : "**Analisa Metode *End Of File* Dalam Penyisipan Pesan Teks Terenkripsi Dengan *Hill Cipher* Pada Gambar**".

Dalam kesempatan ini, penulis mengucapkan terima kasih yang sebesar-besarnya kepada banyak pihak yang telah membantu dalam penyelesaian penyusunan Tugas Akhir ini.

Penulis ingin mengucapkan terima kasih kepada :

1. Orang Tua Bapak Zainullah Harahap dan Ibu. Rosmiati
2. Rektor Universitas Pembangunan Panca Budi, Bapak Dr. H. Muhammad Isa Indrawan, S.E, M.M
3. Rektor I, Bapak Ir. Bhakti Alamsyah, M.T, Ph.D
4. Dekan Fakultas Sains dan Teknologi, Ibu Sri Shindi Indira, ST., M.Sc
5. Ketua Program Studi Sistem Komputer, Bapak Muhammad Iqbal, S.Kom, M.Kom
6. Dosen Pembimbing I, Bapak Zulham Sitorus, S.Kom, M.Kom
7. Dosen Pembimbing II, Bapak Muhammad Iqbal, S.Kom, M.Kom
8. Ka. Sub. Bag Umum dan Kepegawaian Dinas Kesehatan Provsu Bapak Ardi Taufik Simanjuntak, SE, M.SP
9. Untuk Seluruh Keluarga (Istri, Anak, Abang dan Adik-adik ku)
10. Teman-teman Di Dinas Kesehatan Provinsi Sumatera

Penulis juga menyadari bahwa penyusunan Tugas Akhir ini belum sempurna baik dalam penulisan maupun isi disebabkan keterbatasan kemampuan penulis. Oleh karena itu, penulis mengharapkan kritik dan saran yang sifatnya membangun dari pembaca untuk penyempurnaan isi Tugas Akhir ini.

Medan, 28 Agustus 2019
Penulis

Mhd. Anshor Harahap
1724370457

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Dieraglobalisasi sekarang ini perkembangan teknologi begitu pesat, metode penyampaian pesan rahasia pun semakin beragam. Semakin meningkatnya perkembangan komunikasi data membuat semakin pentingnya aspek keamanan dan kerahasiaan data. Teknik-teknik keamanan data/informasi terbagi dari beberapa teknik yaitu kriptografi, steganografi dan watermarking. Steganografi adalah seni menyembunyikan pesan teks sedemikian rupa sehingga tak seorang pun selain pengirim dan penerima yang dituju, mengetahui keberadaan informasi tersebut. Steganografi membutuhkan dua properti yaitu media penampung dan pesan rahasia. Berbeda dengan kriptografi yaitu seni merahasiakan pesan yang bertujuan untuk membuat pesan tidak dapat dibaca oleh pihak ketiga, tetapi tidak menyembunyikan pesan rahasia tersebut (Desi Lilyani, 2010, 3). Terdapat berbagai bentuk pesan rahasia seperti pesan teks, pesan citra, pesan *suara* dan pesan *gambar* yang umum digunakan. Untuk mengamankan pesan teks dapat dilakukan dengan berbagai macam teknik kriptografi. Salah satunya adalah megamankan pesan teks menggunakan kriptografi kunci asimetris. Kriptografi kunci asimetris terdiri dari dua kunci, yaitu kunci publik dan kunci privat. Dalam kriptografi kunci asimetris, kunci publik berfungsi untuk mengenkripsi suatu pesan dan kunci privat berfungsi untuk mendekripsi suatu pesan. Sehingga tingkat

keamanan suatu pesan lebih baik dibandingkan menggunakan kriptografi kunci simetris yang hanya memiliki satu kunci privat saja.

Terdapat berbagai macam metode kriptografi kunci asimetris yang telah digunakan. Salah satunya adalah algoritma Rabin *Public Key*. Algoritma Rabin *Public Key* diperkenalkan oleh Michael O. Rabin pada tahun 1979. Algoritma Rabin menggunakan pemfaktoran bilangan untuk melakukan pengamanan. Metode pemfaktoran bilangan secara cepat sampai saat ini belum terpecahkan. Selain itu, Rabin *Public Key* ini akan menghasilkan empat kemungkinan hasil pendekripsian yang mengharuskan si penerima pesan menentukan hasil dekripsi yang benar.

Namun, teknik kriptografi yang sifatnya mengacak suatu pesan rahasia menimbulkan kecurigaan. Sehingga muncullah teknik steganografi yang merupakan pengembangan dari kriptografi. Steganografi ialah penyembunyian pesan dalam sebuah media dan bersifat tidak mengacak isi *file*. Sehingga, *file* yang disisipkan tidak mencurigakan. Saat ini telah ada beberapa metode steganografi yang umum. Salah satunya adalah metode *End of File* (EOF). Pada metode *End of File*, pesan akan disisipkan pada akhir nilai *file*.

Salah satu cara yang dapat dilakukan untuk memilih metode yang terbaik adalah dengan menguji atau membandingkan antara kedua metode tersebut. Adapun perbandingan yang ingin diuji untuk memilih teknik penyisipan yang lebih baik adalah dengan membandingkan berapa banyak karakter pesan yang dapat disisipkan ke dalam citra digital, berapa perbandingan size file citra dan

bagaimana hasil citra setelah dilakukan penyisipan pesan ke dalam citra digital tersebut

.Berdasarkan berbagai pertimbangan tersebut maka dalam penyusunan skripsi, saya memilih judul **"Analisa Metode End Of File Dalam Penyisipan Pesan Teks Terenkripsi Dengan Hill Cipher Pada Gambar "**.

1.2 Perumusan Masalah

Dalam pelaksanaan penelitian skripsi ini terdapat beberapa permasalahan yang menjadi titik utama pembahasan, diantaranya adalah sebagai berikut:

a. Bagaimana cara kerja *End of File* (EOF) dalam menyisipkan pesan teks yang telah di enkripsi ?
b. Bagaimana menerapkan sistem keamanan data yang dapat melakukan proses enkripsi dan dekripsi suatu pesan teks dengan menggunakan metode *End of File* dan disisipkan ke dalam suatu *file* gambar berformat bitmap, Jpeg dll.
c. Bagaimana membuat aplikasi yang dapat menyisipkan pesan teks dengan menggunakan bahasa pemrograman visual basic 2008 ?

1.3 Batasan Masalah

Dari rumusan masalah yang ada maka dapat diberi batasan-batasan sehingga pembahasannya lebih terarah. Adapun batasan-batasan masalah menjadi acuan dalam pengerjaan skripsi ini sebagai berikut:

a. Menguraikan penyandian penyisipan pesan teks dengan metode *End of File* sebagai layanan keamanan data.
b. Rancangan program metode *End of File* pada data penyisipan pesan teks menggunakan Program *Microsoft Visual Studio 2008.*

c. Jumlah maximal panjang *character* dalam *Identity* dan *sandi* adalah 13 *huruf*.

d. Data yang digunakan adalah data teks dan *file* bitmap.

1.4 Tujuan Penelitian

Sesuai uraian pada latar belakang di atas, maka yang menjadi tujuan skripsi ini adalah:

a. Memperoleh aplikasi yang menggabungkan algoritma kriptografi Hill Cipher dan teknik steganografi *End of File*.

b. Mengetahui proses penyisipan pesan dan pengekstrakan pesan pada suatu *file* berformat bitmap dengan menggunakan metode *End of File*.

c. Menjabarkan cara kerja metode *End of File* dalam memberi layanan kerahasiaan pengamanan *Sandi* pada data *login*.

1.5 Manfaat Penelitian

Manfaat yang bisa di ambil dari penelitian ini adalah :

a. Sebagai bahan rekomendasi bagi peneliti lain yang ingin merancang aplikasi kriptografi dan steganografi sejenis.

b. Menghasilkan referensi suatu metode untuk pengamanan data dalam melakukan proses penyisipan dan pengekstrakan suatu pesan rahasia pada *file* citra dengan menggunakan metode *End of File* (EOF).

c. Untuk membantu mahasiswa maupun masyarakat dalam pengamanan data.

1.6 Sistematika Penulisan

Adapun sistematika penulisan Skripsi ini terdiri dari lima bab yaitu:

BAB I : PENDAHULUAN

Bab ini berisi tentang latar belakang, Rumusan Masalah, Batasan Masalah, Tujuan Penelitian, Manfaat Penelitian, dan Sistematika Penulisan.

BAB II : LANDASAN TEORITIS

Bab ini menjelaskan Pengertian Kriptografi, Pengenalan kriptografi dan teori-teori yang dapat mendukung pembuatan dan penyelesaian Skripsi ini.

BAB III : ANALISA DAN PERANCANGAN

Pada bab ini menjelaskan metodologi penelitian dan secara umum tentang penganalisaan mengenai *End of File* (EOF) dalam merahasiakan data dan perancangan aplikasi perangkat lunak dalam pengamanan data login pada basis data.

BAB IV : IMPLEMENTASI DAN PENGUJIAN

Bab ini akan dipaparkan sistem yang dibuat yang meliputi cara untuk menjalankan sistem dan hasil implementasi dari perancangan sistem pengamanan data login pada basis data dengan *End of File* (EOF).

BAB V : KESIMPULAN DAN SARAN

Bab ini berisikan tentang kesimpulan dan saran tentang hasil akhir dari pemecahan masalah yang telah diselesaikan atau terpecahkan.

BAB II
LANDASAN TEORITIS

A. Aplikasi / Perangkat Lunak

Perangkat lunak adalah sering disebut ‘program produktivitas’ atau ‘program end-user’. Dibandingkan dengan perangkat lunak sistem mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja dan pemutar media.

Berbagai aplikasi yang digabungkan bersama menjadi suatu paket kadang disebut sebagai suatu paket atau paket aplikasi (application package). Contohnya adalah Microsoft Office dan Adobe Master Collection, yang menggabungkan suatu aplikasi pengolah kata, lembar kerja serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi.

B. *Steganography*

Steganografi adalah seni menyembunyikan pesan teks sedemikian rupa sehingga tak seorang pun selain pengirim dan penerima yang dituju, mengetahui keberadaan informasi tersebut. Kata steganografi berasal dari bahasa Yunani yang berarti “tulisan tersembunyi atau terselubung”. Steganografi membutuhkan dua

properti yaitu wadah penampung dan pesan rahasia yang akan disembunyikan. Steganografi digital menggunakan media digital sebagai wadah penampung, misalnya citra, suara, teks dan video. Data rahasia yang disembunnyikan juga dapat berupa citra, suara, teks atau video. Steganografi dapat dipandang sebagai kelanjutan kriptografi, jika pada kriptografi, data yang telah disandikan (*chipertext*) tetap tersedia, maka dengan steganografi chiperteks dapat disembunyikan sehingga pihak ketiga tidak mengetahui keberadaannya. Di negara-negara yang melakukan penyensoran informasi, steganografi sering digunakan untuk menyembunyikan pesan-pesan melalui gambar (*image*), video, atau suara (*audio*).

1. Sejarah *Steganography*

Steganografi sudah digunakan sejak dahulu kala untuk kepentingan politik, militer, diplomatik, serta untuk kepentingan pribadi sebagai alat. Beberapa contoh penggunaan steganografi :

a. Steganografi pada tubuh budak : Pada jaman Yunani kuno, Herodatus (sejarahwan Yunani) mengirim pesan rahasia dengan menggunakan kepala budak atau prajurit sebagai media menyampaikan pesan dengan cara mencukur kepala budak dan mentato kepalanya dengan pesan tersebut. Kemudian saat rambutnya tumbuh kembali, budak dikirimkan pada tempat tujuan dan saat rambutnya kembali digunduli maka pesan akan terbaca.

b. Steganografi dalam lapisan lilin : pada jaman Yunani kuno, tulisan/ pesan rahasia di ukir pada kayu kemudian diberi lapisan lilin untuk menutupi pesan/

tulisan tersebut, dengan begitu tulisan/ pesan dapat disampaikan tanpa menimbulkan kecurigaan.

c. Steganografi pada kertas : Pada Perang Dunia II, pemerintah Amerika Serikat menulis pesan rahasia pada tentaranya yang ditawan oleh Jerman dengan menggunakan tinta tak terlihat di atas atau sebagian kosong pesan lainnya dan untuk mengetahuainya dengan digunakan air.

d. Steganografi dengan microdots : Masih pada Perang Dunia II agen mata-mata dengan menggunakan microdots untuk mengirim informasi. Penggunaan teknik ini biasa digunakan pada microfilm chip yang harus diperbesar sekitar 200 kali.

e. Steganografi dunia digital : menyembunyikan pesan-pesan kegiatan terornya dalam berbagai media yang dapat dijadikan penampung untuk menyembunyikan file seperti pada image, audio dan video Pada suatu pengarahan yang dilakukan FBI pada akhir September 2001, asisten direktur FBI, Ron Dick menyatakan kemungkinan para pembajak pesawat pada serangan 11 september ke gedung World Trade Center menggunakan teknologi internet.

2. *Steganalysis*

Steganalysis merupakan suatu teknik atau yang digunakan untuk mendeteksi pesan yang disembnyikan atau tersamar dari steganografi. Steganalysis menjadi suatu misteri tersendiri untuk dapat diketahui bagaimana teknik untuk melakukan proses deskripsi atau pemecahan atau penemuan pesan

tersebut. Terdapat beberapa software yang dapat melakukan analisis adanya penggunakan teknik steganografi. Dalam prakteknya cara pemecahan teknik apa yang digunakan dalam steganalysis sendiri secara *empiric* berkisar diantara :

a. Menyelidiki dari perubahan yang dilakukan terhadap meta data file tersebut.
b. Menyelidiki dari ciri-ciri file telah menggunakan software tertentu untuk steganografi.
c. Mengindentifikasi file asli, lalu dicari perbedaan pola yang digunakan sehingga dengan cara ini bukan saja dapat diketahui file telah mengalami proses steganografi dapat pula diketahui pesan yang tersembunyi.

Teknik steganalyisis tidak bisa mengetahui pesan yang disembunyikan bila ternyata pesan tersebut di *cryptography* atau pengkodean pesan lagi. Jadi cara yang lebih baik melakukan steganografi adalah dengan menggunakan opini bahwa orang akan tahu ada pesan yang tersembunyi sehingga dilakukan pengamanan dengan *chryptography*.

3. Kriteria *Steganography*

Menyembunyikan data rahasia ke dalam citra digital akan mengubah kualitas citra tersebut, kriteria yang harus diperhatikan dalam penyembunyian data adalah:

a. Fidelity, kwalitas citra penampung tidak jauh berubah. Setelah penambahan data/ pesan rahasia, citra hasil steganografi masih terlihat dengan baik. Pengamat tidak mengtahui kalau didalam citra tersebut terdapat data/ pesan rahasia.

b. Robustness, data/ pesan yang akan disembunyikan harus tahan terhadap manipulasi yang akan dilakukan pada citra penampung (seperti perubahan kontras, penajaman, pemampatan, rotasi, perbesar gambar, pemotongan (*cropping*), enkripsi dan sebagainya). Bila pada citra dilakukan operasi pengolahan citra, maka data yang disembunyikan tidak rusak.

c. Recovery, data yang disembunyikan harus dapat diungkapkan kembali (*recovery*).

Karena tujuan steganografi adalah data hiding, maka sewaktu-waktu data rahasia didalam citra penampung harus dapat diambil kembali untuk digunakan lebih lanjut.

C. Metode *EOF (End Of File)*

Metode EOF (*End Of File*) merupakan salah satu teknik yang menyisipkan data pada akhir file. Teknik ini dapat digunakan untuk menyisipkan data yang ukurannya sama dengan ukuran file sebelum disisipkan data ditambah dengan ukuran data yang disisipkan kedalam file tersebut. Dalam teknik EOF, data yang disisipkan diakhir file diberi tanda khusus sebagai pengenal start dari data tersebut dan pengenal akhir dari data tersebut.

Metode EOF (*End Of File*) merupakan sebuah metode yang diadaptasi dari metode penanda akhir file (*End Of File*) yang digunakan oleh sistem operasi windows. Dalam sistem operasi windows, jika ditemukan penanda EOF pada sebuah akhir file, maka sistem akan berhenti melakukan pembacaan pada file

tersebut. Prinsip kerja EOF menggunakan karakter / symbol khusus sebagai tanda yang diberikan pada setiap akhir file.

EOF menggunakan karakter yang berbeda sebagai penanda awal penyisipan pesan dan penanda akhir penyisipan pesan. Metode EOF menggunakan kelemahan indera manusia yang tidak sensitif sehingga seakan-akan tidak ada perbedaan yang terlihat antara sebelum atau sesudah pesan disisipkan.

Sebagai contoh, akan disisipkan sebuah pesan berjumlah 150 karakter pada sebuah citra digital dengan dimensi 300 x 300 pixel. Maka pesan akan ditempatkan pada baris ke 301 sampai selesai sesuai dengan panjang dari pesan yang disisipkan, Berikut gambar citra yang disisipi pesan dengan citra yang tidak disisipi pesan.

Gambar 2.1 Citra yang tidak disisipi pesan

Gambar 2.2 Citra yang disisipi pesan

Adapun algoritma dari metode EOF (End Of File) ini adalah sebagai berikut:

a. Baca informasi file, tentukan dimana posisi akhir file.

b. Tandai posisi ctrl-z(penanda) sebagai awal baris penyisipan pesan.

c. Sisipkan pesan dimulai dari posisi ctrl-z(penanda) hingga akhir pesan.

d. Sisipkan ctrl-z(penanda) kedua pada akhir pesan.

Teknik EOF tidak mengubah isi awal dari file yang disisipi. Sebagai contoh, jika kita menyisipkan sebuah pesan kedalam sebuah dokumen, isi dari document tidak akan berubah. Ini yang menjadi salah satu keunggulan metode EOF dibandingkan dengan metode steganografi yang lain. Karena disisipkan pada akhir file, pesan yang disisipkan tidak akan bersinggungan dengan isi file, hal ini menyebabkan integrasi data dari file yang disisipi tetap terjaga. Namun, metode EOF tidak dapat menyisipkan pesan berukuran sangat besar karena dapat

membuat citra berubah dan mencurigakan, baiknya pesan tidak terlalu besar agar tidak mencurigakan.

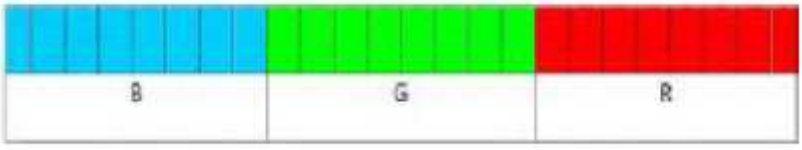
D. Citra Digital

Citra adalah gambar pada bidang dua dimensi. Dalam tinjauan matematis, citra merupakan fungsi kontinu dari intensitas cahaya pada bidang dua dimensi. Ketika sumber cahaya menerangi objek, objek memantulkan kembali sebagian cahaya tersebut. Pantulan ini ditangkap oleh alat-alat pengindera optik, misalnya mata manusia, kamera, scanner dan sebagainya. Bayangan objek tersebut akan terekam sesuai intensitas pantulan cahaya. Ketika alat optik yang merekam pantulan cahaya itu merupakan mesin digital, misalnya kamera digital, maka citra yang dihasilkan merupakan citra digital. Pada citra digital, kontinuitas intensitas cahaya dikuantisasi sesuai resolusi alat perekam.

1. Citra RGB

Suatu citra biasanya mengacu ke citra RGB. Sebenarnya bagaimana citra disimpan dan dimanipulasi dalam komputer diturunkan dari teknologi televisi, yang pertama kali mengaplikasikannya untuk tampilan grafis komputer. Jika dilihat dengan kaca pembesar, tampilan monitor komputer akan terdiri dari sejumlah triplet titik warna merah (*RED*), hijau (*GREEN*) dan biru (*BLUE*). Tergantung pada pabrik monitornya untuk menentukan apakah titik tersebut merupakan titik bulat atau kotak kecil, tetapi akan selalu terdiri dari 3 triplet *red*,*green* dan *blue*. Citra dalam komputer tidak lebih dari sekumpulan sejumlah triplet dimana setiap triplet terdiri atas variasi tingkat keterangan (brightness) dari elemen *red*, *green* dan *blue*. Representasinya dalam citra, triplet akan terdiri dari 3

angka yang mengatur intensitas dari *Red* (R), *Green* (G) dan *Blue* (Blue) dari suatu triplet. Setiap triplet akan merepresentasikan 1 pixel (*picture element*). Suatu triplet dengan nilai 67, 228 dan 180 berarti akan mengeset nilai R ke nilai 67, G ke nilai 228 dan B k nilai 180. Angka-angka RGB ini yang seringkali disebut dengan *color values.* Pada format .bmp citra setiap pixel pada citra direpresentasikan dengan dengan 24 bit, 8 bit untuk R, 8 bit untuk G dan 8 bit untuk B, dengan pengaturan seperti pada gambar di bawah ini.



Gambar 2.3 Warna RGB

2. Citra Grayscale

Dalam komputasi, suatu citra digital grayscale atau greyscale adalah suatu citra dimana nilai dari setiap pixel merupakan sample tunggal. Citra yang ditampilkan dari citra jenis ini terdiri atas warna abu-abu, bervariasi pada warna hitam pada bagian yang intensitas terlemah dan warna putih pada intensitas terkuat.

Citra grayscale berbeda dengan citra ”hitam-putih”, dimana pada konteks komputer, citra hitam putih hanya terdiri atas 2 warna saja yaitu ”hitam” dan ”putih” saja. Pada citra grayscale warna bervariasi antara hitam dan putih, tetapi variasi warna diantaranya sangat banyak. Citra grayscale seringkali merupakan perhitungan dari intensitas cahaya pada setiap pixel pada spektrum

elektromagnetik *single band* .Citra grayscale disimpan dalam format 8 bit untuk setiap sample pixel, yangmemungkinkan sebanyak 256 intensitas. Format ini sangat membantu dalam pemrograman karena manupulasi bit yang tidak terlalu banyak. Pada aplikasi lain seperti pada aplikasi *medical imaging* dan *remote sensing* biasa juga digunakan format 10,12 maupun 16 bit.

3. Karakteristik Citra Digital

Setiap citra digital memiliki beberapa karakteristik, antara lain yaitu :

a. Ukuran Citra Digital

Ukuran citra *(image size*) menyatakan ukuran banyaknya pixel penyusun citra raster yang dinyatakan dalam matrik 2 dimensi, yaitu (X,Y) pixel, dimana X menyatakan ukuran banyaknya pixel perbaris pada arah secara horizontal sedangkan Y menyatakan ukuran banyaknya pixel perkolom pada arah secara vertical. Sebagai contoh, citra digital berukuran 8000 x 6000 piksel, terdiri dari 8000x6000 pixel = 48.000.000 piksel, dengan susunan 8000 piksel setiap baris pada arah horizontal dan 6000 piksel setiap kolom pada arah vertical.

b. Resolusi

Atribut citra digital yang tak kalah pentingnya adalah resolusi (*resolution*), yang didefinisikan sebagai banyaknya piksel dalam setiap satuan panjang. Umumnya, resolusi dinyatakan dalam satuan dpi (*dot per inchi*). Sebagai contoh, citra digital memiliki resolusi 72dpi, berarti terdiri dari 72 dot(titik) pada setiap inchi. Semakin tinggi resolusi suatu citra digital, maka kualitasnya akan semakin baik.

4. Format Citra Digital

Format citra digital menetukan bagaimana informasi data dipresentasikan dalam suatu citra digital. Informasi tersebut meliputi ada tidaknya kompresi, program, aplikasi *(feature)* yang di support, penggunaan enkripsi dan lain-lain. Tiap format citra digital memiliki kelebihan dan kelemahan pada masing-masing format tersebut. Dalam sistem operasi windows biasanya format citra digital dapat dibedakan dari namanya yaitu diakhiri titik atau *extension* (contoh .png).

5. PNG(*Portable Network Graphics*)

(*Portable Network Graphics*) adalah salah satu format penyimpanan citra yang menggunakan metode pemadatan yang tidak menghilangkan bagian dari citra tersebut (*lossless compression*). PNG dibaca "ping", namun biasanya dieja apa adanya - untuk menghindari kerancuan dengan istilah "ping" pada jaringan komputer.

Format PNG ini diperkenalkan untuk menggantikan format penyimpanan citra GIF. Secara umum PNG dipakai untuk Citra Web (World Wide Web). Untuk Web, format PNG mempunyai 3 keuntungan dibandingkan format GIF:

1. Channel Alpha (transparansi)
2. Gamma (pengaturan terang-gelapnya citra en:"brightness")
3. Penayangan citra secara progresif (progressive display)

Selain itu, citra dengan format PNG mempunyai faktor kompresi yang lebih baik dibandingkan dengan GIF (5%-25% lebih baik dibanding format GIF). Satu fasilitas dari GIF yang tidak terdapat pada PNG format adalah dukungan terhadap penyimpanan multi-citra untuk keperluan animasi.

Untuk keperluan pengolahan citra, meskipun format PNG bisa dijadikan alternatif selama proses pengolahan citra - karena format ini selain tidak menghilangkan bagian dari citra yang sedang diolah (sehingga penyimpanan berulang ulang dari citra tidak akan menurunkan kualitas citra) namun format JPEG masih menjadi pilihan yang lebih baik.

E. Keamanan Komputer

Keamanan komputer adalah berhubungan dengan pencegahan diri dan deteksi terhadap tindakan pengganggu yang tidak dikenali dalam sistem komputer. Dalam keamanan sistem komputer yang perlu kita lakukan adalah untuk mempersulit orang lain untuk mengganggu sistem yang kita pakai, baik itu kita menggunakan komputer yang sifatnya *stand alone*, jaringan *local* maupun global. Kita harus memastikan sistem bisa berjalan dengan baik dan kondusif, selain itu program aplikasinya masih bisa dipakai tanpa masalah. Dony Ariyus (2006).

1. Beberapa hal yang menjadikan kejahatan komputer terus terjadi dan cenderung meningkat adalah sebagai berikut:
   a. Meningkatnya pengguna komputer dan *internet*.
   b. Banyaknya *software* yang pada awalnya digunakan untuk melakukan *audit* sebuah sistem dengan cara mencari kelemahan dan celah yang mungkin ada disalah gunakan untuk melakukan *scanning* sistem orang lain.
   c. Banyaknya *software-software* untuk melakukan penyusupan yang tersedia di internet yang bisa di *download* secara gratis.
   d. Meningkatnya kemampuan pengguna komputer dan *internet*.

e. Semakin banyaknya perusahaan yang menghubungkan jaringan LAN mereka ke Internet.
f. Meningkatnya aplikasi bisnis yang menggunakan internet.
g. Banyaknya software yang mempunyai kelemahan (*bugs*).

2. Ada beberapa hal yang bisa menjawab pertanyaan mengapa kita perlu mengamankan sistem computer, antara lain:
   a. Menghindari resiko penyusupan, kita harus memastikan bahwa sistem tidak kemasukaan penyusup yang bisa membaca, menulis dan menjalankan program-program yang bisa mengganggu atau menghancurkan sistem kita.
   b. Mengurangi resiko ancaman, hal ini biasa berlaku di institusi dan perusahaan swasta.
   c. Melindungi sistem dari kerentanan, kerentanan akan menjadikan sistem kita berpotensi untuk memberikan akses yang tidak diizinkan bagi orang lain yang tidak berhak.
   d. Melindungi sistem dari gangguan alam seperti petir dan lain-lainnya.

1. Aspek-Aspek Keamanan Komputer

Inti dari keamanan komputer adalah melindungi komputer dan jaringannya dengan tujuan mengamankan informasi yang berada di dalamnya. Dony Ariyus (2006), Keamanan komputer sendiri meliputi beberapa aspek , antara lain:

a. *Privacy*, adalah sesuatu yang bersifat rahasia (*private*). Intinya adalah pencegahan agar informasi tersebut tidak diakses oleh orang yang tidak berhak. Contohnya adalah *email* atau *file-file* lain yang tidak boleh dibaca orang lain meskipun oleh *administrator*. Pencegahan yang mungkin dilakukan

adalah dengan menggunakan teknologi *enkripsi*, jadi hanya pemilik informasi yang dapat mengetahui informasi yang sesungguhnya.

b. *Confidentiality*, merupakan data yang diberikan ke pihak lain untuk tujuan khusus tetapi tetap dijaga penyebarannya. Contohnya data yang bersifat pribadi seperti: nama, alamat, no ktp, telpon dan sebagainya. *Confidentiality* akan terlihat apabila diminta untuk membuktikan kejahatan seseorang, apakah pemegang informasi akan memberikan infomasinya kepada orang yang memintanya atau menjaga klientnya.

c. *Integrity*, penekanannya adalah sebuah informasi tidak boleh diubah kecuali oleh pemilik informasi. Terkadang data yang telah terenkripsipun tidak terjaga integritasnya karena ada kemungkinan *ciphertext* dari enkripsi tersebut berubah. Contoh: Penyerangan Integritas ketika sebuah *email* dikirimkan ditengah jalan disadap dan diganti isinya, sehingga *email* yang sampai ketujuan sudah berubah.

d. *Autentication,* ini akan dilakukan sewaktu *user login* dengan menggunakan nama *user* dan *password*-nya, apakah cocok atau tidak, jika cocok diterima dan tidak akan ditolak. Ini biasanya berhubungan dengan hak akses seseorang, apakah dia pengakses yang sah atau tidak.

e. *Availability*, aspek ini berkaitan dengan apakah sebuah data tersedia saat dibutuhkan/diperlukan. Apabila sebuah data atau informasi terlalu ketat pengamanannya akan menyulitkan dalam akses data tersebut. Disamping itu akses yang lambat juga menghambat terpenuhnya aspek *availability*. Serangan yang sering dilakukan pada aspek ini adalah *denial of service* (DOS), yaitu

penggagalan *service* sewaktu adanya permintaan data sehingga komputer tidak bisa melayaninya. Contoh lain dari *denial of service* ini adalah mengirimkan *request* yang berlebihan sehingga menyebabkan komputer tidak bisa lagi menampung beban tersebut dan akhirnya komputer *down*.

2. Ancaman Keamanan

Ada begitu banyak peristiwa pertukaran informasi setiap detik di internet. Pertukaran informasi tersebut tentu tak lepas dari terjadinya pencurian informasi oleh pihak-pihak yang tidak bertanggung jawab. Beberapa ancaman keamanan terhadap informasi adalah:

1. *Interruption*: Merupakan suatu ancaman terhadap ketersediaan informasi; data yang berada dalam sistem computer dirusak atau dihapus sehingga saat diperlukan, data atau informasi tersebut sudah tidak ada lagi.
2. *Interception*: Merupakan ancaman terhadap kerahasiaan (*secrecy*). Informasi yang ada disadap atau orang yang tidak berhak bisa mengakses computer tempat informasi tersebut disimpan.
3. *Modifikasi*: Merupakan ancaman terhadap integritas. Orang yang tidak berhak berhasil menyadap lalu lintas pengiriman informasi, lalu mengubahnya sesuai keinginan orang tersebut.
4. *Fabrication*: Merupakan ancaman terhadap integritas. Orang yang tidak berhak berhasil meniru (memalsukan) suatu informasi yang ada sehingga orang yang menerima informasi tersebut menyangka informasi tersebut berasal dari orang yang dikehendaki oleh si penerima informasi tersebut (Ariyus, 2009).

F. Kriptografi

Kriptografi (cryptography) berasal dari Bahasa Yunani: “cryptos” artinya “scret” (rahasia), sedangkan “graphein” artinya “writing” (tulisan). Jadi, kriptografi berarti “secret writing” (tulisan rahasia). Menurut terminologinya kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan ketika dikirim dari suatu tempat ke tempat yang lain. Dalam perkembangannya, kriptografi juga digunakan untuk mengidentifikasikan pengiriman pesan dengan tanda tangan digital dan keaslian pesan dengan sidik jari digital (*fingerprint*) (Ariyus, 2006).

1. Komponen Kriptografi

Menurut (Ariyus, 2009), pada dasarnya, kriptografi terdiri dari beberapa komponen seperti:

a. *Enkripsi :* Enkripsi merupakan bagian dari kriptografi, dan merupakan hal yang sangat penting supaya keamanan data yang dikirimkan bisa terjaga kerahasiaannya. Enkripsi bisa diartikan dengan chiper atau kode, dimana pesan asli (plaintext) diubah menjadi kode-kode tersendiri sesuai metode yang disepakati oleh kedua belah pihak, baik pihak pengirim pesan maupun penerima pesan.

b. *Dekripsi : K*ebalikan dari enkripsi yakni mengubah chipertext menjadi plaintext sehingga berupa data asli atau awal.

c. Kunci: Kunci yang dimaksud di sini adalah kunci yang dipakai untuk melakukan *enkripsi* dan *dekripsi*. Kunci terbagi menjadi dua bagian, yaitu kunci pribadi (*private key*) dan kunci umum (*public key*).

d. *Chipertext:* Merupakan suatu pesan yang sudah melalui proses *enkripsi*. Pesan yang ada pada *chipertext* tidak bisa dibaca karena berisi karakter-karakter yang tidak memiliki makna (arti).
e. *Plaintext*: Sering juga disebut *cleartext*; merupakan suatu pesan bermakna yang ditulis atau diketik dan *plaintext* itulah yang akan diproses menggunakan algoritma *cryptography* agar menjadi *chipertext*.
f. Pesan: Pesan bisa berupa data atau informasi yang dikirim (melalui kurir, saluran komunikasi data, dsb) atau yang disimpan di dalam media perekaman (kertas, *storage*, dsb).
g. *Cryptanalysis*: Bisa diartikan sebagai analisis sandi atau suatu ilmu untuk mendapatkan *plaintext* tanpa harus mengetahui kunci secara wajar. Jika suatu *chipertext* berhasil menjadi *plaintext* tanpa menggunakan kunci yang sah, maka proses tersebut dinamakan *breaking code* yang dilakukan oleh para *cryptanalys*. Analisis sandi juga mampu menemukan kelemahan dari suatu algoritma *cryptography* dan akhirnya bisa menemukan kunci atau *plaintext* dari *chipertext* yang di *enkripsi* menggunakan algoritma tertentu (Ariyus, 2009).

2. Tujuan Kriptografi

Dari paparan awal dapat dirangkumkan bahwa kriptografi bertujuan untuk member layanan keamanan yaitu sebagai berikut :

a. Kerahasiaan (confidentiality) Adalah layanan yang ditujukan untuk menjaga agar pesan tidak dapat dibaca oleh pihak-pihak yang tidak berhak.

b. Integritas data (data integrity) Adalah layanan yang menjamin bahwa pesan masih asli atau belum pernah dimanipulasi selama pengiriman.
c. Otentikasi (authentication) Adalah layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (user autehentication).
d. Non-repudiation Adalah layanan untuk menjaga entitas yang berkomunikasi melakukan penyangkalan.

3. Serangan Terhadap Kriptografi

Di bawah ini dijelaskan beberapa macam penyerangan terhadap pesan yang sudah dienkripsi, berdasarkan ketersediaan data yang ada, dan tingkat kesulitannya bagi penyerang, dimulai dari yang paling sulit adalah :

a. *Ciphertext only attack*, penyerang hanya mendapatkan *ciphertext* dari sejumlah pesan yang seluruhnya telah dienkripsi menggunakan algoritma yang sama. Sehingga, metode yang digunakan untukmemecahkannya adalah *exhaustive key search,* yaitu mencoba semua kemungkinan yang ada untuk menemukan kunci.
b. *Known plaintext attack,* dimana penyerang selain mendapatkan sandi, juga mendapatkan pesan asli. Terkadang disebut pula *clear-text attack.*
c. *Choosen plaintext attack*, sama dengan *known plaintext attack*, namun penyerang bahkan dapat memilih penggalan mana dari pesan asli yang akan disandikan. Serangan jenis ini lebih hebat daripada *known-plaintex*t attack,

karena kriptoanalis dapat memilih *plainteks* tertentu untuk dienkripsikan, yaitu *plainteks-plainteks* yang lebih mengarahkan penemuan kunci

d. *Chosen ciphertext attack.* Pada tipe ini, kriptoanalis dapat memilih *cipherteks* yang berbeda untuk didekripsi dan memiliki akses atas *plaintext* yang didekripsi.
e. *Chosen key attack.* Kriptoanalis pada tipe penyerangan ini memiliki pengetahuan tentang hubungan antara kunci-kunci yang berbeda dan memilih kunci yang tepat untuk mendekripsi pesan.
f. *Rubber hose cryptanalysis*. Pada tipe penyerangan ini, kriptoanalis mengancam, menyiksa, memeras, memaksa, atau bahkan menyogok seseorang hingga mereka memberikan kuncinya. Ini adalah cara yang paling ampuh untuk mendapatkan kunci.
g. *Adaptive chosen plaintext attack,* Penyerangan tipe ini merupakan suatu kasus khusus *chosen-plaintext attack*. Kriptoanalis tidak hanya dapat memilih *plainteks* yang dienkripsi, ia pun memiliki kemampuan untuk memodifikasi pilihan berdasarkan hasil enkripsi sebelumnya. Dalam *chosen-plaintext attack*, kriptoanalis mungkin hanya dapat memiliki *plainteks* dalam suatu blok besar untuk dienkripsi; dalam *adaptive-chosen-plaintext attack* ini ia dapat memilih blok *plainteks* yang lebih kecil dan kemudian memilih yang lain berdasarkan hasil yang pertama, proses ini dapat dilakukannya terus menerus hingga ia dapat memperoleh seluruh informasi. (Ariyus, 2009).

Berdasarkan bagaimana cara dan posisi seseorang mendapatkan pesan-pesan dalam saluran komunikasi, penyerangan dapat dikategorikan menjadi:

1. *Spoofing,* Penyerang misalnya Angga bisa menyamar menjadi Adi. Semua orang dibuat percaya bahwa Angga adalah Adi. Penyerang berusaha meyakinkan pihak-pihak lain bahwa tak ada salah dengan komunikasi yang dilakukan, padahal komunikasi itu dilakukan dengan sang penipu/penyerang. Contohnya jika orang memasukkan PIN ke dalam mesin ATM palsu yang benar-benar dibuat seperti ATM asli tentu sang penipu bisa mendapatkan PIN-nya dan copy pita magentik kartu ATM milik sang nasabah. Pihak bank tidak tahu bahwa telah terjadi kejahatan.
2. *Man in the middle*, Jika *spoofing* terkadang hanya menipu satu pihak, maka dalam skenario ini, saat Adi hendak berkomunikasi dengan Badu, Angga di mata Adi seolah-olah adalah Badu, dan Angga dapat pula menipu Badu sehingga Angga seolah-olah adalah Adi. Angga dapat berkuasa penuh atas jalur komunikas ini, dan bisa membuat berita fitnah.
3. *Sniffing*, secara harfiah berarti mengendus, tentunya dalam hal ini yang diendus adalah pesan (baik yang belum ataupun sudah dienkripsi) dalam suatu saluran komunikasi. Hal ini umum terjadi pada saluran *public* yang tidak aman. Sang pengendus dapat merekam pembicaraan yang terjadi.
4. *Replay attack*, Jika seseorang bisa merekam pesan-pesan *handshake* (persiapan komunikasi), ia mungkin dapat mengulang pesan-pesan yang telah direkamnya untuk menipu salah satu pihak.

Beberapa metode penyadapan data yang biasanya dilakukan oleh penyerang:

1. *Electromagnetic eavesdropping,* Penyadap mencegat data yang ditransmisikan melalui saluran *wireless*, misalnya radio dan *microwive*.

2. *Acoustic Eavesdropping*, Menangkap gelombang suara yang dihasilkan oleh suara manusia.
3. *Wiretapping*, Penyadap mencegat data yang ditransmisikan pada saluran kabel komunikasi dengan menggunakan sambungan perangkat keras.

Jenis-jenis serangan berdasarkan teknik yang digunakan untuk menemukan kunci:

1. *Brute force attack* atau *Exhaustive attack*, Serangan *brute force* adalah sebuah teknik serangan yang menggunakan percobaan terhadap semua kunci yang mungkin untuk mengungkap *plainteks*/kunci.
2. *Analytical attack*, Pada jenis serangan ini, kriptanalis tidak mencoba-coba semua kemungkinan kunci tetapi menganalisis kelemahan algoritma kriptografi untuk mengurangi kemungkinan kunci yang tidak mungkin ada.

4. Kompleksitas Serangan

Kompleksitas serangan kriptografi dapat diukur dengan beberapa cara, yaitu:

a. Kompleksitas data (*data complexity*)

   Jumlah data (*plainteks* dan *cipherteks*) yang dibutuhkan sebagai masukan untuk serangan. Semakin banyak data yang dibutuhkan untuk melakukan serangan, semakin kompleks serangan tersebut, yang berarti semakin bagus sistem kriptografi tersebut.

b. Kompleksitas waktu (*time complexity*)

Waktu yang dibutuhkan untuk melakukan serangan. Semakin lama waktu yang dibutuhkan untuk melakukan serangan, berarti semakin bagus kriptografi tersebut.

c. Kompleksitas ruang memori (*space/storage complexity*)

Jumlah memori yang dibutuhkan untuk melakukan serangan. Semakin banyak memori yang dibutuhkan untuk melakukan serangan, berarti semakin bagus sistem kriptografi tersebut. (Ariyus, 2009).

5. Prinsip Menentukan Algoritma Kriptografi

Pengetahuan mengenai serangan terhadap kriptografi sangatlah penting untuk meningkatkan efektifitas dan kualitas algoritma penyandian yang digunakan. Prinsip yang dipakai dalam menentukan penggunaan suatu algoritma kriptografi adalah :

a. Persamaan matematis yang menggambarkan operasi algoritma kriptografi yang dibuat sangat kompleks sehingga algoritma tidak mungkin dipecahkan secara analitik.
b. Biaya untuk memecahkan *ciphertext* melampaui nilai informasi yang terkandung di dalam *ciphertext* tersebut.
c. Waktu yang diperlukan untuk memecahkan *ciphertext* tersebut melampaui lamanya waktu informasi tersebut harus dijaga kerahasiaannya.

6. Algoritma Kriptografi Klasik

Sebelum komputer ada, kriptografi dilakukan dengan menggunakan pensil dan kertas. Algoritma kriptografi (*cipher*) yang digunakan saat itu dinamakan juga algoritma klasik, adalah berbasis karakter, yaitu enkripsi dan dekripsi dilakukan

pada setiap karakter pesan. Rrinaldi Munir, 2010). Semua algoritma klasik termasuk ke dalam sistem kriptografi simetris dan digunakan jauh sebelum kriptografi kunci *public* ditemukan. Kriptografi klasik memiliki beberapa ciri:

a. Berbasis karakter
b. Menggunakan pena dan kertas saja, belum ada komputer
c. Termasuk ke dalam kriptografi kunci simetris

Pada dasarnya, algoritma kriptografi klasik dapat dikelompokkan ke dalam dua macam *cipher*, yaitu:

1. *Cipher* substitusi (*substitution cipher*)

   Di dalam *cipher* substitusi setiap unit *plainteks* diganti dengan satu unit *cipherteks*. Satu "unit" di sini berarti satu huruf, pasangan huruf, atau dikelompokkan lebih dari dua huruf. Algoritma substitusi tertua yang diketahui adalah *Caesar cipher* yang digunakan oleh kaisar Romawi, *Julius Caesar* (sehingga dinamakan juga *Caesar cipher*), untuk pesan yang dikirimkan kepada gubernurnya.

   Caesar cipher mudah dipecahkan dengan *exhaustive key* karena jumlah kuncinya sangat sedikit (hanya ada 26 kunci). *Caesar cipher* termasuk *cipher* abjad tunggal.

2. Cipher transposisi (*transposition cipher*)

   Pada cipher transposisi, huruf-huruf di dalam *plainteks* tetap saja, hanya saja urutannya diubah. Dengan kata lain algoritma ini melakukan *transpose* terhadap rangkaian karakter di dalam teks. Nama lain untuk metode ini adalah

permutasi atau pengacakan (*scrambling*), karena *transpose* setiap karakter di dalam teks sama dengan mempermutasikan karakter-karakter tersebut.

G. Algoritma Kriptografi Modern

Algoritma kriptografi modern umumnya beroperasi dalam mode bit ketimbang mode karakter (seperti yang dilakukan pada *cipher* substitusi atau *cipher* transposisi dari algoritma kriptografi klasik).

Operasi dalam mode bit berarti semua data dan informasi (baik kunci, *plainteks*, maupun *cipherteks*) dinyatakan dalam rangkaian (*string*) bit biner, 0 dan 1.

Algoritma enkripsi dan dekripsi memproses semua data dan informasi dalam bentuk rangkaian bit. Rangkaian bit yang menyatakan *plainteks* dienkripsi menjadi *cipherteks* dalam bentuk rangkaian bit, demikian sebaliknya.

Perkembangan algoritma kriptografi modern berbasis bit didorong oleh pengguna komputer digital yang merepresentasikan data dalam bentuk biner.

Algoritma kriptografi yang beroperasi dalam mode bit dapat dikelompokkan menjadi dua katagori:

a. *Cipher* aliran (*stream cipher*)

   Algoritma kriptografi beroperasi pada *plainteks/cipherteks* dalam bentuk bit tunggal, yangdalam hal ini rangkaian bit dienkripsikan/didekripsikan bit per bit.

b. *Cipher* blok (*block cipher)*

Algoritma kriptografi beroperasi pada *plainteks/cipherteks* dalam bentuk blok bit, yang dalam hal ini rangkaian bit dibagi menjadi blok-blok bit yang panjangnya sudah ditentukan sebelumnya.

H. Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah sebuah bahasa untuk menentukan, visualisasi, mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak. *Artifact* dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya.

UML merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan sistem yang besar dan kompleks. UML tidak hanya digunakan dalam proses pemodelan perangkat lunak, namun hamper dalam semua bidang yang membutuhkan pemodelan. Adapun bagian-bagian dari UML antara lain:

1. *View*, *View* digunakan untuk melihat sistem yang dimudelkan dari beberapa aspek.
2. *Uses case view*, Mendeskripsikan fungsionalitas sistem yang seharusnya dilakukan sesuai yang diinginkan *external actors*.
3. *Logical view*, Mendeskripsikan bagaimana fungsionalitas dari sistem, struktur statis (*class, object* dan *relationship*) dan kolaborasi dinamis yang terjadi ketika *object* mengirim pesan ke *object* lain dalam suatu fungsi tertentu.
4. *Component view*, Mendeskripsikan implementasi dan ketergantungan modul.
5. *Concurrency view*, Membagi sistem ke dalam proses dan prosesor.

6. *Deployment view*, Mendeskripsikan fisik dari sistem seperti komputer dan perangkat (*nodes*) dan bagaimana hubungannya dengan lainnya.
7. *Diagram, Diagram* berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem.

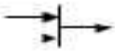
I. Activity Diagram

Activity Diagram merupakan *state* diagram khusus dimana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal proces*sing). Diagram aktivitas lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem itu dirakit. Diagram aktivitas menunjukan aktivitas sistem dalam bentuk kumpulan aksi-aksi.

Ketika digunakan dalam pemodelan software, diagram aktivitas merepsentasikan pemanggilan suatu fungsi tertentu misalnya *call*. Sedangkan bila digunakan dalam pemodelan bisnis, diagram ini menggambarkan aktivitas yang dipicu oleh kejadian-kejadian diluar, seperti pemasangan atau kejadian – kejadian internel. Sama seperti state, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behavior* dalam kondisi tertentu

Menurut Whitten et.al (2004) *diagram activity* digunakan untuk menggambarkan urutan aliran kegiatan-kegiatan dari sebuah proses bisnis atau sebuah *use case*.. Berikut tabel yang menyajikan notasi *Activity Diagram*.

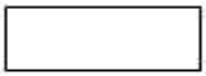
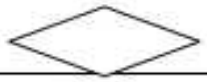
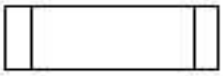
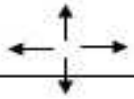
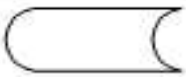
Tabel 2.1 Simbol Activity Diagram

No	Gambar	Nama	Keterangan
1		*Start Point*	Untuk mengawali suatu kegiatan.
2		*End Point*	Untuk mengahiri suatu kegiatan
3		aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kaata kerja.
4		*Fork*	Percabangan
5		*Join*	Penghubung
6		*Decision*	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
7		*Swimlane*	Sebuah cara untuk mengelompokan activity berdasarkan actor.

J. Flowchart

Flowchart adalah bagan (diagram) alir yang merupakan sekumpulan simbol-simbol atau skema yang menunjukkan kegiatan-kegiatan program dari awal sampai akhir. Diagram ini menggambarkan urutan-urutan atau langkah-langkah pengerjaan dari suatu algoritma. Berikut adalah penjelasan arti lambang-lambang *flowchart* atau diagram alir data.

Tabel 2.2 Simbol Flowchart

Simbol	Fungsi
	Terminal, untuk memulai atau mengakhiri program.
	Proses, suatu simbol yang menunjukkan setiap pengolahan yang dilakukan.
	*Input-output*, untuk memasukkan data ataupun menunjukkan hasil dari suatu program.
	*Decision*, suatu kondisi yang akan menghasilkan beberapa kemungkinan jawaban atau pilihan.
	*Predefined process*, proses suatu simbol untuk menyatakan sekumpulan langkah proses yang ditulis sebagai prosedur.
	*Conector*, suatu prosedur akan masuk atau keluar melalui simbol ini lember yang sama.
	*Oof-page connector*, merupakan simbol masuk atau keluar suatu prosedur pada kertas yang lain.
	Arus *flow* dari pada prosedur yang dapat dilakukan atas, bawah ke atas, kiri ke kanan, dan kanan ke kiri
	*Document*, merupakan simbol untuk data yang berbentuk kertas maupun untuk informasi.
	*Display*, simbol untuk *output*, ditunjukkan ke suatu *device* seperti *printer* dan sebagainya.
	Penyimpanan *file* secara sementara.

Sumber : http://id.wikipedia.org/wiki/Diagram_alir

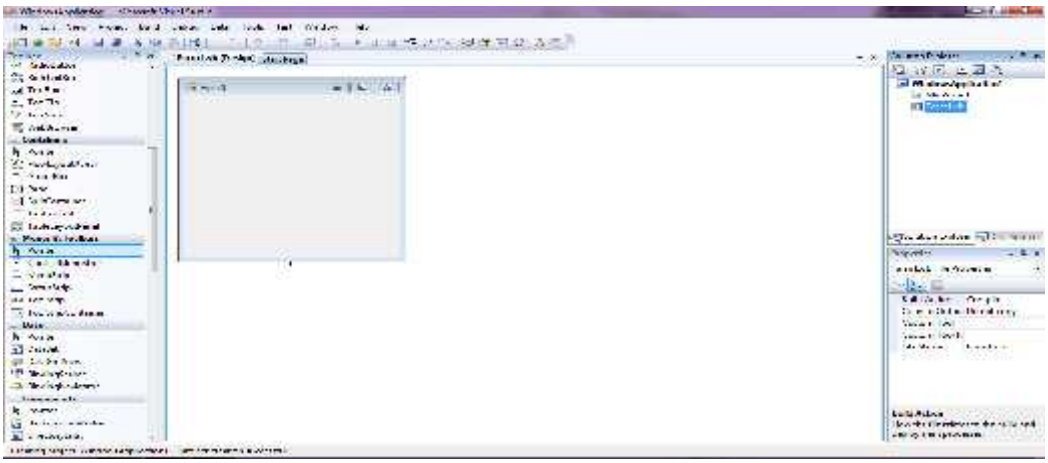
K. Konsep Dasar Bahasa Pemograman

Didalam konsep bahasa pemograman dapat didukung beberapa aplikasi untuk merancang sebuah program yaitu:

1. **Microsoft Visual Studio 2008**

Visual Studio 2008 adalah salah satu produk bahasa pemograman yang dikeluarkan oleh *Microsoft,* salah satu perusahaan *software* terkemuka di dunia. *Visual Studio 2008* merupakan bahasa pemograman yang mudah digunakan untuk mengembangkan aplikasi, baik itu aplikasi kecil maupun aplikasi besar.

Visual Studio 2008 banyak digunakan oleh para programmer dan pengembang aplikasi, karena kemudahan yang ditawarkan. Dalam pengembangan aplikasi para programmer tidak terlalu dipusingkan dengan tampilan dari program, karena *visual studio 2008* menyediakan banyak komponen control untuk desain tampil dari program. Dengan *Visual Studio 2008* dapat dikembangkan berbagai jenis aplikasi, seperti aplikasi *database*, jaringan, internet dan multimedia grafik.



Gambar 2.1 Tampilan Kerja Microsoft Visual Studio 2008

Sumber : Hendrayudi (2008)

BAB III

ANALISA DAN PERANCANGAN SISTEM

3.1 Metode Penelitian

Metodologi penelitian merupakan suatu teknik yang sistematik untuk mengerjakan suatu kasus. Langkah-langkah metodologi penelitian yang digunakan dalam pengerjaan Skripsi ini adalah sebagai berikut:

1. Studi *Literature*
   a. Mempelajari materi mata kuliah kriptografi dengan metode *End of File* (EOF).
   b. Mempelajari pemrograman *Microsoft Visual Studio 2008.*
2. Pengumpulan Data
   a. Mengambil informasi tentang kriptografi dengan algoritma *End of File* (EOF)
3. Melakukan analisa sistem yang digunakan dalam pembuatan aplikasi
   a. Perancangan sistem.
   b. Pengkodean.
   c. Impelentasi dan pengujian terhadap sistem yang dibangun dengan bahasa program *Microsoft Visual Studio 2008.*

3.2 Analisa Sistem

Penyandian dengan *Hill Cipher* dilakukan dengan memanfaatkan operasi matriks biasa. Penyandian dilakukan pada tiap blok teks yang berukuran sama dengan ordo matriks yang digunakan. Sebagai perluasannya, dalam tulisan ini *Hill Cipher* diimplementasikan untuk menyandikan sebuah citra. Ini dimungkinkan mengingat sebuah citra merupakan deretan pixel-pixel yang mempunyai komponen R (Red), G (Green), dan B (Blue). Komponen-komponen ini merupakan bilangan-bilangan bulat sehingga dapat dioperasikan dalam sebuah matriks. Citra yang digunakan dalam tulisan ini dibatasi dalam format BMP, GIF, JPG atau JPEG. Sedangkan matriks yang dipakai adalah matriks bujur sangkar atau persegi berordo 2 × 2 dan 3 × 3 dengan elemen bilangan bulat.

Implementasi akhir dari tulisan ini adalah perancangan suatu program yang dapat berfungsi untuk menyandikan atau mengekripsi citra sehingga tiap komponen warna pada citra tersebut menjadi tidak teratur dan citra menjadi sulit untuk dikenali.

3.3 Cara Kerja *Hill Cipher* Pada Penyandian Citra

Pada tulisan ini penggunaan *Hill Cipher* diperluas pemakaiannya pada citra baik untuk citra berwarna atau monokrom (hitam-putih). Karena tiap-tiap komponen RGB pixel memiliki panjang 8 bit (0-255), maka sistem modulo yang dipakai dalam penyandian adalah $Z_n = Z_{256}$. Program ini nantinya dibuat dengan bahasa pemrograman Visual Basic.

Untuk mengenkripsi citra dengan *Hill Cipher*, mula-mula nilai RGB tiap pixel diambil. Adapun algoritma untuk pengambilan nilai RGB yang dilakukan untuk tiap pixel dengan menggunakan fungsi GetPixel adalah sebagai berikut:

```
For i = 0 To Picture Width – 1
For j = 0 To Picture Heigth – 1
        'GetPixel i,j
         PixelColor = GetPixel(Picture, i,j)
    r  = PixelColor Mod 256
    g  = (PixelColor \ 256) Mod 256
    b  = PixelColor \ 256 \ 256
Next j
Next i
```

Dimana :

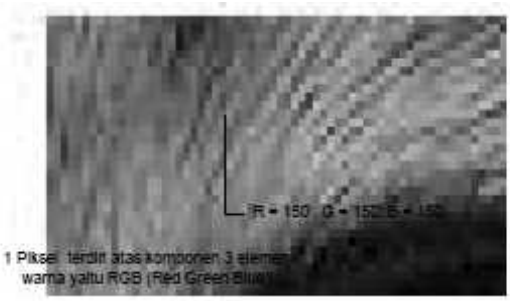
a. Picture Width = Lebar Citra dalam satuan piksel
b. Picture Height = Tinggi Citra dalam satuan piksel.
c. PixelColor = Merupakan hasil pengembalian warna dari fungsi GetPixel dalam nilai *long integer*.
d. r = Merupakan nilai merah (*red*) pada piksel dalam rentang 0 hingga 255.

a. g = Merupakan nilai warna hijau (*Green)* pada piksel dalam rentang 0 hingga 255.

b. b = Merupakan nilai warna biru (*blue*) pada piksel dalam rentang 0 hingga 255.

c. i dan j = Merupakan posisi piksel yang dinyatakan dalam sumbu koordinat.

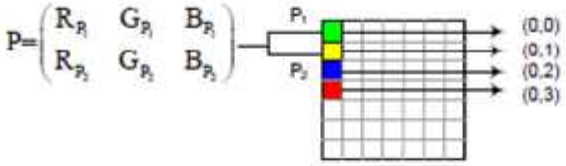
Seperti ketahui bahwa setiap piksel pasti terdiri atas nilai r, g, dan b. Ilustrasi mengenai ini dapat dilihat pada Gambar 3.1 Ilustrasi Nilai RGB pada Piksel



Gambar 3.1 Ilustrasi Nilai RGB pada Piksel

Jika cara di atas diterapkan maka suatu citra dapat dienkripsi dengan *Hill Cipher* seperti halnya dengan teks (ditransformasi dalam nilai ASCII) berdasarkan atas nilai komponen RGB-nya.Masalah yang dihadapi adalah untuk citra satu piksel warna terdapat tiga buah nilai yang harus dienkripsi. Maka nilai-nilai komponen warna tersebut harus disusun sedemikian rupa agar dapat dikalikan dengan matriks kunci karena perkalian matriks bersifat unik. Matriks tidak dapat dikalikan jika jumlah kolom pada matriks pertama tidak sama dengan jumlah baris pada matriks kedua. Berdasarkan algoritma untuk mengambil nilai piksel dan

mengubahnya menjadi komponen RGB maka dapat diketahui pembacaan dimulai secara per kolom. Artinya dimulai dari baris pertama hingga baris terakhir pada kolom pertama baru dilanjutkan pada kolom berikutnya.Jika matriks kunci yang dipakai berordo dua maka penyusunan matriks P (matriks yang berisi nilai RGB citra yang akan disandikan) adalah sebagai berikut:



Gambar 3.2 Representasi Nilai Piksel Untuk Matriks 2 × 2

Dimana:

R_{p1} adalah nilai komponen red pada pixel pertama

G_{p1} adalah nilai komponen green pada pixel pertama

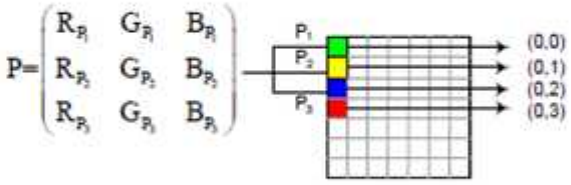
B_{p1} adalah nilai komponen blue pada pixel pertama

R_{p2} adalah nilai komponen red pada pixel kedua

G_{p2} adalah nilai komponen green pada pixel kedua

B_{p2} adalah nilai komponen blue pada pixel kedua

Jika matriks kunci yang dipakai berordo tiga maka penyusun matriks P adalah sebagai berikut



Gambar 3.3 Representasi Nilai Piksel Untuk Matriks 3 x 3

Dimana:

R_{p1} adalah nilai komponen *red* pada *pixel* pertama

G_{p1} adalah nilai komponen *green* pada *pixel* pertama

B_{p1} adalah nilai komponen *blue* pada *pixel* pertama

R_{p2} adalah nilai komponen *red* pada *pixel* kedua

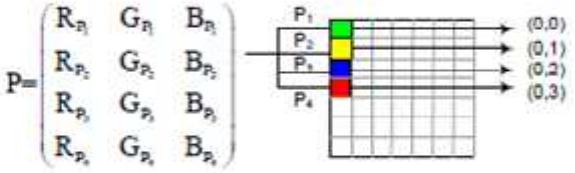
G_{p2} adalah nilai komponen *green* pada *pixel* kedua

B_{p2} adalah nilai komponen *blue* pada *pixel* kedua

R_{p3} adalah nilai komponen *red* pada *pixel* ketiga

G_{p3} adalah nilai komponen *green pada pixel* ketiga

B_{p3} adalah nilai komponen *blue* pada *pixel* ketiga



Gambar 3.4 Representasi Nilai Piksel Untuk Matriks 4 x 4

Dimana:

R_{P1} adalah nilai komponen *red* pada *pixel* pertama

G_{P1} adalah nilai komponen *green* pada *pixel* pertama

B_{p1} adalah nilai komponen *blue* pada *pixel* pertama

R_{P2} adalah nilai komponen *Red* pada *pixel* kedua

G_{P2} adalah nilai komponen *green* pada *pixel* kedua

B_{P2} adalah nilai komponen *blue* pada *pixel* kedua

R_{P3} adalah nilai komponen *red* pada *pixel* ketiga

G_{P3} adalah nilai komponen *green* pada *pixel* ketiga

B_{P4} adalah nilai komponen *blue* pada *pixel* ketiga

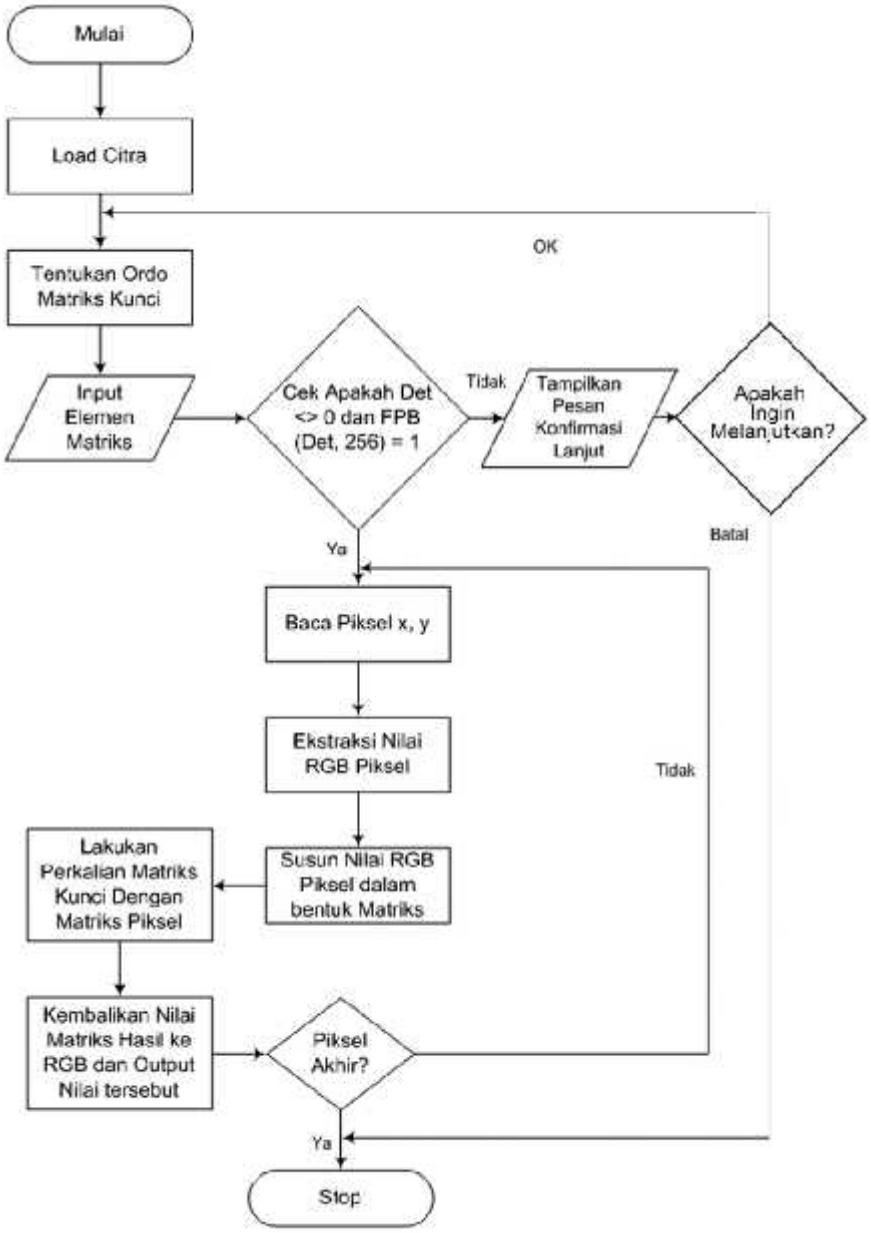
R_{P4} adalah nilai komponen *red* pada *pixel* keempat

G_{P4} adalah nilai komponen *green* pada *pixel* keempat

B_{P4} adalah nilai komponen *blue* pada *pixel* keempat

Secara umum gambaran cara kerja program sesuai dengan algoritma dari *Hill Cipher* yang diimplementasikan dalam mengenkripsi *cipher* dapat digambarkan dalam bentuk diagram alir seperti gambar 3.4 dibawah ini.

Gambar 3.5 Diagram Alir Enkripsi atau Penyandian Citra

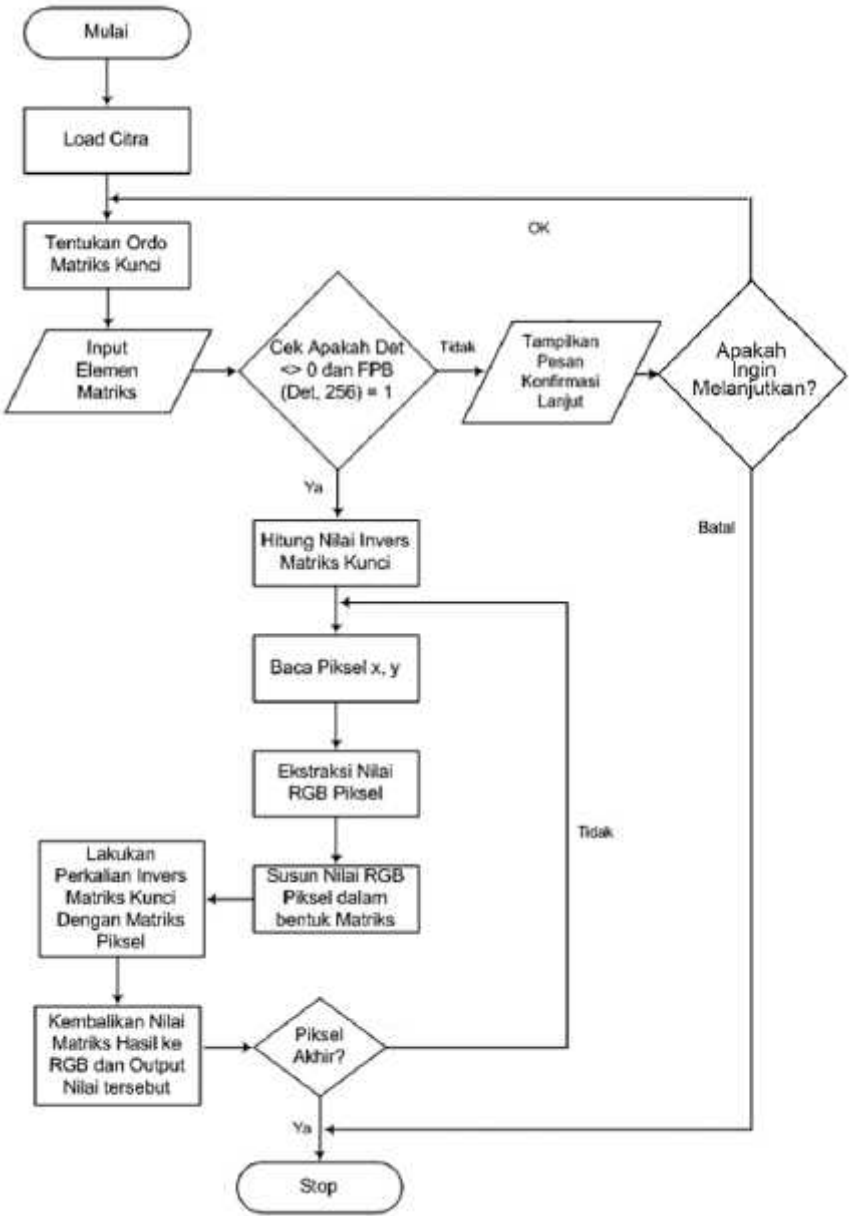


Keterangan:

1. Bagian ini merupakan inisialisasi variabel.
2. Bagian ini merupakan rutin untuk me-*load* citra ke *picture box* berdasarkan nama *file* citra yang dipilih.

3. Menentukan apakah matriks kunci berordo 2, 3 atau 4.
4. Input bilangan bulat untuk tiap elemen matriks kunci.
5. Lakukan pengecekan apakah matriks kunci yang diinput mempunyai determinan dan FPB (det, 256) = 1.
6. Jika tidak tampilkan pesan konfirmasi. Jika *user* memilih “OK” lanjut ke langkah 3.
7. Membaca nilai piksel dari citra sesuai algoritma pembacaan piksel.
8. Ekstraksi Nilai RGB seperti pada algoritma pembacaan piksel.
9. Susun Nilai RGB piksel dalam bentuk matriks sesuai dengan ordo matriks seperti cara di atas.
10. Lakukan perkalian matriks kunci dengan matriks piksel untuk menghasilkan matriks cipher. C = K . P Mod 256
11. Baca nilai RGB dari matriks *cipher* dan kembalikan dengan fungsi SetPixel.
12. Lakukan perulangan bila ternyata bukan piksel akhir dari citra lakukan kembali langkah 7-11.
13. Jika merupakan piksel akhir maka *stop*.

Sedangkan untuk proses sebaliknya yaitu dekripsi citra balik untuk mengembalikan citra asli dengan menggunakan algoritma dekripsi *Hill Cipher* dapat dinyatakan dengan alir (Gambar 3.5) di bawah ini



Gambar 3.6 Diagram Alir Dekripsi atau Penyandian Balik Citra

Keterangan:

1. Bagian ini merupakan inisialisasi variabel.
2. Bagian ini merupakan rutin untuk me-*load* citra ke picture *box* berdasarkan nama *file* citra yang dipilih.

3. Menentukan apakah matriks kunci berordo 2, 3 atau 4.
4. Input bilangan bulat untuk tiap elemen matriks kunci.
5. Lakukan pengecekan apakah matriks kunci yang diinput mempunyai determinan dan FPB (det, 256) = 1.
6. Jika tidak tampilkan pesan konfirmasi. Jika *user* memilih “OK” lanjut ke langkah 3.
7. Menghitung nilai invers dari matriks kunci.
8. Membaca piksel citra input sesuai dengan algoritma membaca piksel.
9. Mengekstraksi komponen RGB dengan algoritma membaca piksel.
10. Menyusun nilai RGB piksel sesuai dengan ordo matriks yang dipakai.
11. Lakukan perkalian invers matriks kunci dengan matriks piksel untuk memperoleh piksel citra asli kembali. $P = K^{-1}$. C Mod 256
12. Mengembalikan nilai matriks hasil ke nilai RGB dengan fungsi SetPixel.
13. Lakukan pengecekan apakah ini merupakan piksel terakhir. Jika tidak ulangi langkah 8-13.
14. Jika y *stop*.

1.4 Contoh Perhitungan *Hill Cipher*

Agar perhitungan cara kerja *Hill Cipher* pada citra dapat dimengerti dan jelas maka pada bagian ini akan dibahas dengan tiga buah contoh perhitungan baik dengan matriks 2 × 2 dan matriks 3 × 3. Berikutnya algoritma enkripsi *Hill Cipher* pada citra diterapkan pada barisan nilai RGB piksel dengan mengambil *n* = 256. Dekripsi juga dilakukan dengan cara yang sama. Misalkan matriks kunci

berordo 2 × 2 dan 3 × 3 dipakai untuk mengenkripsi potongan citra yang mempunyai 6 buah piksel dimana komponennya adalah:

$R_{P1} = 200$	$R_{P2} = 200$	$R_{P3} = 100$
$G_{P1} = 150$	$G_{P2} = 150$	$G_{P3} = 120$
$B_{P1} = 200$	$B_{P2} = 150$	$B_{P3} = 10$
$R_{P4} = 100$	$R_{P5} = 40$	$R_{P6} = 0$
$G_{P4} = 100$	$G_{P5} = 20$	$G_{P6} = 100$
$B_{P4} = 70$	$B_{P5} = 30$	$B_{P6} = 20$

Nilai-nilai RGB tersebut akan diambil dan diproses per tiap baris dengan matriks kunci. Untuk proses dengan matriks kunci 2 × 2 proses akan dilakukan tiap dua piksel per baris citra sedangkan proses dengan matriks kunci 3 × 3 akan dilakukan tiap tiga piksel per baris citra. Sebagai contoh di atas bila proses

dilakukan dengan matriks kunci 2 × 2 maka diperlukan tiga kali proses untuk menghasilkan *output.* Nilai tiap-tiap piksel akan disusun sebagai berikut:

$$P_1 = \begin{pmatrix} R_{P1} & G_{P1} & B_{P1} \\ R_{P2} & G_{P2} & B_{P2} \end{pmatrix};\quad P_2 = \begin{pmatrix} R_{P3} & G_{P3} & B_{P3} \\ R_{P4} & G_{P4} & B_{P4} \end{pmatrix};\quad P_3 = \begin{pmatrix} R_{P5} & G_{P5} & B_{P5} \\ R_{P6} & G_{P6} & B_{P6} \end{pmatrix}$$

Sedangkan bila proses dilakukan dengan matriks kunci 3 × 3 maka diperlukan dua kali proses saja untuk menghasilkan *output*. Nilai tiap-tiap piksel akan disusun sebagai berikut

$$P_1 = \begin{pmatrix} R_{P1} & G_{P1} & B_{P1} \\ R_{P2} & G_{P2} & B_{P2} \\ R_{P3} & G_{P3} & B_{P3} \end{pmatrix};\quad P_2 = \begin{pmatrix} R_{P4} & G_{P4} & B_{P4} \\ R_{P5} & G_{P5} & B_{P5} \\ R_{P6} & G_{P6} & B_{P6} \end{pmatrix}$$

Agar lebih jelasnya bagaimana cara perhitungan penyandian dengan *Hill Cipher* maka berikut ini piksel-piksel di atas akan disandikan atau dienkripsikan

dengan menggunakan tiga buah matriks berikut yaitu matriks 2 × 2 = $\begin{pmatrix} 2 & 1 \\ 5 & 3 \end{pmatrix}$ dan matriks 3 × 3 = $\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$

3.4.1 Perhitungan Penyandian dengan Matriks 2 × 2

Matriks Kunci K = $\begin{pmatrix} 2 & 1 \\ 5 & 3 \end{pmatrix}$ dan piksel yang akan diproses: P_1 = $\begin{pmatrix} 200 & 150 & 200 \\ 200 & 150 & 150 \end{pmatrix}$; P_2 $\begin{pmatrix} 100 & 120 & 10 \\ 100 & 10 & 70 \end{pmatrix}$; dan P_3 = $\begin{pmatrix} 40 & 20 & 30 \\ 0 & 100 & 20 \end{pmatrix}$.

Adapun persamaan yang digunakan untuk menghasilkan piksel *output* adalah sebagai berikut : $C_n = K \,.\, P_n \,.\, \text{Mod } 256$

Untuk proses pertama ($n = 1$) → $C_1 = K \,.\, P_1 \,.\, \text{Mod } 256$

$$C_1 = \begin{pmatrix} 2 & 1 \\ 5 & 3 \end{pmatrix}\begin{pmatrix} 200 & 150 & 200 \\ 200 & 150 & 150 \end{pmatrix} \text{ Mod } 256$$

$$C_1 = \begin{pmatrix} 600 & 450 & 550 \\ 1600 & 1200 & 1450 \end{pmatrix} \text{ Mod } 256$$

$$C_1 \quad \begin{pmatrix} 88 & 194 & 38 \\ 64 & 176 & 170 \end{pmatrix}$$

Untuk proses kedua ($n = 2$) → $C_2 = K \,.\, P_2 \,.\, \text{Mod } 256$

$$C_2 = \begin{pmatrix} 2 & 1 \\ 5 & 3 \end{pmatrix}\begin{pmatrix} 100 & 120 & 10 \\ 100 & 10 & 70 \end{pmatrix} \text{ Mod } 256$$

$$C_2 \quad \begin{pmatrix} 300 & 250 & 90 \\ 800 & 630 & 260 \end{pmatrix} \text{ Mod } 256$$

$$C_2 = \begin{pmatrix} 44 & 250 & 90 \\ 32 & 118 & 4 \end{pmatrix}$$

Untuk proses ketiga ($n = 3$) → $C_3 = K \,.\, P_3 \,\text{Mod } 256$

$$C_3 = \begin{pmatrix} 2 & 1 \\ 5 & 3 \end{pmatrix}\begin{pmatrix} 40 & 20 & 30 \\ 0 & 100 & 20 \end{pmatrix} \text{ Mod } 256$$

$$C_3 = \begin{pmatrix} 80 & 140 & 80 \\ 200 & 400 & 210 \end{pmatrix} \text{ Mod } 256$$

$$C_3 = \begin{pmatrix} 80 & 140 & 80 \\ 200 & 144 & 210 \end{pmatrix}$$

Dari perhitungan di atas untuk tiap pikselnya dalam komponen RGB didapat hasil sebagai berikut:

$R_{P1} = 88$ $G_{P1} = 194$ $B_{P1} = 38$	$R_{P2} = 64$ $G_{P2} = 176$ $B_{P2} = 170$	$R_{P3} = 44$ $G_{P3} = 250$ $B_{P3} = 90$
$R_{P4} = 32$ $G_{P4} = 118$ $B_{P4} = 4$	$R_{P5} = 80$ $G_{P5} = 140$ $B_{P5} = 80$	$R_{P6} = 200$ $G_{P6} = 144$ $B_{P6} = 210$

Nilai-nilai komponen RGB hasil perhitungan ini dengan menggunakan fungsi SetPixel akan dikembalikan menjadi sebuah piksel dengan warna tertentu sesuai dengan komponen warna Red, Green, dan Blue-nya. Untuk proses sebaliknya yaitu dekripsi langkah yang dipakai hampir sama dengan proses enkripsi. Perbedaannya adalah pada proses dekripsi matriks kunci harus diinvers dahulu dan hasilnya perkalian matriks tidak dimodulokan tetapi ditambahkan dengan nilai 256 hingga tidak dihasilkan bilangan negatif jika hasil yang didapat dalam bilangan negatif. Sedangkan untuk nilai yang didapat merupakan bilangan positif maka baru dimodulokan dengan bilangan 256.

Adapun persamaan yang digunakan untuk menghasilkan piksel *output* adalah sebagai berikut : $P_n = K^{-1} . C_n . \text{Mod } 256$ Sebagai contoh diambil dari proses enkripsi di atas:

Matriks Kunci K = $\begin{pmatrix} 2 & 1 \\ 5 & 3 \end{pmatrix}$ maka $K^{-1} = \frac{1}{\det K}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} = \frac{1}{1}\begin{pmatrix} 3 & -1 \\ -5 & 2 \end{pmatrix}$

$K^{-1}= \begin{pmatrix} 3 & -1 \\ 5 & 2 \end{pmatrix}$.

Piksel yang akan diproses: $C_1 = \begin{pmatrix} 88 & 140 & 80 \\ 200 & 144 & 210 \end{pmatrix}$; $C_2= \begin{pmatrix} 44 & 250 & 90 \\ 32 & 118 & 4 \end{pmatrix}$; dan $C_3= \begin{pmatrix} 80 & 140 & 80 \\ 200 & 140 & 210 \end{pmatrix}$

Untuk proses pertama (n = 1) →

$$P_1 \quad K^{-1} \cdot C_1$$

$$P_1 = \begin{pmatrix} 3 & 1 \\ -5 & 2 \end{pmatrix} \begin{pmatrix} 88 & 194 & 38 \\ 64 & 176 & 170 \end{pmatrix}$$

$$P_1 = \begin{pmatrix} 200 & 406 & 56 \\ -312 & -618 & 150 \end{pmatrix}$$

$$P_1 = \begin{pmatrix} 200 \ Mod\ 256 & 406 \ Mod\ 256 & 56 + (1 \times 256) \\ -312 + (2 \times 256) & -618 + (3 \times 256) & 150 \ Mod\ 256 \end{pmatrix}$$

$$P_1 = \begin{pmatrix} 200 & 150 & 200 \\ 200 & 150 & 150 \end{pmatrix}$$

Untuk proses kedua (n = 2) →

$$P_2 = K^{-1} \cdot C_2$$

$$P_2 = \begin{pmatrix} 3 & -1 \\ -5 & 2 \end{pmatrix} \begin{pmatrix} 44 & 250 & 90 \\ 32 & 118 & 4 \end{pmatrix}$$

$$P_2 = \begin{pmatrix} 100 & 632 & 266 \\ 156 & 1014 & 442 \end{pmatrix}$$

$$P_2 = \begin{pmatrix} 100 \ Mod\ 256 & 632 \ Mod\ 256 & 266 \ Mod\ 256 \\ -156 + (1 \times 256) & -1014 + (4 \times 256) & -442 + (2 \times 256) \end{pmatrix}$$

$$P_2 = \begin{pmatrix} 100 & 120 & 10 \\ 100 & 10 & 70 \end{pmatrix}$$

Untuk proses ketiga ($n = 3$) →

$$P_3 = K^{-1} \cdot C_3$$

$$P_3 = \begin{pmatrix} 3 & -1 \\ 5 & 2 \end{pmatrix} \begin{pmatrix} 80 & 140 & 80 \\ 200 & 144 & 210 \end{pmatrix}$$

$$P_3 = \begin{pmatrix} 40 & 276 & 30 \\ 0 & -412 & 20 \end{pmatrix}$$

$$P_3 = \begin{pmatrix} 40 \; Mod \; 256 & 276 \; Mod \; 256 & 30 \; Mod \; 256 \\ 0 \; Mod \; 256 & -412 + (2 \; x \; 256) & 20 \; Mod \; 256 \end{pmatrix}$$

$$P_3 = \begin{pmatrix} 40 & 20 & 30 \\ 0 & 100 & 20 \end{pmatrix}$$

Dari perhitungan di atas untuk tiap pikselnya dalam komponen RGB didapat hasil sebagai berikut:

$R_{P1} = 200$	$R_{P2} = 200$	$R_{P3} = 100$
$G_{P1} = 150$	$G_{P2} = 150$	$G_{P3} = 120$
$B_{P1} = 200$	$B_{P2} = 150$	$B_{P3} = 10$
$R_{P4} = 100$	$R_{P5} = 40$	$R_{P6} = 0$
$G_{P4} = 10$	$G_{P5} = 20$	$G_{P6} = 100$
$B_{P4} = 70$	$B_{P5} = 30$	$B_{P6} = 20$

Nilai-nilai komponen RGB hasil perhitungan ini dengan menggunakan fungsi SetPixel akan dikembalikan menjadi sebuah piksel dengan warna tertentu sesuai dengan komponen warna Red, Green, dan Blue-nya

3.4.2 Perhitungan Penyandian dengan Matriks 3 × 3.

Matriks Kunci K= $\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$ dan piksel yang akan diproses adalah sebagai berikut: $P_1 = \begin{pmatrix} 200 & 150 & 200 \\ 200 & 150 & 150 \\ 100 & 120 & 10 \end{pmatrix}$; $P_2 = \begin{pmatrix} 100 & 10 & 70 \\ 40 & 20 & 30 \\ 0 & 100 & 20 \end{pmatrix}$

Adapun persamaan yang digunakan untuk menghasilkan piksel output sama seperti penyandian dengan menggunakan matriks kunci 2 × 2 yaitu sebagai berikut :

$C_n = K \,.\, P_n \,.\, \text{Mod } 256$

Untuk proses pertama ($n = 1$) →

$K \,.\, P_1 \,.\, \text{Mod } 256$

$$C_1 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} . \begin{pmatrix} 200 & 150 & 200 \\ 200 & 150 & 150 \\ 100 & 120 & 10 \end{pmatrix} \text{Mod } 256$$

$$C_1 = \begin{pmatrix} 500 & 420 & 360 \\ 300 & 270 & 160 \\ 300 & 270 & 210 \end{pmatrix} \text{Mod } 256$$

$$C_1 = \begin{pmatrix} 244 & 164 & 104 \\ 44 & 14 & 160 \\ 44 & 14 & 210 \end{pmatrix}$$

$$C_1 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} . \begin{pmatrix} 200 & 150 & 200 \\ 200 & 150 & 150 \\ 100 & 120 & 10 \end{pmatrix} \text{Mod } 256$$

$$C_1 = \begin{pmatrix} 500 & 420 & 360 \\ 300 & 270 & 160 \\ 300 & 270 & 210 \end{pmatrix} \text{Mod } 256$$

$$C_1 = \begin{pmatrix} 244 & 164 & 104 \\ 44 & 14 & 160 \\ 44 & 14 & 210 \end{pmatrix}$$

Untuk proses pertama (n = 2) →

C2 = K . P2 . Mod 256

$$C_2 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} . \begin{pmatrix} 100 & 10 & 70 \\ 40 & 20 & 30 \\ 0 & 100 & 20 \end{pmatrix} \text{Mod } 256$$

$$C_2 = \begin{pmatrix} 140 & 130 & 120 \\ 40 & 120 & 50 \\ 100 & 110 & 90 \end{pmatrix} \text{Mod } 256$$

$$C_2 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} . \begin{pmatrix} 100 & 10 & 70 \\ 40 & 20 & 30 \\ 0 & 100 & 20 \end{pmatrix} \text{Mod } 256$$

$$C_2 = \begin{pmatrix} 140 & 130 & 120 \\ 40 & 120 & 50 \\ 100 & 110 & 90 \end{pmatrix} \text{Mod } 256$$

Dari perhitungan di atas untuk tiap pikselnya dalam komponen RGB didapat hasil sebagai berikut

$R_{P1} = 244$ $G_{P1} = 164$ $B_{P1} = 104$	$R_{P2} = 44$ $G_{P2} = 14$ $B_{P2} = 160$	$R_{P3} = 44$ $G_{P3} = 14$ $B_{P3} = 210$
$R_{P4} = 140$ $G_{P4} = 130$ $B_{P4} = 120$	$R_{P5} = 40$ $G_{P5} = 120$ $B_{P5} = 50$	$R_{P6} = 100$ $G_{P6} = 110$ $B_{P6} = 90$

Nilai-nilai komponen RGB hasil perhitungan ini dengan menggunakan fungsi SetPixel akan dikembalikan menjadi sebuah piksel dengan warna tertentu sesuai dengan komponen warna Red, Green, dan Blue-nya.

Untuk proses sebaliknya yaitu dekripsi langkah yang dipakai hampir sama dengan proses enkripsi dan aturan yang berlaku untuk perhitungan dengan matriks kunci 2 × 2 juga berlaku untuk perhitungan dengan matriks kunci 3 × 3.

Adapun persamaan yang digunakan untuk menghasilkan piksel *output* adalah sebagai berikut : $P_n = K^{-1}.C_n.\text{Mod } 256$

Sebagai contoh diambil dari proses enkripsi di atas:

Matriks kunci K = $\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$. Untuk mencari invers dari matriks ini maka digunakan persamaan berikut $K^{-1} = \frac{1}{\det K} adj\, K$. Untuk itu maka langkah pertama adalah mencari nilai determinan untuk matriks 3 × 3 di atas terlebih dahulu dengan menggunakan aturan Sarrus.

$$\text{Det K} = \begin{pmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{pmatrix} = (K_{11} * K_{22} * K_{33}) + (K_{12} * K_{23} * K_{31}) + (K_{13} * K_{21} * K_{32}) - (K_{31} * K_{22} * K_{13}) - (K_{32} * K_{23} * K_{11}) \quad (K_{33} * K_{21} * K_{12})$$

$$\text{Det K} - \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} - (1 * 1 * 1) + (1 * 1 * 1) + (1 * 0 * 0) - (1 * 1 * 1) - (0 * 1 * 1) - (1 * 1 * 0) = 1 + 1 + 0 - 1 - 0 - 0 = 1$$

SSedangkan untuk mencari nilai **adj** K (dibaca *adjoin* matriks K) ditentukan dengan persamaan:

Adj K = $\begin{pmatrix} \sim_{11} & \sim_{21} & \sim_{31} \\ \sim_{12} & \sim_{22} & \sim_{32} \\ \sim_{13} & \sim_{23} & \sim_{33} \end{pmatrix}$ dengan α_{ij} adalah kofaktor dari K_{ij} yang dapat dicari

dengan menggunakan rumus: $\alpha ij = (-1)^{i+j} |M_{ij}|$ dimana $|M_{ij}|$ adalah minor dari K_{ij} dari matriks K. Minor dicari dengan menghapus elemen-elemen pada baris ke-i dan kolom ke-j dari matriks K maka akan diperoleh matriks persegi berordo 2. Determinan dari matriks persegi berordo 2 itu yang disebut dengan minor.

Maka untuk menentukan minor dan kofaktor dari matriks K di atas adalah sebagai berikut :

Kofaktor dari K_{11} adalah α_{11} $(1)^{1+1} |M_{11}|$ $- \begin{vmatrix} 1 & 1 \\ 0 & 1 \end{vmatrix}$ $+(1 . 1 \quad 1 . 0) \quad 1$

Kofaktor dari K_{12} adalah α_{12} $(1)^{1+2} |M_{12}|$ $\begin{matrix} 0 & 1 \\ 1 & 1 \end{matrix}$ $(0 . 1 \quad 1 . 1) \quad 1$

Kofaktor dari K_{13} adalah α_{13} $(1)^{1+3} |M_{13}|$ $+ \begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix}$ $-(0 . 0 \quad 1 . 1) \quad 1$

Kofaktor dari K_{21} adalah α_{21} $(1)^{2+1} |M_{21}|$ $\begin{matrix} 1 & 1 \\ 0 & 1 \end{matrix}$ $(1 . 1 \quad 1 . 0) \quad 1$

Kofaktor dari K_{22} adalah α_{22} $(1)^{2+2} |M_{22}|$ $+ \begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix}$ $+(1 . 1 \quad 1 . 1) \quad 0$

Kofaktor dari K_{23} adalah $\alpha_{23} = (-1)^{2+3} |M_{23}| = - \begin{matrix} 1 & 1 \\ 1 & 0 \end{matrix} = -(1 . 0 - 1 . 1) = 1$

Kofaktor dari K_{31} adalah $\alpha_{31} - (-1)^{3+1} |M_{31}| - + \begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix} - +(1 . 1 - 1 . 1) - 0$

Kofaktor dari K_{32} adalah $\alpha_{32} - (-1)^{3+2} |M_{32}| - - \begin{vmatrix} 1 & 1 \\ 0 & 1 \end{vmatrix} - -(1 . 1 - 1 . 0) - -1$

Kofaktor dari K_{33} adalah $\alpha_{33} - (-1)^{3+3} |M_{33}| - + \begin{vmatrix} 1 & 1 \\ 0 & 1 \end{vmatrix} - +(1 \quad 1 - 1 . 0) - 1$

Dari hasil kofaktor yang didapat maka hasil disusun kembali untuk membentuk adj K yaitu: Adj K = $\begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ -1 & 1 & 1 \end{pmatrix}$; maka untuk mencari invers dari K atau K^{-1} adalah sebagai berikut: $K^{-1} = \frac{1}{1}\begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ -1 & 1 & 1 \end{pmatrix}$ didapat hasil: $K^{-1} = \begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ -1 & 1 & 1 \end{pmatrix}$

Selanjutnya untuk mendapatkan piksel citra hasil maka dilakukan perkalian matriks dengan menggunakan persamaan: $P_n = K^{-1} . C_n . \text{Mod } 256$

Matriks Kunci K^{-1} $\begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ -1 & 1 & 1 \end{pmatrix}$ dan piksel yang akan diproses adalah sebagai berikut: $C_1 = \begin{pmatrix} 244 & 164 & 104 \\ 44 & 14 & 160 \\ 44 & 14 & 210 \end{pmatrix}$; $C_2 = \begin{pmatrix} 140 & 130 & 120 \\ 40 & 120 & 50 \\ 100 & 110 & 99 \end{pmatrix}$

Untuk proses pertama ($n = 1$) →

$P_1 = K^{-1} . C_1$

$$P_1 = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ -1 & 1 & 1 \end{pmatrix} . \begin{pmatrix} 244 & 164 & 104 \\ 44 & 14 & 160 \\ 44 & 14 & 210 \end{pmatrix}$$

$$P_1 = \begin{pmatrix} 200 & 150 & 56 \\ 200 & 150 & -106 \\ -156 & -136 & 266 \end{pmatrix}$$

$$P_1 = \begin{pmatrix} 200 \text{ Mod } 256 & 150 \text{ Mod } 256 & 56 + 1.256 \\ 200 \text{ Mod } 256 & 150 \text{ Mod } 256 & 106 + 1.256 \\ -156 + 1.256 & -136 + 1.256 & 266 \text{ Mod } 256 \end{pmatrix}$$

$$P_1 = \begin{pmatrix} 200 & 150 & 200 \\ 200 & 150 & 150 \\ 100 & 120 & 10 \end{pmatrix}$$

Untuk proses kedua ($n = 2$) $\rightarrow P_2 = K^{-1} C_2$

$$P_2 = \begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & -1 \\ -1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 140 & 130 & 120 \\ 40 & 120 & 50 \\ 100 & 110 & 90 \end{pmatrix}$$

$$P_2 = \begin{pmatrix} 100 & 10 & 70 \\ 40 & 20 & 30 \\ 0 & 100 & 20 \end{pmatrix}$$

$$P_2 = \begin{pmatrix} 100 \text{ Mod } 256 & 10 \text{ Mod } 256 & 70 \text{ Mod } 256 \\ 40 \text{ Mod } 256 & 20 \text{ Mod } 256 & 30 \text{ Mod } 256 \\ 0 \text{ Mod } 256 & 100 \text{ Mod } 256 & 20 \text{ Mod } 256 \end{pmatrix}$$

$$P_2 = \begin{pmatrix} 100 & 10 & 70 \\ 40 & 20 & 30 \\ 0 & 100 & 20 \end{pmatrix}$$

Dari perhitungan di atas untuk tiap pikselnya dalam komponen RGB didapat hasil sebagai berikut:

$R_{11} = 200$ $G_{11} = 150$ $B_{11} = 200$	$R_{12} = 200$ $G_{12} = 150$ $B_{12} = 150$	$R_{13} = 100$ $G_{13} = 120$ $B_{13} = 10$
$R_{21} = 100$ $G_{21} = 10$ $B_{21} = 70$	$R_{22} = 40$ $G_{22} = 20$ $B_{22} = 30$	$R_{23} = 0$ $G_{23} = 100$ $B_{23} = 20$

Nilai-nilai komponen RGB hasil perhitungan ini dengan menggunakan fungsi SetPixel akan dikembalikan menjadi sebuah piksel dengan warna tertentu sesuai dengan komponen warna Red, Green, dan Blue-nya.

3.5 Perancangan

Antarmuka merupakan suatu media interaksi (interaktif) antara komputer dengan pemakai (user). Pada system operasi yang berbasis grafis (*graphis user interface* atau GUI) seperti Windows, antarmuka dari suatu perangkat lunak biasanya berupa jendela (*window).* Melalui jendela inilah pemakai dapat berinteraksi dengan perangkat lunak yang digunakannya. Perancangan program ini meliputi penempatan dan penyusunan objek-objek yang terdapat dalam *form Visual Basic*. *Form* yang dirancang terlebih dahulu dibuat rancangan dasarnya agar memudahkan sewaktu membuat rancangan di dalam Visual Basic.

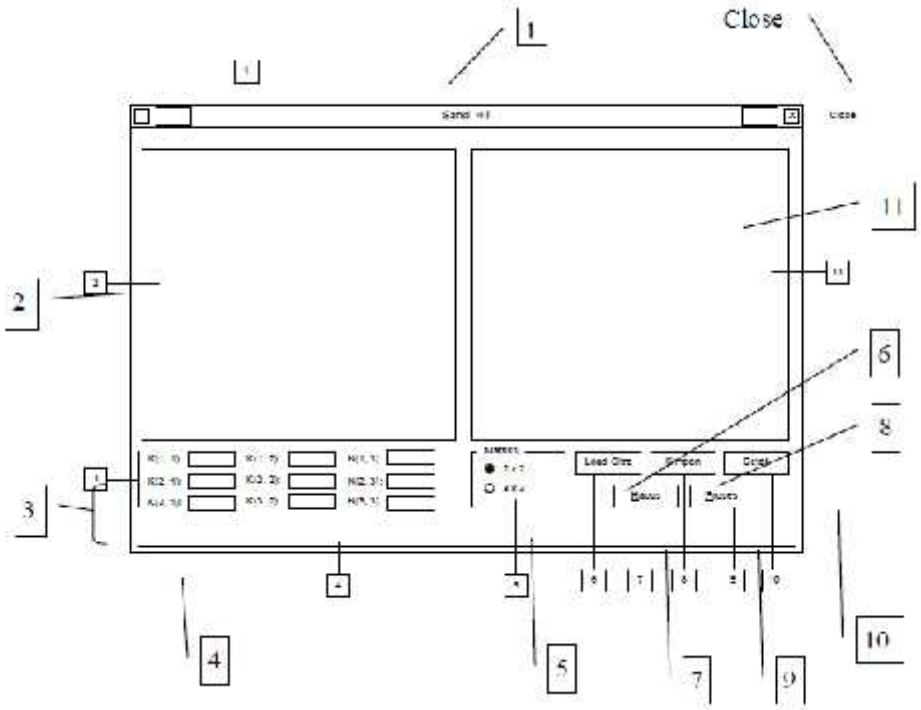
3.5.1 Perancangan *Form*

Form pada program ini adalah terdiri atas tiga buah *form* yaitu *form splash, form* utama, *form* konfirmasi. Adapun fungsi dari ketiga *form-form* ini dapat dirinci sebagai berikut:

1. *Form Splash* untuk memunculkan keterangan program beserta nama penulis pada saat pertama kali program di-*load*. Dengan kata lain *form* ini berfungsi sebagai *splash screen*.
2. *Form* Utama merupakan bentuk tampilan utama program dimana *user* dapat melakukan hal-hal seperti me-*load* citra yang akan diproses, menyimpan, memprosesnya dengan *Hill Cipher*.
3. *Form* Konfirmasi mempunyai fungsi untuk meminta konfirmasi *user* jumlah iterasi yang akan dilakukan sekaligus meminta *user* apakah akan

melakukan proses enkripsi atau sebaliknya dekripsi pada citra yang diinput.

Bentuk rancangan dasar atau biasanya disebut dengan layout dari *form* utama dapat dilihat pada Gambar 3.6 berikut ini:

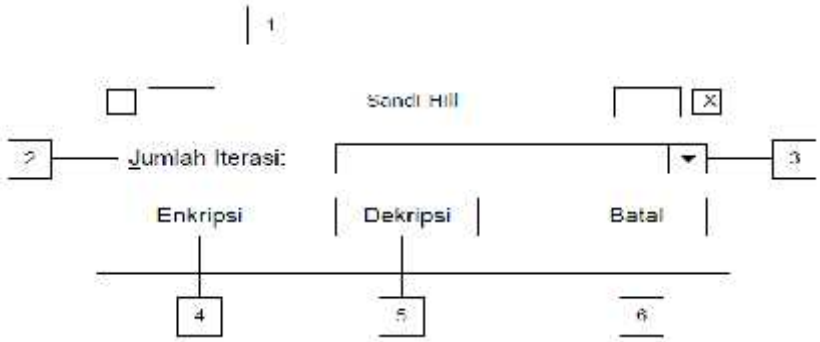


Gambar 3.7 Perancangan *Form* Utama

Pada gambar di atas terdapat beberapa komponen-komponen visual yang mempunyai fungsi-fungsi sebagai berikut:

1. Bagian ini biasanya disebut dengan *title bar* yang berfungsi sebagai judul program.
2. Bagian ini adalah objek *picture box* yang digunakan sebagai tempat untuk memunculkan citra input oleh *user*.

3. Bagian ketiga ini dibuat dari objek label untuk label keterangannya sedangkan kotak pengisiannya dibuat *text box*. Bagian ini berfungsi untuk meminta input matriks kunci (K).
4. Bagian ini merupakan progress bar yang berfungsi untuk menampilkan status kemajuan proses pada saat *enciphering* dan *deciphering* citra.
5. Bagian ini dibuat dengan dua jenis objek yaitu objek *frame* untuk mengelompokkan tiga buah *option button* yang ada di dalamnya. Berfungsi untuk menentukan apakah ordo matriks adalah dua, tiga atau empat. Bila pilihannya hanya dua yang ditandai maka bagian keempat hanya tersedia empat buah *text box* untuk diisi nilainya.
6. Bagian ini merupakan *command button* yang berfungsi untuk memilih *file* citra yang akan dijadikan citra input. Bila *user* mengklik pada tombol ini akan dimunculkan sebuah kotak dialog *open*.
7. Bagian ini merupakan objek *command button* yang bertujuan untuk membersihkan *picture box* pada bagian ketiga dan kesebelas.
8. Bagian ini merupakan objek *command button* yang berfungsi untuk menyimpan citra *output* ke dalam format *bitmap* dengan memunculkan sebuah kotak dialog *save*.
9. Bagian ini merupakan objek *command button* yang berfungsi untuk memunculkan *form* konfirmasi.
10. Bagian ini merupakan objek *command button* yang berfungsi untuk mencetak citra hasil ke *printer*. Bentuk rancangan dasar dari *form* konfirmasi dapat dilihat pada gambar 3.8 dibawah ini.



Gambar 3.8 Perancangan *Form* Konfirmasi

Pada gambar diatas terdapat beberapa komponen-komponen visual yang mempunyai fungsi-fungsi sebagai berikut:

1. Bagian ini biasanya disebut dengan *title bar* yang berfungsi sebagai judul program.
2. Bagian ini merupakan objek label yang berfungsi untuk mencetak tulisan jumlah Iterasi.
3. Bagian ini merupakan objek *combo* box untuk mengisi dan memilih jumlah Iterasi.
4. Bagian ini merupakan objek *command button* untuk melakukan proses *ciphering* citra.
5. Bagian ini merupakan objek *command button* untuk melakukan proses *deciphering* citra.
6. Bagian ini merupakan objek *command button* untuk membatalkan proses, menutup *forn* konfirmasi dan tampilan kembali ke *form* utama.

4.5.2 Perancangan *Class Module*

Pada perancangan dan pembuatan program ini dirancang sebuah *Class Module* yang diberi nama clsHill.cls (singkatan dari Class Hill). Pada *class module* ini dikodekan rutin-rutin untuk proses *ciphering* dan *deciphering* dengan algoritma *Hill Cipher*. Fungsi pembacaan *piksel* juga disertakan dalam *class module* ini. Adapun rutin-rutin yang terdapat pada *class module* ini adalah :

1. Function Enhill adalah fungsi untuk melakukan proses *ciphering.* Fungsi ini akan mengahasilkan nilai *boolean* berupa *true* jika proses *ciphering* berhasil dan *false* jika proses *ciphering* gagal. Bentuk deklarasinya adalah sebagai berikut: Function EnHill(PicSource As PictureBox, picDest As PictureBox, MatrixKey(), MatrixSize As Byte) As Boolean. Fungsi diatas memiliki empat buah parameter yaitu sebagai berikut:
   a. picSource yang merupakan parameter input citra yang akan disandikan dengan *Hill Cipher*.
   b. picDest merupakan parameter citra output hasil penyandian dengan *Hill Cipher*.
   c. MatrixKey() merupakan *array* matriks kunci yang berisikan nilai-nilai dalam bentuk bilangan bulat.
   d. MatrixSize merupakan parameter untuk menyatakan ukuran ordo matriks kunci.
2. Fungsi DeHill adalah fungsi untuk melakukan proses *deciphering.* Fungsi ini akan menghasilkan nilai *Boolean* berupa *true* jika proses *deciphering* berhasil dan *false* jika proses *deciphering* gagal. Bentuk deklarasinya adalah sebagai

berikut: DeHill(picSource As PictureBox, picDest As PictureBox, MatrixKey(), MatrixSize As Byte) As Boolean. Fungsi di atas memiliki empat buah parameter yaitu sebagai berikut:

a. picSource merupakan parameter *input* citra yang akan disandikan balik dengan *Hill Cipher*.
b. picDest merupakan parameter citra *output* hasil penyandian balik dengan *Hill Cipher*.
c. MatrixKey() merupakan *array* matriks kunci yang berisikan nilai-nilai dalam bentuk bilangan bulat.
d. MatrixSize merupakan parameter untuk menyatakan ukuran ordo matriks kunci.

BAB IV

ALGORITMA DAN INMPLEMENTASI

4.1 Algoritma

Sebagai upaya dalam mempermudah proses penulisan *coding* dengan bahasa pemrograman yang telah ditetapkan sebelumnya, maka penulis merancang algoritma program yang digunakan sebagai bahan acuan. Bentuk rancangan algoritma program dari implementasi Algoritma *End of File* (EOF).

4.2 Kebutuhan Sistem

Implementasi sistem adalah prosedur yang dilakukan untuk menyelesaikan desain sistem yang ada didalam dokumen yang disetujui, hal ini untuk menguji, menginstal serta memulai menggunakan sistem baru atau sistem yang diperbaiki, tujuannya adalah:

1. Mentiapkan desain sistem yang ada dalam dokumen desain sistem yang disetujui untuk menyusun dokumen baru atau dokumen yang akan diperbaiki.
2. Menyediakan perangkat keras *(hardware)* dan perangkat lunak *(software).* Tahap ini merupakan persiapan untuk penulis merancang program yang sudah direncanakan, untuk itu penulis menyediakan segala perangkat lunak diantaranya sistem operasi, bahasa pemrograman.

3. Menulis program ke komputer setelah *software* dan hardware tersedia, maka penulis akan mengetik instruksi atau listing rancangan sistem ke komputer sesuai dengan bahasa pemrograman yang digunakan.
4. Menguji sistem, tahap ini merupakan pengevaluasian apakah sistem yang telah dibuat sesuai dengan yang diharapkan.

4.2.1 Perangkat Keras

Perangkat keras merupakan komponen yang membentuk sistem pengolahan data dan peralatan lainnya yang mendukung terlaksananya pengolahan data. Agar perangkat lunak dalam penerapan kriptografi rantai segitiga ini dapat berjalan dengan baik ada beberapa perangkat yang harus dipenuhi. Adapun perangkat keras *(hardware)* yang digunakan untuk menjalankannya antara lain sebagai berikut :

a. 1 (satu) unit *Laptop* dengan *Processor Intel Core i3-2330M*
b. RAM (Memory) 2 *GB DDR3 Memory*
c. *Harddisk* dengan kapasitas penyimpanan 320 *Gigabyte HDD*
d. Dengan *keyboard* dan *Mouse* sebagai media *inputan*
e. 1 (satu) buah printer sebagai media *output*

Processor adalah komponen utama dari sebuah komputer yang berfungsi sebagai pusat pengolahan data, bisa juga dikatakan sebagai otaknya komputer. Harddisk adalah media penyimpanan data yang bersifat permanen dan mempunyai daya tampung yang bervariasi serta kecepatan akses yang lebih tinggi dibanding media penyimpanan lainnya. *Memory* komputer adalah tempat pengolahan data

yang membantu mempercepat pemrosesan data dan berfungsi sebagai media penyimpanan yang bersifat sementara.

4.2.2 Perangkat Lunak

Perangkat lunak *(software)* yang digunakan pada saat implementasi keamanan data dengan sfesifikasi berikut:

a. Sistem operasi (*operating sistem*) yaitu *Windows 7*.
b. *Visual Studio 2008* untuk mengetikkan listing program.

4.3 Implementasi Sistem

Dari hasil implementasi rancangan sistem yamg dibuat ini telah sesuai dengan yang diharapkan. Berikut ini bentuk hasil dari perancangan sistem tersebut.

4.3.1 Menu Utama

Menu utama ini digunakan untuk melakukan pilihan pada menu utama seperti terlihat pada gambar dibawah ini :

Gambar 4.1 Menu Utama

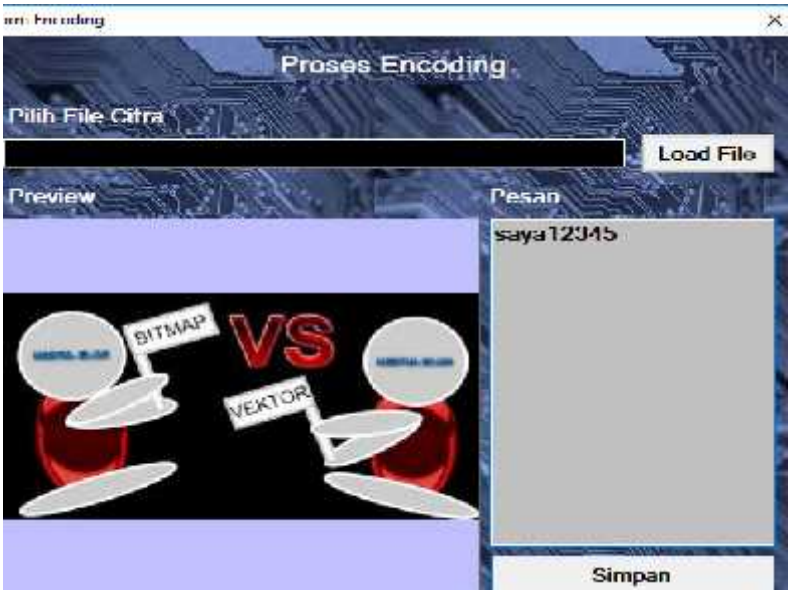
Penjelasan Menu utama.

Dalam halaman utama terdapat beberapa menu yaitu:

1. File : Merupakan menu yang digunakan untuk melakukan proses *ecoding* dan *decoding* pada Algoritma *End of File* (EOF)
2. Exit : Merupakan menu yang digunakan untuk keluar dari menu utama

4.3.2 Form *Ecoding*

Form utama ini digunakan untuk melakukan proses pilihan *ecoding* atau, seperti terlihat pada gambar dibawah ini :



Gambar 4.2 Form *Ecoding*

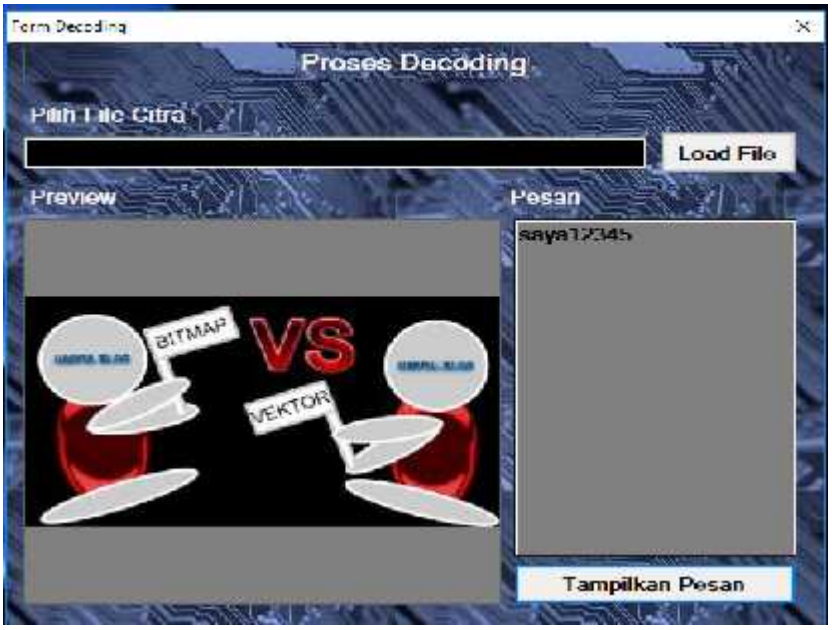
Penjelasan Form Enkripsi

Dalam halaman utama terdapat beberapa menu yaitu:

1. Plainteks : merupakan menu yang digunakan untuk mengupload gambar
2. Kunci : merupakan menu yang digunakan untuk kunci berupa angka dan huruf
3. Hasil *ecoding*: menu tempat hasil setelah proses *ecoding*
4. Keluar : merupakan menu yang digunakan untuk keluar

4.3.3 Form *Decoding*

Form utama ini digunakan untuk melakukan proses pilihan *decoding*, seperti terlihat pada gambar dibawah ini :



Gambar 4.3 Form *Decoding*

Penjelasan Form Dekripsi

Dalam halaman utama terdapat beberapa menu yaitu:

1. Hasil : menu tempat hasil setelah proses *decoding*
2. *Decoding* : menu untuk melakukan proses *decoding*
3. Keluar : merupakan menu yang digunakan untuk keluar

4.3.4 Form About Me

Form utama ini digunakan untuk melihat dari profil penulis, seperti terlihat pada gambar dibawah ini :

Gambar 4.4 Form About Me

Penjelasan Form About Me

Dalam halaman utama terdapat beberapa menu yaitu:

1. Keluar : merupakan menu yang digunakan untuk keluar

4.4 Kelebihan dan Kelemahan Sistem

Adapun kelebihan dan kelemahan algoritma *End of File* (EOF)yang yang dirancang seperti berikut :

1. Kelebihan sistem
   a. Aplikasi *End of File* (EOF)ini dapat mempermudah seseorang dalam merahasiakan atau menyisipkan pesan kedalam gambar terhadap sebuah data atau informasi rahasia.
   b. Proses perahasiaan pesan atau data dapat dilakukan dengan mudah.

c. Data atau informasi yang dirahasiakan akan lebih terjamin keamanan kualitas dan kerahasiaan data juga lebih terjaga.

2. Kelemahan sistem
   a. Aplikasi kriptografi hanya dalam ruang lingkup merahasiakan pesan dengan teknik Hill Cipher.
   b. Menu yang terdapat dalam aplikasi hanya melakukan proses penyisipan pesan kedalam gambar, hal ini dikarenakan keterbatasan ilmu pengetahan dan juga keterbatasan waktu pembuatan progran.
   c. Sistem informasi yang dirancang masih bersifat satu pengguna *(single user)*, belum menerapkan sistem jaringan (*multi user*).

BAB V

KESIMPULAN DAN SARAN

5.1 Simpulan

Berdasarkan aplikasi yang telah dibuat beserta uji coba yang telah dilakukan, maka dapat disimpulkan sebagai berikut :

1. Dengan adanya penerapan Algoritma *End of File* (EOF) maka pengamanan dalam penyisipan teks pada gambar yang bertipe Bmp yang dilakukan secara manual diarahkan kepada pemanfaatan teknologi komputer.
2. Aplikasi penyisipan file teks pada gambar ini memiliki rancangan format yang mudah dimengerti sehingga akan memberikan kemudahan dalam proses penyisipan pesan atau data rahasia oleh pengguna.
3. Algoritma *End of File* (EOF) ini digunakan untuk memberikan keamanan pesan yang disisipkan pada gambar yang bersifat rahasia.

5.2 Saran

Adapun yang menjadi saran yang dapat disampaikan adalah sebagai berikut :

1. Diharapkan perkembangan yang lebih dari program yang dirancang agar dapat dipadu terhadap kriptografi moderen dengan pembelajaran metode

penyandian pesan teks pada gambar yang lain sehingga menjadi aplikasi yang tangguh.

2. Diharapkan program yang telah dirancang ini akan meningkatkan mutu kualitas keamanan data.
3. Program ini masih jauh dari sempurna dan penulis berharap untuk kedepannya program ini dapat dikembangkan lagi jauh lebih baik dari sebelumnya.
4. Diharapkan kepada pengguna aplikasi menjalin kerja sama yang baik antara sipengirim dan sipenerima data agar tidak terjadi kecurangan dan pesan rahasia akan lebih terjamin kualitasnya.

DAFTAR PUSTAKA

Ariyus Dony, 2006, ***Keamanan Data dan Komunikasi***, Jakarta, Penerbit Graha Ilmu.

Bahri, S. (2018). ***Metodologi Penelitian Bisnis Lengkap Dengan Teknik Pengolahan Data SPSS***. Penerbit Andi (Anggota Ikapi). Percetakan Andi Ofsset. Yogyakarta.

Barus, S., Sitorus, V. M., Napitupulu, D., Mesran, M., & Supiyandi, S. (2018). ***Sistem Pendukung Keputusan Pengangkatan Guru Tetap Menerapkan Metode Weight Aggregated Sum Product Assesment (WASPAS)***. Jurnal Media Informatika Budidarma, 2(2).

Desi Lilyani (2014). ***Implementasi Steganografi Pada Citra Digital DenganMenggunakan Metode Dynamic Cell Spreading*** , Pelita Informatika BudiDarma, 1-8

Diantoro, M., Maftuha, D., Suprayogi, T., Iqbal, M. R., Mufti, N., Taufiq, A., ... & Hidayat, R. (2019). ***Performance of Pterocarpus Indicus Willd Leaf Extract as Natural Dye TiO2-Dye/ITO DSSC***. Materials Today: Proceedings, 17, 1268-1276.

Fitriani, W., Rahim, R., Oktaviana, B., & Siahaan, A. P. U. (2017). ***Vernam Encypted Text in End of File Hiding Steganography Technique***. Int. J. Recent Trends Eng. Res, 3(7), 214-219.

Hariyanto, E., Iqbal, M., Siahaan, A. P. U., Saragih, K. S., & Batubara, S. (2019, March). ***Comparative Study of Tiger Identification Using Template Matching Approach based on Edge Patterns***. In Journal of Physics: Conference Series (Vol. 1196, No. 1, p. 012025). IOP Publishing.

Hidayat, Wildan. 2013. ***Perlindungan Pesan Rahasia Pada Citra Digital Menggunakan Metode Least Significant Bit Steganografi***. Skripsi. Medan, Indonesia : Universitas Sumatera Utara

Kurniawan, H. (2018). ***Pengenalan Struktur Baru untuk Web Mining dan Personalisasi Halaman Web***. Jurnal Teknik dan Informatika, 5(2), 13-19.

Khairul, K., IlhamiArsyah, U., Wijaya, R. F., & Utomo, R. B. (2018, September). ***Implementasi augmented reality sebagai media promosi penjualan rumah***. In Seminar Nasional Royal (SENAR) (Vol. 1, No. 1, pp. 429-434).

Krisnawati (2008). Metode *Least Significant Bit* (LSB) Dan *End Of File* (EOF) Untuk Menyisipkan Teks Ke Dalam Citra *Grayscale,* Seminar Nasional Informatika, 39-44.

Ramadhan, Z., Zarlis, M., Efendi, S., & Siahaan, A. P. U. (2018). ***Perbandingan Algoritma Prim dengan Algoritma Floyd-Warshall dalam Menentukan Rute Terpendek (Shortest Path Problem)***. JURIKOM (Jurnal Riset Komputer), 5(2), 135-139.

Rahim, R., Supiyandi, S., Siahaan, A. P. U., Listyorini, T., Utomo, A. P., Triyanto, W. A., ... & Khairunnisa, K. (2018, June). ***TOPSIS Method Application for Decision Support System in Internal Control for Selecting Best Employees.*** In Journal of Physics: Conference Series (Vol. 1028, No. 1, p. 012052). IOP Publishing.

Sadikin Rifki, 2007, ***Kriptografi untuk Keamanan Jaringan***, Jakarta, Penerbit Andi.

Santoni, 2011, ***Algoritma Chiper Subtitusi Rantai Segitiga***, Jurnal Teknologi dan Informatika, Vol.4, No.2.

Sibero Alexander F.K, 2010, ***Dasar Dasar Visual Basic.Net***, Yogyakarta, Penerbit MediaKom.

Suherman, S., & Khairul, K. (2018). ***Seleksi Pegawai Kontrak Menjadi Pegawai Tetap Dengan Metode Profile Matching***. IT Journal Research and Development, 2(2), 68-77.

Sulistianingsih, I., Suherman, S., & Pane, E. (2019). ***Aplikasi Peringatan Dini Cuaca Menggunakan Running Text Berbasis Android***. IT Journal Research and Development, 3(2), 76-83.

Sari, R. D., Supiyandi, A. P. U., Siahaan, M. M., & Ginting, R. B. (2017). ***A Review of IP and MAC Address Filtering in Wireless Network Security***. Int. J. Sci. Res. Sci. Technol, 3(6), 470-473.

Sidik, A. P., Efendi, S., & Suherman, S. (2019, June). ***Improving One-Time Pad Algorithm on Shamir's Three-Pass Protocol Scheme by Using RSA and ElGamal Algorithms***. In Journal of Physics: Conference Series (Vol. 1235, No. 1, p. 012007). IOP Publishing.

Siahaan, A. P. U., Aryza, S., Nasution, M. D. T. P., Napitupulu, D., Wijaya, R. F., & Arisandi, D. (2018). ***Effect of matrix size in affecting noise reduction level of filtering.***

Tasril, V. (2018). ***Sistem Pendukung Keputusan Pemilihan Penerimaan Beasiswa Berprestasi Menggunakan Metode Elimination Et Choix Traduisant La Realite***. INTECOMS: Journal of Information Technology and Computer Science, 1(1), 100-109.

Widyartono Agustinus, 2011, ***Algoritma Elgamal Untuk Enkripsi Data Menggunakan GNUPG***, Jurnal Teknologi dan Informatika, Vol.1, No.1.

Zelvina Anandia, et..al, 2012, ***Perancangan Aplikasi Pembelajaran Kriptografi Kunci Publik Elgamal Untuk Mahasiswa,*** Jurnal Dunia Teknologi Informasi, Vol.1, No.1, 56-62.