BAB II

LANDASAN TEORI

II.1. Steganografi

Steganografi adalah seni dan ilmu menulis atau menyembunyikan pesan tersembunyi dengan suatu cara sehingga selain si pengirim dan si penerima, tidak ada seorangpun yang mengetahui atau menyadari bahwa ada suatu pesan rahasia. Sebaliknya, kriptografi menyamarkan arti dari suatu pesan, tapi tidak menyembunyikan bahwa ada suatu pesan. Kata steganografi berasal dari bahasa Yunani yaitu *steganos*, yang artinya "tersembunyi atau terselubung", dan *graphein*, "menulis" (*Andreas, 2014*).

Teknik steganografi meliputi banyak sekali metode komunikasi untuk menyembunyikan pesan rahasia (teks atau gambar) didalam *file-file* lain yang mengandung teks, gambar, bahkan audio tanpa menunjukkan ciri-ciri perubahan yang nyata atau terlihat dalam kualitas dan struktur dari file semula. Metode ini termasuk tinta yang tidak tampak, *microdots*, pengaturan kata, tanda tangan digital, jalur tersembunyi dan komunikasi spektrum lebar. Tujuan dari steganografi adalah merahasiakan atau menyembunyikan keberadaan dari sebuah pesan tersembunyi atau sebuah informasi. Dalam prakteknya kebanyakan diselesaikan dengan membuat perubahan tipis terhadap data digital lain yang isinya tidak akan menarik perhatian dari penyerang potensial, sebagai contoh sebuah gambar yang terlihat tidak berbahaya. Perubahan ini bergantung pada kunci (sama pada kriptografi) dan pesan untuk disembunyikan. Orang yang

menerima gambar kemudian dapat menyimpulkan informasi terselubung dengan cara mengganti kunci yang benar kedalam algoritma yang digunakan (*Andreas, 2014*).

Pada metode steganografi cara ini sangat berguna jika digunakan pada cara steganografi komputer karena banyak format *file* digital yang dapat dijadikan media untuk menyembunyikan pesan. Format yang biasa digunakan diantaranya :

1. Format gambar : bitmap (bmp), gif, pcx, jpeg, dll.
2. Format audio : wav, voc, mp3, dll.
3. Format lain : teks file, html, pdf, dll.

Kelebihan steganografi dari pada kriptografi adalah pesan-pesannya tidak menarik perhatian orang lain. Pesan-pesan berkode dalam kriptografi yang tidak disembunyikan, walaupun tidak dapat dipecahkan, akan menimbulkan kecurigaan. Seringkali, steganografi dan kriptografi digunakan (*Andreas, 2014*). secara bersamaan untuk menjamin keamanan pesan rahasianya

Sebuah pesan steganografi (*plaintext*), biasanya pertama-tama dienkripsikan dengan beberapa arti tradisional, yang menghasilkan *ciphertext*. Kemudian, *covertext* dimodifikasi dalam beberapa cara sehingga berisi *ciphertext*, yang menghasilkan *stegotext*. Contohnya, ukuran huruf, ukuran spasi, jenis huruf, atau karakteristik *covertex*t lainnya dapat dimanipulasi untuk membawa pesan tersembunyi, hanya penerima (yang harus mengetahui teknik yang digunakan) dapat membuka pesan dan mendekripsikannya (*Andreas, 2014*).

II.1.1 Tujuan Steganografi

Sebenarnya ilmu Steganografi diera digital ini sangat penting untuk diketahui dan pelajari. Karena hampir semua sektor kegiatan manusia modern saat ini sudah menggunakan informasi yang berbentuk digital. Mulai dari belajar disekolah, bermain, menyelesaikan pekerjaan dikantor, menyimpan arsip-arsip penting, sampai dengan melakukan penelitian senjata rahasia kimia.

Hampir semua kegiatan manusia modern termasuk berkomunikasi sehari-hari dilakukan dengan amat mudah menggunakan komunikasi digital. Sedemikian luasnya penggunaan media digital dalam komunikasi, teknologi Steganografi menjadi sangat luas implementasinya. Bisa hanya untuk sekadar hobi berkomunikasi dengan rekan Anda tanpa diketahui orang lain, bisa juga teknologi ini digunakan untuk hal-hal yang buruk seperti pencurian informasi penelitian, strategi perusahaan, penyelundupan informasi lokasi logistik militer, dan sebagainya.

Semua informasi genting tersebut bisa dengan mudah dicuri dan dikirimkan ke pihak-pihak yang tidak berwenang tanpa terlihat dan terdeteksi oleh mata kita. Untuk itulah, mengapa teknologi Steganografi sangat penting untuk Anda ketahui (*Simon Batara, 2013*).

Terdapat beberapa tujuan steganografi :

1. Tujuan Baik

Banyak sekali yang dapat kita lakukan dengan memanfaatkan teknologi ini untuk tujuan baik atau tujuan yang tidak mencelakakan, menipu, dan membahayakan orang lain. Biasanya untuk tujuan bisnis seperti *Copyright*

protection, yang akan mencegah terjadinya pembajakan dari produk-produk digital seperti musik, *software*, dan banyak lagi. Digital *watermarking* atau penandaan tak kasat mata secara digital pada sebuah materi digital yang biasanya dilakukan untuk menandai kepemilikan sesuatu, perizinan sesuatu, dan banyak lagi.

2. Tujuan Netral

Penggunaan dengan tujuan netral biasanya dilakukan oleh para pehobi atau amatiran yang memerlukan Steganografi sekadar untuk kesenangan atau kepentingan lain yang tidak mengganggu. Selain itu, peneliti juga biasanya akan banyak menggunakan teknik Steganografi untuk berbagai keperluan seperti melindungi hasil penelitiannya, berkomunikasi dengan peneliti lain tentang hasil penelitian rahasia dan lainnya.

3. Tujuan Buruk

Yang paling ditakutkan adalah penggunaan Steganografi untuk tujuan jahat atau yang dapat mencelakai orang banyak. Untuk dapat mengetahui dan mencegah tujuan jahat ini, akan sangat sulit jika disertai dengan penggunaan Steganografi di dalamnya. Banyak kabar beredar di berbagai media bahwa Osama bin Laden menggunakan teknik Steganografi untuk berkomunikasi dengan antek-anteknya dalam usahanya melakukan penyerangan terhadap gedung WTC (*World Trade Center*) dan Pentagon beberapa waktu lalu. Dari contoh ini, tentunya dapat disimpulkan bahwa kejahatan mungkin akan aman terlindung dengan Steganografi. Keamanan *Firewall* jenis apapun, konfigurasi *filtering* secanggih apapun, sistem *Intrusion Detection* sehebat apapun, tidak akan dapat mengetahui

dan mencegah informasi jahat ini untuk keluar masuk. Untuk itu, sebaiknya hindarilah penggunaan teknologi ini untuk kepentingan yang dapat mencelakakan orang lain. Selain itu, pornografi juga merupakan ladang yang nyaman untuk disusupi teknologi steganografi.(*Aminah Rizki Lubis, 2012*).

II.1.2 Kriteria Steganografi

Steganografi juga memiliki kelemahan. Steganografi memerlukan banyak ruang untuk dapat menyembunyikan beberapa bit pesan. Akan tetapi, kelemahan ini sedikit demi sedikit dapat diatasi seiring dengan perkembangan teknik-teknik dalam melakukan steganografi. menyembunyikan pesan, ada beberapa kriteria yang harus dipenuhi :

1. *Impersepbility*

Keberadaan pesan tidak dapat dipersepsi oleh indrawi. Jika pesan disisipkan kedalam sebuah citra, citra yang telah disisipi pesan harus tidak dapat dibedakan dengan citra asli oleh mata. Begitu pula dengan suara, telinga harus dapat membedakan suara asli dan suara yang telah di sisipi pesan.

2. *Fidelity*

Mutu media penampung tidak berubah banyak akibat penyisipan. Perubahan yang terjadi harus tidak dapat dipersepsi oleh indrawi.

3. *Recovery*

Pesan yang disembunyikan harus dapat diungkap kembali. Tujuan steganografi adalah menyembunyikan informasi, maka sewaktu-waktu informasi

yang disembunyikan ini harus dapat diambil kembali untuk dapat digunakan lebih lanjut sesuai keperluan.(*Shinta Puspita Sari, 2012*)

II.1.3 Cara Kerja Steganografi

Secara garis besar teknik penyembunyian data dengan Steganografi adalah dengan cara menyisipkan sepotong demi sepotong informasi asli pada sebuah media, sehingga informasi tersebut tampak kalah dominan dengan media pelindungnya. Seperti contohnya pesan rahasia yang ditato dikepala pengantar pesan misalnya. Contoh sederhana lainnya, namun sangat umum dan efektif digunakan pada saat perang dunia kedua adalah teknik *null cipher*. *Null cipher* merupakan contoh sederhana dari teknik Steganografi modern (*Shinta Puspita Sari, 2012*).

II.1.4 Media Steganografi

File media merupakan komponen penting pada proses penyembunyian informasi. Dengan *file* yang terlihat sama sekali tidak mencurigakan, data yang sebenarnya akan tetap tidak terdeteksi dengan mata telanjang. Secara teori, semua *file* umum yang ada didalam komputer dapat digunakan sebagai media, seperti *file* gambar berformat JPG, GIF, BMP, atau didalam musik MP3, atau bahkan didalam sebuah film dengan format WAV atau AVI. Semua bisa dijadikan tempat bersembunyi, asalkan file tersebut memiliki *bit-bit* data yang redundan yang dapat dimodifikasi. *Bit-bit* data redundan maksudnya adalah *bit-bit* data yang merupakan *bit* ganda yang jika dimodifikasi, maka kualitas dan tampilan *file* yeng

sesungguhnya tidak akan terganggu banyak. *File-file* yang dapat disisipi juga tergantung pada aplikasi Steganografi apa yang digunakan. Kebanyakan aplikasi Steganografi memiliki teknik dan ciri khas sendiri-sendiri dalam menggunakan *file-file* yang menjadi media pelindungnya. Ada yang hanya bisa menggunakan *file* gambar, *file* musik, dan banyak lagi. Jadi tidak akan mudah bagi siapapun untuk mendeteksi informasi tersembunyi pada *file-file* umum tersebut. Biasanya, *file-file* tersebut umumnya tidak terganggu fungsinya dan kualitas tidak akan jauh berbeda dengan aslinya (*Aminah Rizki Lubis, 2012*).

II.1.5 Perbedaan Steganografi Dengan Enkripsi

Enkripsi juga merupakan salah satu teknik untuk menjaga kerahasiaan data penting. Data yang telah melalui proses enkripsi, tidak akan mudah untuk diketahui oleh orang lain. Namun, konsep antara teknik enkripsi dengan Steganografi tidaklah sama. Arti enkripsi secara umum adalah sebuah proses mengacak data secara sistematis dan berpola, yang kemudian tetap memungkinkan untuk disusun kembali menjadi bentuk aslinya (*Wasino, 2012*).

Data atau informasi yang diacak ini memang tidak terlihat bentuk aslinya lagi setelah melalui proses enkripsi. Namun secara fisiknya, data atau informasi ini masih dapat terlihat. Contoh kasusnya, misalnya ada sebuah file yang berisikan *password-password* penting sebuah rekening bank ingin dikirim melalui *e-mail* oleh pemiliknya keseseorang. Mengetahui akan bahayanya informasi ini jika sampai ketangan yang salah, sebelumnya *file* tersebut telah dienkrip dan diberi *password*. Ketika sampai disisi penerima, *file* tersebut didekrip dengan sebuah

aplikasi dekriptor yang memintanya untuk memasukan *password* yang telah disetujui Sebelumnya. Setelah berhasil, maka *file* tersebut telah berhasil sampai dari si pengirim ke si penerima (*Wasino, 2012*).

Namun ternyata komputer si penerima disadap dan dengan berbagai cara, *e-mail* tadi sampai keorang ketiga. Pihak ketiga tersebut dapat dengan leluasa melihat fisik dari *file* tersebut, ia dapat mengetahui bahwa *file* tersebut memang bermaksud untuk dikirimkan ke si penerima. Namun untuk membuka *file* tersebut, orang ketiga tadi tentu saja akan kesulitan karena selain dienkripsi, untuk didekrip juga membutuhkan sebuah *password* lagi (*Wasino, 2012*).

Dengan demikian untuk saat itu, informasi yang ada di dalamnya tetap aman sampai *file* tersebut berhasil didekrip dengan berbagai cara dan tipu muslihat. Jika pada contoh kasus diatas si pengirim menggunakan teknik Steganografi untuk mengirimkan file informasi *password*-nya tersebut, maka orang ketiga tadi sama sekali tidak akan mengetahui bahwa ada informasi *password* rekening bank yang dikirimkan ke si penerima dari hasil sadapannya tersebut. Karena secara fisik, *file* tersebut benar-benar tidak terlihat (*Wasino, 2012*).

Steganografi agak berbeda dengan *watermarking*. *Watermarking* merupakan aplikasi dari steganografi. Jika pada steganografi informasi rahasia disembunyikan dalam media digital dimana media digital tidak berarti apa-apa, maka pada *watermarking* justru media digital tersebut yang akan dilindungi kepemilikannya dengan pemberian label hak cipta (*watermark*). Meskipun

steganografi dan *watermarking* tidak sama, namun secara prinsip proses penyisipan informasi kedalam data digital tidak jauh berbeda (*Wasino, 2012*).

II.2. Kriptografi

Kriptografi adalah ilmu yang mempelajari tentang cara menjaga keamanan suatu pesan atau informasi. Pesan atau informasi dapat dikategorikan kedalam dua jenis, yaitu pesan yang dapat dibaca dengan mudah *(plaintext)* dan pesan yang tidak mudah dibaca *(ciphertext).* Untuk melakukan kriptografi digunakan algoritma kriptografi. Algoritma kriptografi terdiri dari dua bagian, yaitu fungsi enkripsi dan dekripsi. Enkripsi adalah proses untuk mengubah *plaintext* menjadi *ciphertext*, sedangkan dekripsi adalah kebalikannya yaitu mengubah *ciphertext* menjadi *plaintext*. Salah satu teknik enkripsi adalah teknik subtitusi, yaitu mengganti setiap karakter *plaintext* dengan karakter lain (Ardiyanto, 2011). Terdapat empat cara dalam menggunakan teknik subtitusi, yaitu :

1. Monoalphabet, dimana setiap karakter *ciphertext* mengganti satu macam karakter *plaintext* tertentu.
2. Polialphabet, dimana setiap karakter *ciphertext* mengganti lebih dari satu macam karakter *plaintext*.
3. Monograf/unilateral, dimana satu enkripsi dilakukan terhadap satu karakter *plaintext*. Monograf/unilateral, dimana satu enkripsi dilakukan terhadap satu karakter *plaintext*.
4. Poligraf/multilateral, dimana satu enkripsi dilakukan terhadap lebih dari satu karakter *plaintext*.

II.2.1 Tujuan Kriptografi

Menurut Stalling, ada beberapa tuntutan yang terkait dengan isu keamanan data yaitu :

1. *Confidentiality*

Menjamin bahwa data-data tersebut hanya bisa diakses oleh pihak-pihak tertentu saja.

2. *Authentication*

Baik pada saat mengirim atau menerima informasi, kedua belah pihak perlu mengetahui bahwa pengirim dari pesan tersebut adalah orang yang sebenarnya seperti yang diklaim.

3. *Integrity*

Tuntutan ini berhubungan dengan jaminan setiap pesan yang dikirim pasti sampai pada penerimanya tanpa ada bagian dari pesan tersebut yang diganti, diduplikasi, dirusak, diubah urutannya, dan ditambahkan.

4. *Nonrepudiation*

Nonrepudiation mencegah pengirim maupun penerima mengingkari bahwa mereka telah mengirimkan atau menerima suatu pesan/informasi. Jika sebuah pesan dikirim, penerima dapat membuktikan bahwa pesan tersebut memang dikirim oleh pengirim yang tertera. Sebaliknya, jika sebuah pesan diterima, pengirim dapat membuktikan bahwa pesannya telah diterima oleh pihak yang ditujunya.

5. *Access Control*

Membatasi sumber-sumber data hanya kepada orang-orang tertentu.

6. *Availability*

Jika diperlukan setiap saat semua informasi pada sistem komputer harus tersedia bagi semua pihak yang berhak atas informasi tersebut (Ardiyanto, 2011).

Dari keenam aspek keamanan data tersebut, empat diantaranya dapat diatasi dengan menggunakan kriptografi yaitu *confidentiality*, *integrity*, *authentication*, dan *nonrepudiation.*

II.2.2 Jenis Sistem Kriptografi

Berdasarkan pemakaian kunci maka sistem kriptografi (c*ryptosystems*) dapat digolongkan atas 2 jenis sistem yakni sistem kriptografi kunci publik (*public key cryptography*) dan sistem kriptografi kunci rahasia (*secret key cryptography*). Dalam sistem kriptografi kunci rahasia yang dikenal juga dengan *symmetric cryptosystems*, pihak pengirim dan penerima bersama-sama menyepakati sebuah kunci rahasia yang akan digunakan dalam proses enkripsi dan dekripsi tanpa diketahui oleh pihak lain. Sedangkan dalam sistem kriptografi kunci publik atau dikenal dengan *assymmetric cryptosystem*, pihak pengirim maupun pihak penerima mendapatkan sepasang kunci yakni kunci dan kunci rahasia dimana kunci publik dipublikasikan dan kunci rahasia tetap dirahasiakan. Enkripsi dilakukan dengan menggunakan kunci publik sedangkan dekripsi dilakukan dengan menggunakan kunci rahasia.

II.2.3 Kriptografi Kunci Rahasia (*Secret Key Cryptography*)

Kriptografi kunci rahasia merupakan bentuk kriptografi yang lebih tradisional, dimana kunci tunggal dapat digunakan untuk enkripsi dan dekripsi suatu pesan. Kriptografi kunci rahasia tidak hanya digunakan untuk enkripsi, tetapi juga untuk otentikasi. Salah satu teknik untuk pekerjaan ini disebut *message authentication codes* (MAC).

Masalah utama dengan kriptografi kunci rahasia membuat pengirim dan penerima pesan setuju atas kunci rahasia yang digunakan tanpa orang lain mampu mendapatkan dan mengetahuinya. Atau dengan kata lain bagaimana memilih kunci rahasia yang benar-benar aman. Hal ini membutuhkan suatu metode dimana kedua pihak dapat berkomunikasi tanpa kekhawatiran akan tercecernya kunci tersebut. Akan tetapi, keuntungan dari kriptografi kunci rahasia adalah biasanya lebih cepat dibandingkan dengan kriptografi kunci publik. Metode yang paling umum untuk kriptografi kunci rahasia adalah *block ciphers*, *stream ciphers*, dan *message authentication codes* (MAC).

II.3. *Caesar Cipher*

Teknik enkripsi substitusi yang pertama kali dikenal dan paling sederhana ditemukan oleh *Julius Caesar*. Metode yang digunakan dalam *caesar chiper* ini adalah dengan mempertukarkan setiap huruf dari *plaintext* dengan huruf lain dengan interval 3 huruf dari huruf *plaintext* (Ardiyanto, 2011). Sebagai contoh dapat dilihat berikut ini :

A	B	C	D	E	F	G	H	I	J	K	L	M	N	P	O	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Gambar II.1. Tabel *Caesar Cipher*
Sumber : Ardiyanto, 2011

Untuk menyandikan sebuah pesan, cukup mencari setiap huruf yang hendak disandikan dialfabet biasa, lalu tuliskan huruf yang sesuai pada alfabet sandi. Untuk memecahkan sandi tersebut gunakan cara sebaliknya. Contoh penyandian sebuah pesan adalah sebagai berikut.

teks terang: kirim pasukan ke sayap kiri

teks tersandi: NLULP SDVXNDQ NH VDBDS NLUL

Proses penyandian (enkripsi) dapat secara matematis menggunakan operasi modulus dengan mengubah huruf-huruf menjadi angka, A = 1, B = 2,..., Z = 26. secara matematis dituliskan dengan,

$$C=E(P)=(P+K) \bmod (26)$$

Sedangkan pada proses pemecahan kode (dekripsi), hasil dekripsi adalah:

$$P=D(C)=(C-K) \bmod (26)$$

Setiap huruf yang sama digantikan oleh huruf yang sama disepanjang pesan, sehingga sandi *Caesar* digolongkan kepada substitusi *monoalfabetik*, yang berlawanan dengan *substitusi polialfabetik.*

Untuk menyandikan sebuah pesan, cukup mencari setiap huruf yang hendak disandikan, lalu tuliskan huruf yang sesuai pada sandi. Untuk memecahkan sandi tersebut gunakan cara sebaliknya. Contoh penyandian sebuah pesan adalah sebagai berikut:

Alfabet Biasa: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Alfabet Sandi: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Teks Asli : JANGAN MENDEKATI BLOK D

Teks Sandi : MDQJD QPHQG HNDWL EORN G

Untuk karakter ASCII dari huruf A sampai dengan Z, sebagai berikut:

Tabel II.1 Karakter ASCII

A	B	C	D	E	F	G	H	I	J	K	L	M
65	66	67	68	69	70	71	72	73	74	75	77	77
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
78	79	80	81	82	83	84	85	86	87	88	89	90

Sumber : Ardiyanto, 2011

II.4. *Pixel Value Differencing*

Pixel Value Differencing (PVD) skema menggunakan nilai perbedaan antara dua piksel berturut-turut diblok untuk menentukan berapa banyak bit rahasia harus tertanam. Ada dua jenis tabel kisaran kuantisasi dalam metode *Wu* dan *Tasi* itu. Yang pertama didasarkan pada memilih lebar kisaran [8, 8, 16, 32, 64, 128], untuk menyediakan kapasitas yang besar. Yang kedua didasarkan pada memilih lebar kisaran [2, 2, 4, 4, 4, 8, 8, 16, 16, 32, 32, 64, 64], untuk memberikan *imperceptibility* tinggi. Sebagian besar penelitian terkait fokus pada peningkatan kapasitas menggunakan LSB dan proses penyesuaian, sehingga pendekatan mereka terlalu Selaras dengan LSB pendekatan. Ada sangat sedikit penelitian yang berfokus pada desain meja jangkauan. Selain itu, intuitif untuk merancang dengan menggunakan lebar kekuatan dua. Karya ini desain tabel kisaran kuantisasi baru berdasarkan jumlah persegi yang sempurna untuk memutuskan *payload* dengan nilai perbedaan antara piksel berturut-turut.

Penelitian kami memberikan sudut pandang baru bahwa jika kita memilih lebar yang tepat untuk setiap rentang dan menggunakan metode yang diusulkan, kita bisa memperoleh jumlah gambar yang lebih baik dan kapasitas yang lebih tinggi. Selain itu, kami menawarkan analisis teoritis untuk menunjukkan metode kami didefinisikan dengan baik. Hasil penelitian juga menunjukkan skema yang diusulkan memiliki jumlah gambar yang lebih baik dan kapasitas yang lebih tinggi *(J.K Mandal, Journal of Applied Mathematics, Volume 2013 (2013), Article ID 189706).*

Proses penyisipan pada metode ini dilakukan dengan cara membandingkan dua *pixel* yang bertetangga P_i dan P_{i+1} dengan menggunakan persamaan (1).

$$d= |P_i – P_{i+1}|\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)$$

Hasil dari perbandingan tersebut digunakan untuk mengetahui berapa banyak *bit* yang dapat disisipkan kedalam dua *pixel* yang dibandingkan. Metode ini menggunakan skema *Wu* dan *Tsai* untuk mengetahui *range* dari perbandingan *pixel* sebelumnya. Skema *Wu* dan *Tsai* yang digunakan yaitu R ={[0,7],[8,15],[16,31],[32,63],[64,127],[128,255]}. Skema ini digunakan untuk mengetahui terdapat di *range* mana selisih dari dua *pixel* tersebut, jika telah diketahui dimana letak *range* nya, maka jumlah *bit* pesan yang disisipkan dapat diketahui dengan persamaan (2).

$$t = | \mathrm{Log}_2 w_i|\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

w_i : Nilai terkecil dari skema *wu* dan *tsai,* letak *range* selisih perbandingan dua *pixel*. Penyisipan pesan dapat dilakukan dengan mengambil sebanyak t *bit* dari

pesan yang akan disisipkan. Selanjutnya dihitung nilai *difference value* yang baru untuk penyisipan kedalam citra menggunakan persamaan (3).

$$d'_i = l_i + b \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

d_i : Nilai terkecil dari skema *wu* dan *tsai*, letak *range* selisih perbandingan dua *pixel*. Untuk menyisipkan pesan ada beberapa aturan yang harus dipenuhi yaitu :

1. Jika $P_i \geq P_{i+1}$ dan $d'_i > d_i$, maka$(P_i + |m/2|, P_{i+1} - |m/2|)$
2. Jika $P_i < P_{i+1}$ dan $d'_i > d_i$, maka$(P_i - |m/2|, P_{i+1} + |m/2|)$
3. Jika $P_i \geq P_{i+1}$ dan $d'_i \leq d_i$, maka$(P_i - |m/2|, P_{i+1} + |m/2|)$
4. Jika $P_i < P_{i+1}$ dan $d'_i \leq d_i$, maka$(P_i + |m/2|, P_{i+1} - |m/2|)$

Dimana m didapat dari selisih d'_i dengan d_i dengan menggunakan persamaan (4).

$$m = | d'_i - d_i| \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..(4)$$

Proses-proses tersebut dilakukan terus hingga *bit* pesan tersisipi semuanya kedalam citra. Proses ekstraksi pesan dari citra *stego* menggunakan metode ini dimulai dengan menghitung nilai *difference value* (d_i) antara dua *pixel* yang bertetangga. Nilai *difference value* tersebut digunakan untuk mengetahui nilai *continuous ranges* (R) yang sudah didefinisikan menggunakan skema *wu* dan *tsai*.

II.5 *Unified Modelling Language (UML)*

Unified Modeling Language (UML) merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah. Bahasa Pemodelan UML lebih cocok untuk pembuatan perangkat lunak dalam

bahasa pemrograman berorientasi objek (C++, Java, VB.NET), namun demikian tetap dapat digunakan pada bahasa pemrograman prosedural (Ardiyanto, 2011).

UML merupakan pemodelan objek yang fokus pada pendefinisian struktur statis dan model sistem informasi yang dinamis daripada mendefinisikan data dan model proses yang tujuannya adalah pengembangan tradisional. UML menawarkan diagram yang dikelompokan menjadi lima perspektif berbeda untuk memodelkan suatu sistem. Adapun daigram yang terdapat dalam uml adalah sebagai berikut *(*Ardiyanto, 2011*)*:

1. *Use Case Diagram*

Use Case Diagram secara grafis menggambarkan interaksi antara sistem, sistem eksternal, dan pengguna. Dengan kata lain *Use Case* diagram secara grafis mendeskripsikan siapa yang akan menggunakan sistem dan dalam cara apa pengguna (*user*) mengharapkan interaksi dengan sistem itu. *Use Case* secara naratif digunakan untuk secara tekstual menggambarkan sekuensi langkah-langkah dari setiap interaksi.

2. Diagram Struktur Statis

UML menawarkan dua diagram untuk memodelkan struktur statis sistem informasi, yaitu:

a. *Class Diagram*

Class Diagram menggambarkan struktur *object* sistem. Diagram ini menunjukan *class object* yang menyusun sistem dan juga hubungan antara *class object* tersebut.

b. *Object Diagram*

Object diagram hamper memiliki kemiripan dengan *class diagram*, tetapi *object diagram* memodelkan *isntance object actual* dengan menunjukan nilai-nilai saat ini dari atribut *instance*. *Object Diagram* menyajikan "*snapshot*/potret" tentang objek sistem pada point waktu tertentu. Diagram ini tidak digunakan sesering *Class Diagram*, tetapi saat digunakan dapat membantu seorang developer memahami struktur sistem secara lebih baik.

3. Diagram Interaksi

Diagram interaksi memodelkan sebuah interaksi, terdiri dari satu set objek, hubungan-hubungannya, dan pesan yang terkirim di antara objek. Model diagram ini memodelkan *behavior* (kelakuan) sistem yang dinamis dan UML memiliki dua diagram untuk tujuan ini, yaitu:

a. Diagram rangkaian/*Sequence Diagram*

Objek ini secara grafis menggambarkan bagaimana objek berinteraksi dengan satu sama lain melalui pesan pada sekuensi sebuah *use case* atau operasi. Diagram ini mengilustrasikan bagaimana pesan terkirim dan diterima diantara objek dan dalam sekuensi atau *timing*.

b. Diagram kolaborasi/*Collaboration Diagram*

Diagram ini menggambarkan interaksi (atau kolaborasi) antara objek dalam sebuah format jaringan. Diagram rangkaian maupun diagram kolaborasi merupakan *isomorphic* artinya kita dapat mengubah dari satu diagram ke diagram lain.

4. Diagram State/*State Diagram*

UML memiliki sebuah diagram untuk memodelkan *behavior* objek khusus yang kompleks (*statecahrt*) dan sebuah diagram untuk memodelkan *behavior* dari sebuah *use case* atau sebuah metode, yaitu:

a. *Diagram statechart*

Komponen ini memodelkan *behavior* objek khusus yang dinamis. Diagram ini mengilustrasikan siklus hidup objek berbagai keadaan yang dapat diasumsikan oleh objek dan *event-event* (kejadian) yang menyebabkan objek beralih dari satu *state* ke *state* lain.

b. Diagram aktivitas/*Activity Diagram*

Activity diagram dapat juga digunakan untuk memodelkan *action* yang akan dilakukan saat sebuah operasi dieksekusi, dan memodelkan hasil dari *action* tersebut.

5. Diagram Implementasi

Diagram implementasi juga memodelkan struktur sistem informasi, yaitu:

a. Diagram komponen/*Component Diagram*

Digunakan untuk menggambarkan organisasi dan ketergantungan komponen-komponen *software* sistem. Komponen diagram dapat digunakan untuk menunjukan bagaimana kode pemrograman dibagi menjadi modul-modul (atau komponen).

b. Diagram penguraian/*Deployment*

Digunakan untuk mendeskripsikan arsitektur fisik dalam istilah ”node” untuk *hardware* dan s*oftware* dalam sistem. Diagram ini menggambarkan

konfigurasi komponen-komponen *software real-time,* prosesor, dan peralatan yang membentuk arsitektur sistem.

UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan bahasa pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam mengkomunikasikan beberapa aspek dalam sistem melalui sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. UML mempunyai banyak diagram yang dapat mengakomodasi berbagai sudut pandang dari suatu perangkat lunak yang akan dibangun. Diagram-diagram tersebut digunakan untuk (Ardiyanto, 2011):

1. Mengkomunikasikan ide.
2. Melahirkan ide-ide baru dan peluang-peluang baru.
3. Menguji ide dan membuat prediksi.
4. Memahami struktur dan relasi-relasinya.

Berdasarkan uraian diatas maka penulis membuat sebuah alur sistem yang ditampilkan dalam bentuk *Use Case diagram*, *Activity diagram*, dan *Class diagram* dalam model *Unified Modelling Language* (UML).

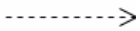
II.5.1 *Use Case Diagram*

Use Case diagram adalah model fungsional sebuah sistem yang menggunakan *actor* dan *use case*. *Use Case* adalah layanan (*services)* atau fungsi-fungsi yang disediakan oleh sistem untuk pengguna-penggunanya (Ardiyanto,

2011). *Use Case* adalah suatu pola atau gambaran yang menunjukan kelakukan atau kebiasaan sistem. Setiap *Use Case* adalah suatu urut-urutan (*sequence*) transaksi yang saling berhubungan dan dilakukan oleh sebuah *actor* dan sistem dalam bentuk sebuah dialog.

Simbol-simbol yang digunakan dalam *Use Case diagram* adalah sebagai berikut :

Tabel II.2 Daftar simbol dalam *Use Case Diagram*

GAMBAR	NAMA	KETERANGAN
	*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
	Extend	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
	*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
	*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
	*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Sumber : Ardiyanto, 2011

II.5.2 *Activity Diagram*

Activity diagram adalah diagram yang menggambarkan sifat dinamis secara alamiah sebuah sistem dalam bentuk model aliran dan kontrol dari aktivitas ke aktivitas lainnya (Ardiyanto, 2011). Sebuah aktivitas merepresentasikan suatu operasi pada beberapa *class* dalam sistem yang menghasilkan suatu perubahan keadaan (*state*) dari sistem tersebut. Secara khusus, *activity diagram* biasa digunakan untuk memodelkan diagram alir sebuah sistem kerja (*workflow*) atau proses bisnis dan operasi-operasi secara internal. Simbol-simbol yang digunakan dalam *Activity diagram* adalah sebagai berikut :

Tabel II.3 Daftar simbol dalam *Activity Diagram*

GAMBAR	NAMA	KETERANGAN
	*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.
	*Initial Node*	Bagaimana objek dibentuk atau diawali.
	*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan.
	*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran.

Sumber : Ardiyanto, 2011

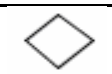
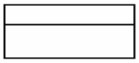
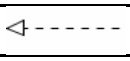
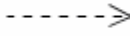
II.5.3 *Class diagram*

Class adalah kumpulan objek-objek dengan dan yang mempunyai struktur umum, *behavior* umum, relasi umum, dan *semantic*/kata yang umum. *Class-class* ditentukan/ditemukan dengan cara memeriksa objek-objek dalam *sequence*

diagram dan *collaboration diagram*. Sebuah *class* digambarkan seperti sebuah bujur sangkar dengan tiga bagian ruangan.

Class diagram adalah diagram yang menunjukan *class-class* yang ada dari sebuah sistem dan hubungannya secara logika. *Class diagram* menggambarkan struktur statis dari sebuah sistem. Karena itu *class diagram* merupakan tulang punggung atau kekuatan dasar dari hampir setiap metode berorientasi objek termasuk UML (Ardiyanto, 2011).

Tabel II.4 Daftar simbol dalam *Class Diagram*

GAMBAR	NAMA	KETERANGAN
	Generalization	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*N-Ary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
	*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
	*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	Association	Apa yang menghubungkan antara objek satu dengan objek lainnya.

Sumber : Ardiyanto, 2011

II.6. *Borland Delphi 7*

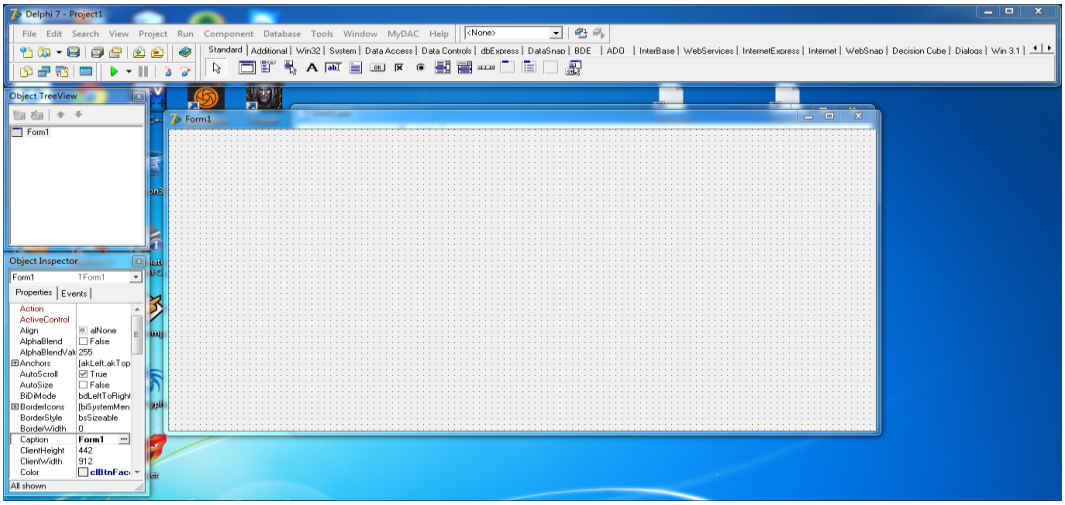
B*orland Delphi 7.0 (Delphi),* adalah paket bahasa pemrograman yang bekerja dalam sistem operasi *Windows*. *Delphi* merupakan bahasa pemrograman yang mempunyai cakupan kemampuan yang luas dan sangat canggih. Berbagai aplikasi dapat

dibuat dengan *Delphi*, termasuk aplikasi untuk pengolah teks, grafik, angka, *database* dan aplikasi *web* (Leon Andreti Abdillah, 2006).

Delphi menyediakan fasilitas yang lengkap untuk mengelolah *database*. Berbagai format *database* dapat diolah dengan *Delphi*, misalnya *database* dengan format *Paradox, dBase, MS-Access, ODBC, SyBASE, Oracle* dan lain-lain. Sebelum *Delphi*, *Borland* sudah lama mengeluarkan program untuk manajemen *database* yang sangat terkenal, yaitu program *Paradox*. Dengan *Delphi*, kemampuan yang ada pada program *Paradox* menjadi lebih baik dan makin sempurna. Dalam format *Paradox*, satu *file database* hanya berisi satu tabel *database*. Jadi berbeda dengan format *Microsoft Access*, yang memungkinkan membuat beberapa tabel dalam satu *file database* (Abdul Ghofur, 2010).

II.6.1 *Integrated Development Environment*

Tampilan bidang kerja yang disebut dengan IDE (*Integrated Development Environment*) *Delphi 7* bisa dilihat pada gambar II.2 dibawah ini. IDE ini secara garis besar terdiri atas tiga bagian utama, yaitu *Window* Utama, *Object Inspector* dan *Editor*.



Gambar II.2 Tampilan *IDE Delphi 7*
Sumber : Abdul Ghofur, 2010

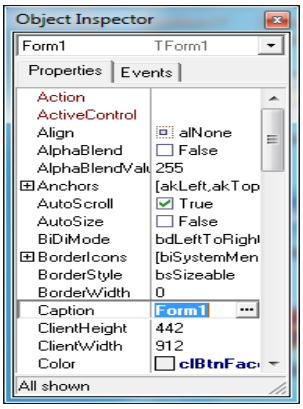
II.6.2 *Main menu, Speed Bar* dan *Palette Bar*

Didalam *main menu*, terdapat fasilitas-fasilitas yang disediakan oleh *Delphi* untuk mengontrol serta menangani masalah yang berhubungan dengan pekerjaan kita (contohnya menu untuk menyimpan, menu untuk meng-*compile* dan masih banyak lagi). Kegunaan *speed bar seperti main menu*, tapi di *speed bar* berupa *icon* sehingga kita cukup mengklik saja untuk mrnjalankan fasilitas yang kita kehendaki. *Palette bar* adalah tempat *object-object* yang kita gunakan untuk menyusun sebuah program.

Gambar II.3 *Main Menu, Speed bar dan Palette Bar*
Sumber : Abdul Kadir2009

II.6.3 *Object Inspector*

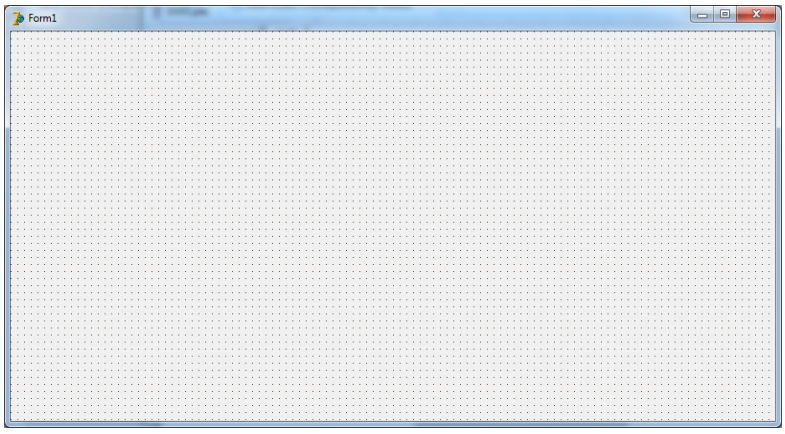
Object inspector adalah sarana pengaturan objek yang kita pasangkan pada *form*, atau *form* itu sendiri. Dua hal penting yang bisa kita setel pada komponen adalah Properti dan *event*. Properti adalah yang terkait dengan sifat komponen seperti ukuran, warna dan sebagainya. Sedangkan *event* adalah kejadian atau peristiwa yang kita inginkan terpasang pada komponen tersebut kaitannya dengan proses pemakaian. Contoh *event* misalnya klik, klik ganda, *drag* (geser), *drop* dan sebagainya.



Gambar II.4 *Object Inspector*
Sumber :Abdul Kadir, 2009

II.6.4 *Form*

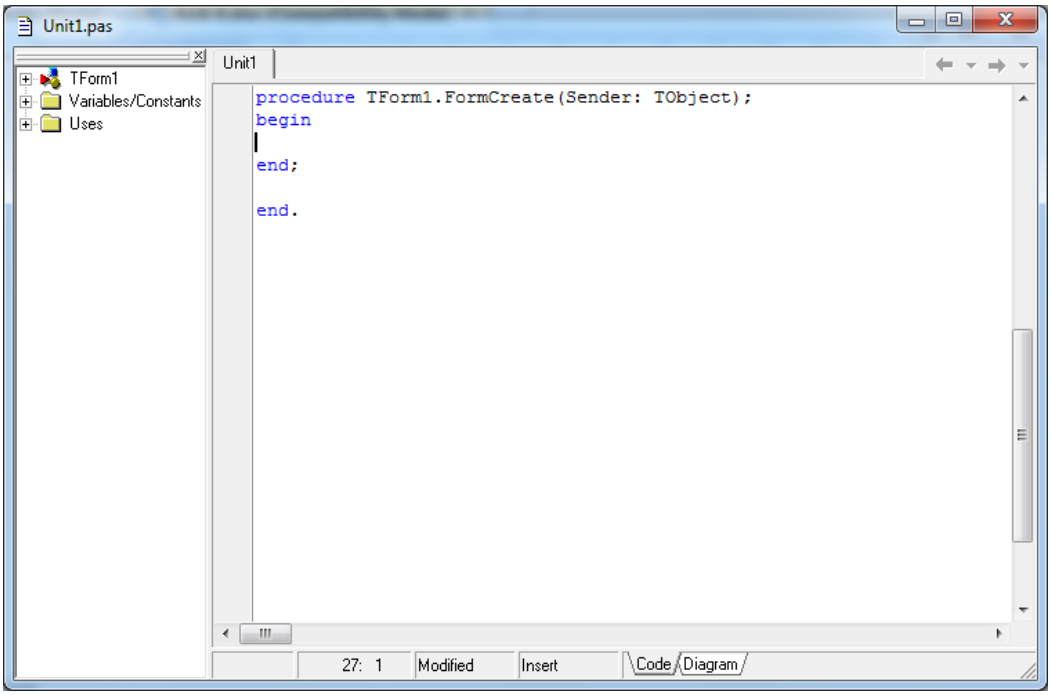
Form adalah bahan dasar yang akan menjadi jendela aplikasi kita. Pada *form* terdapat tiga tombol kontrol, yaitu *Minimize*, *Maximize*/*Restore* dan *Close*. Terdapat juga *caption bar* tempat kita menempatkan judul *Form* (yang kelak menjadi judul *window*) dan *icon*. Pembatas form juga bisa diubah ukurannya dengan cara *drag and drop*. Pada *form* kita bisa meletakkan komponen-komponen yang kita perlukan dalam suatu *User Interface*.



Gambar II.5 *Form*
Sumber : Abdul Kadir, 2009

II.6.5 *Code Editor*

Code Editor adalah tempat kita menuliskan program dalam bahasa *Object Pascal*. Secara *default Code Editor* ini terletak dibelakang *Form Editor*.



Gambar II.6 *Code Editor*
Sumber : Abdul Kadir, 2009