BAB 2
TINJAUAN PUSTAKA

2.1 Aplikasi

Aplikasi berasal dari kata *application* yang artinya penerapan; lamaran; penggunaan. Secara istilah aplikasi adalah program siap pakai yang dibuat untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju (www.totalinfo.or.id). Pengertian aplikasi menurut Yan Tirtobisono (1999:21) adalah istilah yang digunakan untuk pengguna komputer bagi pemecahan masalah. Biasanya istilah aplikasi dipasangkan atau digabungkan dalam suatu perangkat lunak sehingga akan dapat memberikan makna atau arti baru yaitu suatu program yang ditulis atau dibuat untuk menangani masalah tertentu.

2.2 Sistem Informasi

Sistem Informasi adalah kombinasi dari teknologi informasi dan aktivitas orang yang menggunakan teknologi itu untuk mendukung operasi dan manajemen. Dalam arti yang sangat luas, istilah sistem informasi yang sering digunakan merujuk kepada interaksi antara orang, proses algoritmik, data, dan teknologi [20]. Sistem berarti kumpulan dari metode, prosedur, rutinitas yang diformulasikan untuk melakukan aktifitas, tugas atau memecahkan masalah. Sedangkan Informasi tercipta dari pengolahan berbagai data baik dalam bentuk teks, gambar, *audio*, *video* dan lain sebagainya. Jadi Informasi diperoleh dari hasil pengolahan data.

2.3 Keamanan Sistem Informasi

Masalah keamanan merupakan salah satu aspek penting dari sebuah sistem informasi. Namun, masalah keamanan ini sering kali kurang mendapat perhatian. Seringkali masalah keamanan berada di urutan kedua, atau bahkan di urutan terakhir dalam daftar hal-hal yang dianggap penting. Apabila menggangu performansi dari sistem, seringkali keamanan dikurangi

atau ditiadakan. Informasi saat ini sudah menjadi sebuah komoditi yang sangat penting. Bahkan ada yang mengatakan bahwa kita sudah berada di sebuah "*information-based society*". Kemampuan untuk mengakses dan menyediakan informasi secara cepat dan akurat menjadi sangat esensial bagi sebuah organisasi, baik yang berupa organisasi komersial (perusahaan), perguruan tinggi, lembaga pemerintahan, maupun individual (pribadi). Sangat pentingnya nilai sebuah informasi menyebabkan seringkali informasi diinginkan hanya boleh diakses oleh orang-orang tertentu. Jatuhnya informasi ke tangan pihak lain dapat menimbulkan kerugian bagi pemilik informasi.[1]

Garfinkel [13] mengemukakan bahwa keamanan komputer (*computer security*) melingkupi empat aspek, yaitu *privacy, integrity, authentication*, dan *availability*.

Ada empat aspek utama dalam keamanan data dan informasi.

1. *Privacy/Confidentiality* yaitu usaha menjaga data informasi dari orang yang tidak berhak mengakses (memastikan bahwa data atau informasi pribadi kita tetap pribadi).
2. *Integrity* yaitu usaha untuk menjaga data atau informasi tidak boleh diubah tanpa seijin pemilik informasi.
3. *Authentication* yaitu usaha atau metoda untuk menyatakan bahwa informasi betul-betul asli, orang yang mengakses atau memberikan informasi adalah betul-betul orang yang dimaksud, atau *server* yang kita hubungi adalah betul-betul *server* yang asli.
4. *Availability* berhubungan dengan ketersediaan sistem dan data (informasi) ketika dibutuhkan.

Pada zaman teknologi informasi sekarang, data atau informasi merupakan suatu asset yang sangat berharga dan harus dilindungi. Hal ini juga diikuti oleh kemajuan teknologi komputer. Kemajuan teknologi komputer membantu semua aspek kehidupan manusia. Dengan adanya kemajuan dalam teknologi informasi, komunikasi dan komputer maka kemudian muncul masalah baru, yaitu masalah keamanan akan data dan

informasi dan dalam hal ini akan membuka peluang bagi orang-orang yang tidak bertanggung jawab untuk menggunakannya sebagai tindak kejahatan. Dan tentunya akan merugikan pihak tertentu. Agar data atau informasi penting tidak jatuh ke tangan orang yang bertanggung jawab maka perlu dilakukan teknik untuk mengamankan data atau informasi tersebut. Teknik pengamanan yang bisa kita pakai yaitu steganografi dan kriptografi.

2.4 Citra Digital

Secara umum, pengolahan citra digital menunjukan pada pemrosesan gambar dua dimensi menggunakan komputer. Dalam konteks lebih luas, pengolahan citra digital mengacu pada pemrosesan setiap data dua dimensi. Citra digital merupakan sebuah larik (*array*) yang berisi nilai-nilai real maupun komplek yang direpresentasikan dengan deret bit tertentu.

Suatu citra dapat didefinisikan sebagai fungsi (*x*,*y*) berukuran M baris dan N kolom, dengan x dan y adalah koordinat spasial, dan amplitudo *f* di titik koordinat (*x*,*y*) dinamakan intensitas atau tingkat keabuan dari citra pada titik tersebut. Apabila nilai x, y, dan nilai amplitudo f secara keseluruhan berhingga (*finite*) dan bernilai diskrit maka dapat dikatakan bahwa citra tersebut adalah citra digital. Citra digital dinyatakan dengan matriks berukuran N x M (baris/tinggi = N, kolom/lebar = M) [14]

N = jumlah baris $0 \leq y \leq N - 1$

M = jumlah kolom $0 \leq x \leq M - 1$

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \dots & f(1,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix}$$

Gambar 2.1 Matriks Citra Digital

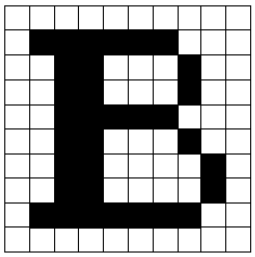
Citra digital dibentuk dari blok-blok kecil yang dinamakan piksel. Piksel merupakan elemen penyusun warna terkecil yang menyusun suatu citra. Untuk menunjukkan tingkat pencahayaan piksel, digunakan bilangan bulat yang besarnya 8 *bit* (1 *byte*) untuk setiap pikselnya, dengan lebar selang antara 0-255, dimana 0 untuk warna hitam, 255 untuk warna putih,

dan tingkat keabuan ditandai dengan nilai diantara 0-255.

Citra digital dapat dikualifikasikan berdasarkan sifat dari nilai terkuantisasinya sebagai berikut :

a. Citra Biner

Citra biner (*binary image*) adalah citra yang hanya mempunyai dua nilai derajat keabuan: hitam dan putih. *Pixel-pixel* objek bernilai 1 dan *pixel-pixel* latar belakang bernilai 0. Pada waktu menampilkan gambar, 0 adalah putih dan 1 adalah hitam. Jadi, pada citra biner, latar belakang berwarna putih sedangkan objek berwarna hitam. [15]

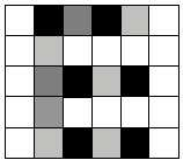


0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	1	1	1	0	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	0	0	0	0	1	0
0	0	1	1	0	0	0	0	1	0
0	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0

Gambar 2.2 Citra Biner dan Representasi Citra Biner

b. Citra *Grayscale*

Citra *grayscale* disebut juga dengan citra 8-*bit* karena memiliki 2^8 (256) kemungkinan nilai pada masing-masing pikselnya.Nilai tersebut dimulai dari nol untuk warna hitam dan 255 untuk warna putih.Citra ini disebut juga citra hitam putih atau citra *monoakromatik.* Skala keabuan 4 *bit* jumlah kemungkinan 2^4 = 16 warna, kemungkinan warna 0 (*min*) sampai 15 (*max*). Skala keabuan 8 *bit* jumlah kemungkinan 2^8 = 256 warna kemungkinan warna 0 (*min*) sampai 255 (*max*).



= 15 0 6 0 13 15
= 15 12 15 15 15 15
= 15 5 0 12 0 15
= 15 8 15 15 15 15
= 15 10 0 13 0 15

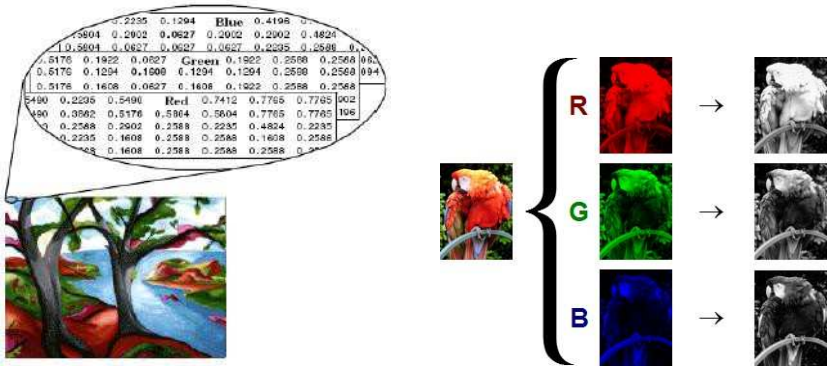
Gambar 2.3 Skala Keabuan 4 *bit*

c. Citra Warna

Citra warna adalah citra yang nilai piksel-nya merepresentasikan warna tertentu.Banyaknya warna yang mungkin digunakan bergantung

kepada kedalaman *pixel* citra yang bersangkutan. Citra berwarna direpresentasikan dalam beberapa kanal *(channel*) yang menyatakan komponen-komponen warna penyusunnya. Banyaknya kanal yang digunakan bergantung pada model warna yang digunakan pada citra tersebut.

Intensitas suatu pada titik pada citra berwarna merupakan kombinasi dari tiga intensitas : derajat keabuan (merah fmerah(x,y)), hijau fhijau(x,y) dan biru (fbiru(x,y)). Persepsi visual citra berwarna umumnya lebih kaya di bandingkan dengan citra hitam putih. Citra berwarna menampilkan objek seperti warna aslinya (meskipun tidak selalu tepat demikian).Warna-warna yang diterima oleh mata manusia merupakan hasil kombinasi cahaya dengan panjang gelombang berbeda.



Gambar 2.4 Citra Warna

2.4.2 Format Citra Digital

Macam – macam format pada citra digital :

1. BMP (*Bitmap Image*)

   Format file ini merupakan format grafis yang fleksibel untuk *platform Windows* sehingga dapat dibaca oleh program grafis manapun. Format ini mampu menyimpan informasi dengan kualitas tingkat 1 *bit* samapi 24 *bit*. Kelemahan format file ini adalah tidak mampu menyimpan *alpha channel* serta ada kendala dalam pertukaran *platform*. Untuk membuat sebuah objek sebagai *desktop wallpaper*, simpanlah dokumen Anda dengan format file ini. Anda dapat

mengkompres format file ini dengan kompresi RLE. Format file ini mampu menyimpan gambar dalam mode warna RGB, *Grayscale*, *Indexed Color*, dan *Bitmap.*

2. JPG/JPEG (*Joint Photographic Expert Group*)

   Format file ini mampu mengkompres objek dengan tingkat kualitas sesuai dengan pilihan yang disediakan. Format file sering dimanfaatkan untuk menyimpan gambar yang akan digunakan untuk keperluan halaman *web*, multimedia, dan publikasi elektronik lainnya. Format file ini mampu menyimpan gambar dengan mode warna RGB, CMYK, dan *Grayscale*. Format file ini juga mampu menyimpan *alpha channel*, namun karena orientasinya ke publikasi elektronik maka format ini berukuran relatif lebih kecil dibandingkan dengan format file lainnya.

3. GIF (*Graphic Interchange Format*)

   Format file ini hanya mampu menyimpan dalam 8 *bit* (hanya mendukung mode warna *Grayscale, Bitmap* dan *Indexed Color*). Format file ini merupakan format standar untuk publikasi elektronik dan internet. Format file mampu menyimpan animasi dua dimensi yang akan dipublikasikan pada internet, desain halaman *web* dan publikasi elektronik. Format file ini mampu mengkompres dengan ukuran kecil menggunakan kompresi LZW.

4. PNG (*Portable Network Graphic*)

   PNG (*Portable Network Graphics*) adalah salah satu format penyimpanan citra yang menggunakan metode pemadatan yang tidak menghilangkan bagian dari citra tersebut (*lossless compression*).

   Format PNG ini diperkenalkan untuk menggantikan format penyimpanan citra GIF. 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.

PNG (Format berkas grafik yang didukung oleh beberapa *web browser*. PNG mendukung transparansi gambar seperti GIF, berkas PNG bebas paten dan merupakan gambar *bit*map yang terkompresi. PNG diciptakan untuk menggantikan keberadaan GIF karena masalah lisensi. Format PNG lebih baik daripada GIF. Masalahnya ada pada kurangnya dukungan yang dimampukan oleh *web browser*. Format ini dibuat sebagai alternatif lain dari format GIF. Format ini digunakan untuk menyimpan berkas dengan kedalaman 24 *bit* serta memiliki kemampuan untuk menghasilkan *background* transparan dengan pinggiran yang halus.

Format PNG menggunakan metode kompresi *lossless* untuk menampilkan gambar 24-*bit* atau warna-warna solid pada media daring (*online*). Format ini mendukung transparansi di dalam *alpha channel*. Format PNG sangat baik digunakan pada dokumen daring (*online*), dan mempunyai dukungan warna yang lebih baik saat dicetak dari pada format GIF.

Pada dasarnya, format PNG bukan merupakan format baru karena telah dikembangkan pada tahun 1995 untuk mengganti format GIF dan format TIFF. Format ini tidak digunakan lagi secara luas oleh *browser* dan perangkat lunak aplikasi pengolah gambar, sehingga dukungan terhadap format tidak begitu besar hingga tahun 2003, di mana format PNG semakin dikenal dan dipergunakan untuk aplikasi manipulasi gambar.

Secara garis besar, format PNG mempunyai fitur sebagai berikut:

- Sebagai pengganti format GIF dan TIFF.

- Format terbuka atau *open*, efisien, gratis, dan kompresi jenis *lossless*.
- Tiga mode warna, yaiut : *paletted* (8 *bit*), *greyscale* (16 *bit*), *truecolour* (hingga 48 *bit*)
- Dukungan terhadap *profile colour*, *gamma*, dan *metadata*.
- Mempunyai fitur transparansi serta dukungan penuh terhadap *alpha channel*.
- Dukungan luas bagi *software* menipulasi grafis dan *web browser*.

2.5 Steganografi

Steganografi adalah suatu ilmu, teknik dan seni tentang bagaimana menyembunyikan data rahasia didalam wadah (media) digital sehingga keberadaan data rahasia tersebut tidak diketahui oleh orang lain. Steganografi membutuhkan dua properti, yaitu media penampung dan data rahasia yang akan disembunyikan. Steganografi digital menggunakan media digital sebagai wadah penampung, misalnya citra (gambar), suara (*audio*), teks, dan *video*. Data rahasia yang disembunyikan juga dapat berupa citra, suara, teks, atau *video,* Steganografi dapat dipandang sebagai kelanjutan kriptografi. Jika pada kriptografi data yang telah disandikan (*ciphertext*) tetap tersedia, maka dengan steganografi *ciphertext* dapat disembunyikan sehingga pihak ketiga tidak mengetahui keberadaannya. Data rahasia yang disembunyikan dapat diekstraksi kembali persis sama seperti keadaan aslinya.

Pada umumnya, pesan steganografi muncul dengan rupa lain seperti gambar, artikel, daftar belanjaan, atau pesan-pesan lainnya. Pesan yang tertulis ini merupakan tulisan yang menyelubungi atau menutupi. Contohnya, suatu pesan bisa disembunyikan dengan menggunakan tinta yang tidak terlihat di antara garis-garis yang kelihatan. Teknik steganografi meliputi banyak sekali metode komunikasi untuk menyembunyikan pesan rahasia (teks atau gambar) di dalam berkas-berkas lain yang mengandung

teks, *image*, bahkan *audio* tanpa menunjukkan ciri-ciri perubahan yang nyata atau terlihat dalam kualitas dan struktur dari berkas semula.

Tujuan dari steganografi adalah merahasiakan atau menyembunyikan keberadaan dari sebuah pesan tersembunyi atau sebuah informasi. Dalam prakteknya, kebanyakan pesan disembunyikan 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 ke dalam algoritma yang digunakan.Pada metode steganografi cara ini sangat berguna jika digunakan pada cara steganografi komputer karena banyak format berkas digital yang dapat dijadikan media untuk menyembunyikan pesan. Format yang biasa digunakan di antaranya:

1. Format *image* : *bit*map (bmp), gif, pcx, jpeg, dll.
2. Format *audio* : wav, voc, mp3, dll.
3. Format lain : teks *file*, html, pdf, dll.

Kelebihan steganografi jika dibandingkan dengan 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 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 *covertext* lainnya dapat dimanipulasi untuk membawa pesan tersembunyi; hanya penerima (yang harus mengetahui teknik yang digunakan) dapat membuka pesan dan mendekripsikannya.

2.5.1 Sejarah Steganografi

Sejarah steganografi cukup panjang. Awalnya adalah penggunaan *hieroglyphic* oleh bangsa Mesir, yakni menulis menggunakan karakter-karakter dalam wujud gambar. Tulisan Mesir kuno tersebut menjadi ide untuk membuat pesan rahasia saat ini. Oleh karena itulah, tulisan mesir kuno yang menggunakan gambar dianggap sebagai steganografi pertama di dunia.[5]

Menurut penelitian para ahli, Yunani termasuk bangsa yang menggunakan steganografi setelah bangsa Mesir. Herodotus mendokumentasikan konflik antara Persia dan Yunani pada abad ke-50 sebelum masehi. Dokumentasi pada masa Raja Xerxes, raja dari Persia, disimpan di Yunani menggunakan steganografi. Berikut adalah beberapa contoh penggunaan teknik steganografi klasik[6]:

1. Abad ke-15 orang Italia menggunakan tawas dan cuka untuk menulis pesan rahasia diatas kulit telur. Kemudian telur tersebut direbus hingga "tinta" yang ada meresap dan tidak terlihat pada kulit telur. Penerima pesan cukup mengupas kulit telur tersebut untuk membaca pesan.
2. Selama terjadinya Perang Dunia ke-2, tinta yang tidak tampak (*invisible ink*) telah digunakan untuk menulis informasi pada lembaran kertas sehingga saat kertas tersebut jatuh di tangan pihak lain hanya akan tampak seperti lembaran kertas kosong biasa.
3. Pada sejarah Yunani kuno, masyarakatnya biasa menggunakan seorang pembawa pesan sebagai perantara pengiriman pesan. Pengirim pesan tersebut akan dicukur rambutnya, untuk kemudian dituliskan suatu pesan pada kepalanya yang sudah botak. Setelah pesan dituliskan, pembawa pesan harus menunggu hingga rambutnya tumbuh kembali sebelum dapat mengirimkan pesan kepada pihak penerima. Pihak penerima kemudian akan mencukur rambut pembawa pesan tersebut untuk melihat pesan yang tersembunyi.
4. Metode lain yang digunakan oleh masyarakat Yunani kuno adalah dengan menggunakan lilin sebagai media penyembunyi pesan mereka.

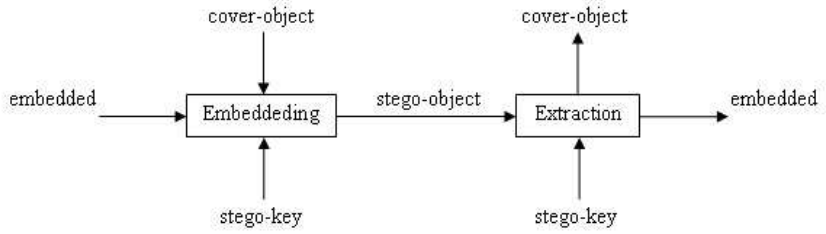
Pesan dituliskan pada suatu lembaran, dan lembaran tersebut akan ditutup dengan lilin untuk menyembunyikan pesan yang telah tertulis. Pihak penerima kemudian akan menghilangkan lilin dari lembaran tersebut untuk melihat pesan yang disampaikan oleh pihak pengirim.

2.5.2 Konsep dan Teriminologi Steganografi

Terdapat beberapa istilah yang berkaitan dengan steganografi [7], yaitu:

1. *Hiddentext* atau *embedded message*: pesan yang disembunyikan.
2. *Cover-object*: pesan yang digunakan untuk menyembunyikan *embedded message*.
3. *Stego-object*: pesan yang sudah berisi *embedded message*.

Di dalam steganografi digital, baik *embedded message* maupun *cover-object* dapat berupa teks, citra, *audio*, maupun *video*. Penyisipan pesan ke dalam media *cover-object* dinamakan *encoding*, sedangkan ekstraksi pesan dari *stego-object* dinamakan *decoding*. Kedua proses ini mungkin memerlukan kunci rahasia (*stegokey*) agar hanya pihak yang berhak saja yang dapat melakukan penyisipan pesan dan ekstraksi pesan sehingga menambah tingkat keamanan data. Proses umum penyisipan pesan dan ekstraksi pesan dapat dilihat pada Gambar 2.5



Gambar 2.5 Proses Umum Penyisipan dan Ekstrasi Pesan

2.5.3 Teknik Steganografi

Pada dasarnya, terdapat tujuh teknik yang digunakan dalam steganografi [5], yaitu:

1. *Injection*, merupakan suatu teknik menanamkan pesan rahasia secara langsung ke suatu media. Salah satu masalah dari teknik ini adalah ukuran media yang diinjeksi menjadi lebih besar dari ukuran normalnya sehingga mudah dideteksi.
2. Teknik substitusi (*Substitution Techniques*), pada teknik ini data asli digantikan dengan data rahasia. Biasanya, hasil teknik ini tidak terlalu mengubah ukuran data asli, tetapi tergantung pada file media dan data yang akan disembunyikan. Teknik substitusi ini bisa menurunkan kualitas media penampung.
3. Teknik Domain Transformasi (*Domain Transform Techniques*), yaitu dengan cara menyimpan informasi rahasia pada transformasi ruang (misalnya domain frekuensi) dari media penampung (*cover*). Akan lebih efektif jika teknik ini diterapkan pada *file* berekstensi Jpeg (gambar).
4. Teknik Spread Spectrum (*Spread Spectrum Techniques*), merupakan sebuah teknik pentransmisian menggunakan *pseudo-noise code*, yang independen terhadap data informasi sebagai modulataor bentuk gelombang untuk menyebarkan energi sinyal dalam sebuah jalur komunikasi (*bandwidth*) yang lebih besar dari pada sinyal jalur komunikasi informasi. Penerima mengumpulkan kembali sinyal dengan menggunakan *replica pseudo-noise code* tersinkronisasi.
5. Teknik Statistik (*Statistical Techniques*), dengan teknik ini data diencoding melalui pengubahan beberapa informasi statistik dari media penampung (*cover*). Media penampung di bagi dalam blok-blok dimana setiap blok tersebut menyimpan satu *pixel* informasi rahasia yang disembunyikan. Perubahan statistik ditunjukkan dengan indikasi 1 dan jika tidak ada perubahan, terlihat indikasi 0. Sistem ini bekerja berdasarkan kemampuan penerima dalam membedakan antara informasi yang dimodifikasi dan yang belum.
6. Teknik Distorsi (*Distortion Techniques*), informasi yang hendak disembunyikan disimpan berdasarkan distorsi sinyal. Teknik ini menciptakan perubahan atas benda yang ditumpangi oleh data rahasia.

7. Teknik Pembangkitan Wadah (*Cover Generation Techniques*), Teknik ini menyembunyikan informasi rahasia sejalan dengan pembangkitan *cover*.

2.6 BPCS Steganografi

Bit-plane complexity segmentation (BPCS) adalah salah satu teknik steganografi yang diperkenalkan oleh Eiji Kawaguchi dan R. O. Eason pada tahun 1997. Pada BPCS dokumen citra pada setiap segmen dibagi menjadi segmen-segmen dengan ukuran 8x8 piksel. Pada dokumen citra 8-*bit*, setiap satu segmen akan memiliki 8 buah *bit-plane* yang merepresentasikan piksel-piksel dari setiap *bit* tersebut. Proses penyisipan data dilakukan pada segmen yang memiliki kompleksitas yang tinggi atau disebut juga sebagai *noise-like region*.

Teknik BPCS merupakan teknik steganografi yang memiliki kapasitas besar, karena dapat menampung data rahasia dengan kapasitas yang relatif besar jika dibandingkan dengan metode steganografi lain seperti LSB (*Least Significant Bit*), karena pada teknik BPCS penyisipan dilakukan tidak hanya pada *least significant bit*, tapi pada seluruh *bit-plane*. Teknik BPCS ini adalah teknik steganografi yang tidak berdasarkan teknik pemrograman, tetapi teknik yang menggunakan sifat penglihatan manusia. Sifat penglihatan manusia yang dimanfaatkan yaitu ketidakmampuan manusia menginterpretasi pola biner yang sangat rumit [2].

Eiji Kawaguchi dan R. O. Eason memperkenalkan teknik BPCS ini pada dokumen citra berwarna yang tidak terkompresi dengan format BMP. Dokumen citra tersebut dibagi menjadi beberapa segmen, setiap segmen dibagi dengan ukuran 8x8 piksel. Pada dokumen citra 8-*bit*, setiap satu segmen memiliki 8 buah *bit plane* yang merepresentasikan piksel-piksel dari setiap *bit* tersebut. Proses pembagian segmen 8x8 piksel menjadi 8 buah *bit plane* disebut proses *bit slicing*. Pada BPCS, proses penyisipan dilakukan pada *bit plane* dengan sistem CGC (*Canonical Gray Code*) karena proses *bit slicing* pada CGC cenderung lebih baik dibandingkan pada PBC (*Pure*

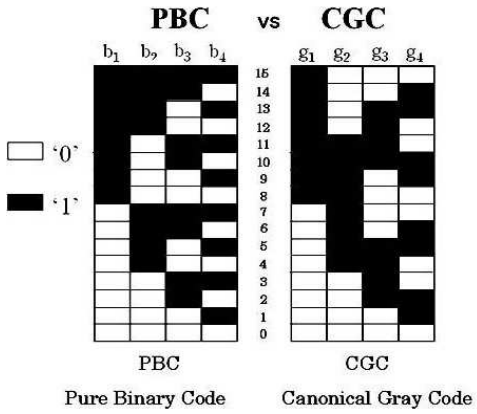
Binary Code). Sehingga pada proses penyisipan, *bit plane* dengan sistem PBC diubah menjadi *bit plane* dengan sistem CGC.

2.6.1 CGC Pada BPCS

Pure Binary Code (PBC) pada *bit-plane* menyediakan daerah yang lebih besar untuk penyisipan. Tapi PBC mengalami masalah "*Hamming Cliff*", dimana perubahan kecil dalam warna mempengaruhi banyak *bit* dari nilai warna.

Pertimbangkan dalam gambar 8-*bit*, ada dua piksel berturut-turut memiliki nilai intensitas 127 dan 128 masing-masing. Pada PBC, 127 direpresentasikan sebagai 01111111 dan 128 direpresentasikan sebagai 10000000. Kedua piksel tampaknya identik dengan mata manusia tetapi sangatlah berbeda dalam representasi *bit*. Ini disebut konsep "*Hamming Cliff*". Jika data rahasia tertanam, maka ada kemungkinan bahwa bisa menjadi 01111111 bisa menjadi 11111111 dan 10000000 bisa menjadi 00000000.

Ada perbedaan intensitas satu tingkat keabuan yang diabaikan oleh mata manusia. Sekarang, setelah *embedding*, perbedaan tingkat abu-abu adalah bahwa dari 255 yaitu satu piksel muncul hitam gelap sementara piksel lainnya tampak putih murni. Perubahan ini mudah terlihat oleh mata manusia. Kelemahan ini dihindari oleh *Cannonical Gray Coding* (CGC), pada CGC teknik pengkodean abu-abu digunakan. Dengan demikian, 127 yang direpresentasikan dalam bentuk biner sebagai 01111111 sekarang direpresentasikan sebagai 01000000 di CGC. Demikian pula, 128 direpresentasikan dalam CGC sebagai 11000000. Sekarang, dua piksel terlihat sama, tetapi berbeda hanya dengan satu *bit*. Hal ini persis berlawanan dengan PBC. Jadi, CGC tidak mengalami masalah dari "*Hamming Cliff*". Setelah *embedding*, 01000000 bisa menjadi 11000000 dan 110000000 bisa menjadi 01000000 [16]. Jadi, perubahan piksel terjadi dalam tingkat intensitas yang tidak terlalu mencolok. Oleh karena itu, sistem CGC lebih dipakai daripada sistem PBC di BPCS.



Gambar 2.6 Sistem PBC dan CGC

Berikut ini adalah rumus persamaaan antara gambar biner PBC dan CGC (dengan ⊗ adalah *Ekclusive* OR)

$g_1 = b_1$

$g_i = b_{i-1} \otimes b_i$, $i>1$

$b_1 = g_1$

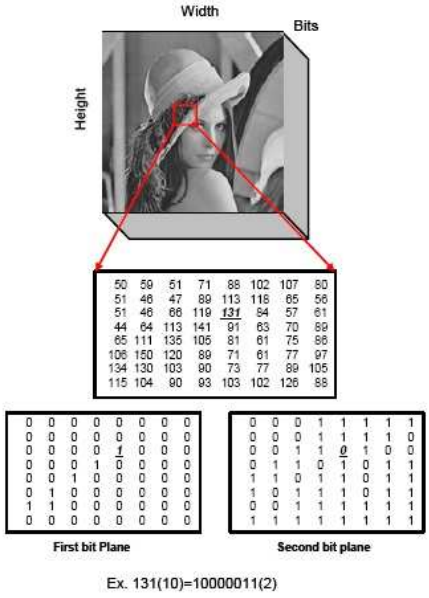
$b_i = g_i \otimes b_{i-1}$, $i>1$

dengan g_i : nilai *bit* ke-i pada sistem CGC

b_i : nilai *bit* ke-i pada sistem PBC

2.6.2 *Bit-plane Slicing* Pada BPCS

Sebuah citra *multi-valued* dengan kedalaman n *bit* dapat diuraikan menjadi bentuk n-gambar biner (*bit plane*) dengan operasi *bit slicing* [2]. *Bit-pale slicing* dapat dipahami pada gambar 2.7



Gambar 2.7 *Bit* Plane Slicing

Piksel adalah nomor digit yang terdiri dari *bit*. Dalam satu piksel terdiri dari 8-*bit*. Satu *bit* terdiri dari 8 *bit plane*. *Plane* ‘0’ berisi *bit* urutan terendah (LSB) dari semua piksel dalam gambar, sementara *plane* ‘7’ berisis *bit* orde tinggi (MSB). Sebagai contoh, misalkan ada citra P dengan kedalaman n-*bit*, dapat ditunjukkan

$$P = (P_1, P_2, \ldots, P_n)$$

Pi merupakan *bit* plane ke-i, dengan i = 1, 2, …, n.

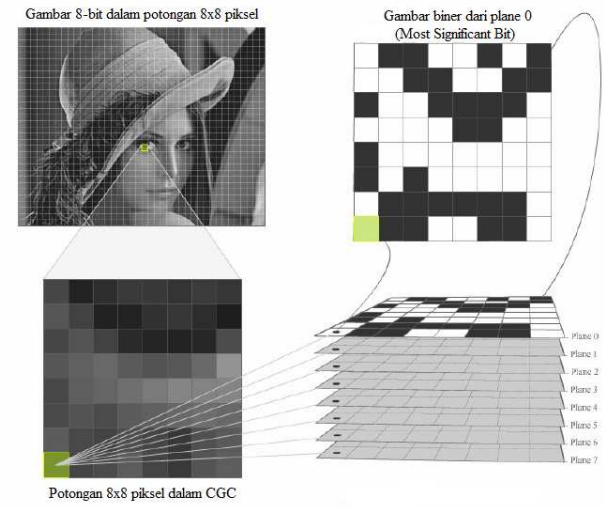
Jika citra P terdiri dari 3 warna, *red*, *green*, *blue*, maka dapat ditunjukkan

$P = (P_{R1}, P_{R2}, \ldots, P_{Rn}, P_{G1}, P_{G2}, \ldots, P_{Gn}, P_{B1}, P_{B2}, \ldots, P_{Bn})$

P_{Ri}: *bit-plane* ke-i untuk *red*

P_{Gi}: *bit-plane* ke-i untuk *green*

P_{Bi}: *bit-plane* ke-i untuk *blue*



Gambar 2.8 Citra dan Potongan 8x8 piksel

2.6.3 Kompleksitas Pada *Bit-Plane*

Ukuran kompleksitas berdasarkan panjang perbatasan hitam dan putih dalam gambar biner (*black-white border image complexity*) [2]. Total panjang perbatasan hitam dan putih adalah sama dengan penjumlahan dari jumlah perubahan warna di sepanjang baris dan kolom dalam gambar.

	B	
B	W	B
	B	

Gambar 2.9 *Pixel* Putih yang Dikelilingi 4 *Pixel* Hitam

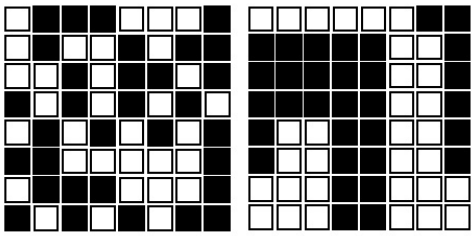
Pada gambar 2.9, sebuah piksel putih dikelilingi oleh 4 piksel hitam. Jadi ada total 4 perubahan warna dengan panjang perbatasan adalah 4. Perubahan warna yang dihitung adalah berdasarkan posisi atas, kanan, bawah, dan kiri. Secara horizontal dan vertikal, tidak secara diagonal.

Kompleksitas gambar dilambangkan dengan 'α' dan diberikan oleh persamaan

$$\alpha = \frac{k}{2\ x\ 2^m x\ (2^m - 1)}$$

Dimana ‘k’ adalah perubahan warna hitam-putih dan α adalah nilai kompleksitas . Untuk sebuah citra biner persegi dengan ukuran 2m x 2m

,kemungkinan maksimal perubahan warna adalah $2*2^m*(2^m-1)$ dan kemungkinan minimum perubahan warnanya adalah 0, diperoleh untuk gambar semua hitam atau semua putih [16].



Gambar 2.10 Contoh dengan Nilai Kompleksitas Antara 0 dan 1

Karena kedua gambar 2.10 memiliki ukuran 8x8, nilai m =3 (2^3 x 2^3), maka nilai total panjang (k) adalah 2 x 8 x 7 = 112. Untuk kedua kondisi di atas diperoleh hasil gambar kiri memiliki α = 69/112 = 0,62. Sedangkan gambar kanan adalah α = 29/112 = 0,23.

2.6.4 *Informative* dan *Noise-like Region*

Informative image berarti gambar yang simpel, sedangkan *noise-like region* berarti gambar yang kompleks. Hal ini hanya berlaku pada kasus dimana sebuah gambar biner merupakan bagian dari sebuah gambar yang natural [2]. Kompleksitas sebuah area *bit-plane* adalah parameter yang digunakan dalam menentukan sebuah *bit-plane* merupakan *informative* atau *noise-like region*.

Setelah menghitung kompleksitas masing-masing *plane*, kita dapat menentukan plane mana saja yang *"informative"* dan *"noise-like"*. Kita dapat menentukan nilai *threshold* (α_0) sebagai batas penentuan nilai kompleksitas. Umumnya nilai *threshold* yang digunakan adalah α_0 = 0,3. Sehingga dapat disimpulkan, semua *plane* yang memiliki nilai kompleksitas di bawah 0,3 adalah *'informative region"* dan yang memiliki kompleksitas di atas 0,3 adalah *"noise-like region"* yang dapat digunakan untuk penyembunyian data.

Untuk *threshold*, semakin memiliki nilai maksimum 1 (kompleksitas maksimum) dan nilai minimum 0 (tidak ada kompleksitas).

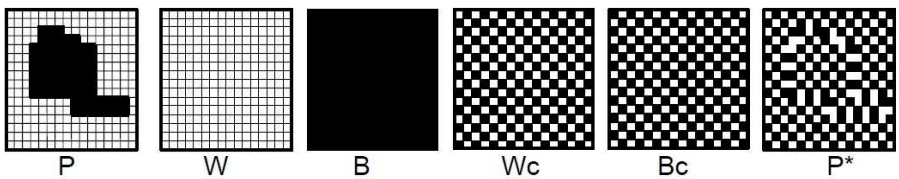
Semakin besar nilai *threshold* yang digunakan, maka semakin tinggi batasan noise yang diambil, sehingga jumlah *noise* yang bisa dimanfaatkan juga semakin sedikit. Namun pengaruhnya terhadap kualitas gambar stego adalah, gambar stego yang dihasilkan akan semakin baik kualitasnya, karena semakin sedikit *noise* yang diganti dengan pesan.

Pada gambar 2.10, diperoleh bahwa gambar kiri adalah *"noise-like region"* sedangkan yang kanan adalah "*informative region"*.

2.6.5 Operasi Konjugasi Pada Gambar Biner

Citra biner terdiri dari wilayah informatif (*informative region*) dan wilayah *noise* (*noise-like region*). Pola informatif sederhana sementara daerah *noise* kompleks. Jika data rahasia adalah sebuah *noise* maka langsung tertanam dalam daerah *noise* dari gambar pembawa pesan. Jika data rahasia informatif maka harus menjalani operasi konjugasi untuk mengubahnya dengan pola yang kompleks. Baru kemudian ditanamkan pada daerah *noise* dari gambar pembawa.

Konjugasi dari suatu gambar biner P adalah sebuah gambar biner lainnya yang memiliki nilai kompleksitas sebesar satu dikurangi nilai kompleksitas P. Misalkan sebuah gambar hitam-putih P berukuran 8x8 piksel memiliki warna *background* putih dan warna *foreground* hitam. W adalah pola dengan semua piksel berwarna putih dan B adalah pola dengan semua piksel berwarna hitam. Wc dan Bc adalah pola papan catur, dengan piksel pada bagian kiri atas berwarna putih pada Wc dan hitam pada Bc. P* adalah konjugasi dari gambar P.



Gambar 2.11 Ilustrasi Proses Konjugasi

Dari Gambar 2.11 dapat dilihat bahwa P adalah gambar yang memiliki piksel *background* dengan pola W dan piksel *foreground* dengan

pola B. P* yang merupakan konjugasi dari P memiliki spesifikasi sebagai berikut [2]:

1. Memiliki bentuk area *foreground* sama dengan P.
2. Memiliki pola area *foreground* sama dengan pola Bc
3. Memiliki pola area *background* sama dengan pola Wc.

Untuk membangun sebuah konjugasi P* dari sebuah gambar P, dapat dilakukan dengan rumus berikut, dimana "⊗" menandakan operasi *exclusive* OR (XOR).

$P^* = P \otimes Wc$

$(P^*)^* = P$

$P^* \neq P$

Jika (P) adalah kompleksitas dari P, maka :

$\alpha(P^*) = 1 - \alpha(P)$

2.6.6 Peta Konjugasi

Peta konjugasi yang dibuat bertujuan untuk menjadi penanda dan petunjuk bagi program dalam proses deskripsi. Peta konjugasi akan ditempatkan secara permanen pada bagian noise pertama dari gambar penampung.

Peta konjugasi juga dibuat dalam wujud *bit-plane*, dan akan dikonjugasi untuk menambah nilai kompleksitasnya. Baris pertama dari peta konjugasi akan diisi dengan nilai *bit* 1, yaitu 00000001, sebagai penanda bahwa gambar penampung yang bersangkutan sudah berisi pesan rahasia.

Baris kedua berisi jumlah noise yang harus diterjemahkan kembali menjadi string pesan rahasia. Jumlah ini juga dibuat dalam bentuk *bit*. Tambahkan penanda untuk menkonjugasikan *plane noise* terakhir jika dibutuhkan pada baris ke-3 pada peta. Sisa peta konjugasi akan diisi dengan nilai 00000000 sampai menjadi *bit-plane*, baru kemudian dikonjugasikan untuk menambah *noise* dan disisipkan menggantikan *noise* pertama dari

gambar penampung.

2.6.7 Algoritma BPCS

Algoritma penyisipan pada BPCS :

1. Ubah gambar penampung dari sistem PBC menjadi sistem CGC.
2. Sebelumnya, gambar tersebut dipotong terlebih dahulu menjadi *bit-plane*. Setiap *bit-plane* mewakili *bit* dari setiap piksel.
3. Segmentasi setiap *bit-plane* pada gambar penampung menjadi *informative* dan *noise like region* dengan menggunakan nilai batas/*threshol*d (α0).
4. Bagi setiap *byte* pada data rahasia menjadi blok-blok(S).
5. Jika blok(S) tidak lebih kompleks dibandingkan dengan nilai batas, maka lakukan konjugasi terhadap S untuk mendapatkan S* yang lebih kompleks.
6. Sisipkan setiap blok data rahasia ke *bit-plane* yang merupakan *noise-like region*. Kemudian simpan data konjugasi pada “conjugation map”.
7. Sisipkan juga pemetaan konjugasi yang telah dibuat.
8. Ubah gambar stego dari sistem CGC menjadi sistem PBC.

Proses ekstraksi data rahasia dapat dilakukan dengan menerapkan langkah-langkah penyisipan secara terbalik. Berikut algoritmanya :

1. Ubah gambar stego dari sistem PBC menjadi sistem CGC. Sebelumnya, gambar tersebut dipotong terlebih dahulu menjadi *bit-plane*. Setiap *bit-plane* mewakili *bit* dari setiap piksel.
2. Segmentasi setiap *bit-plane* pada gambar penampung menjadi *informative* dan *noise like region* dengan menggunakan nilai batas/*threshold* (α0). Dimana *bit-plane* berisi pesan adalah *noise region* dari gambar stego.
3. Terjemahkan peta konjugasi pada noise pertama. Hitung banyak noise yang harus diterjemahkan kembali menjadi pesan.
4. Ubah *bit-plane* menjadi *bit-bit* per barisnya, kemudian terjemahkan menjadi *string* huruf pesan rahasia. Untuk *noise* terakhir berisi pesan,

jika perlu dikonjugasi, maka lakukan konjugasi terlebih dahulu sebelum diterjemahkan menjadi pesan.

5. Kumpulkan string-string pesan secara keseluruhan, kemudian susun kembali menjadi pesan rahasia.

2.7 Kriptografi

Kriptografi berasal dari bahasa yunani, menurut bahasa kriptografi dibagi menjadi dua *kripto* dan *graphia*, *kripto* berarti *secret* (rahasia) dan *graphia* berarti *writing* (tulisan). Menurut teminologinya kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan ketika pesan di kirim dari suatu tempat ketempat yang lain. Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan data, keabsahan data, integritas data, serta autentikasi data. Ketika suatu pesan dikirim dari suatu tempat ke tempat lain, isi pesan tersebut mungkin dapat disadap oleh pihak lain yang tidak berhak. Agar pesan tidak terbaca, maka pesan tersebut dapat diubah menjadi suatu kode yang tidak dapat dimengerti oleh pihak lain.

Suatu sistem kriptografi terdiri dari sebuah algoritma, seluruh kemungkinan *plainteks*, *cipherteks* dan kunci-kuncinya. Sistem kriptografi merupakan suatu fasilitas untuk mengkonversikan plainteks menjadi *cipherteks*, dan sebaliknya. Beberapa istilah yang digunakan pada kriptografi adalah :

1. *Plaintext* adalah pesan yang hendak dikirimkan (berisi data asli).
2. *Ciphertext* adalah pesan ter-enkrip (tersandi) yang merupakan hasil enkripsi.
3. Kunci adalah suatu bilangan yang dirahasiakan yang digunakan dalam proses enkripsi dan dekripsi.
4. Enkripsi adalah sebuah proses penyandian yang melakukan perubahan sebuahkode (pesan) dari yang bisa dimengerti (*plainteks*) menjadi sebuah kode yang tidak bisa dimengerti (*cipherteks*).

5. Dekripsi adalah proses merubah cipherteks menjadi plainteks disebut. Proses enkripsi dan dekripsi memerlukan suatu mekanisme dan kunci tertentu.
6. *Cipher* adalah suatu fungsi matematis yang digunakan untuk melakukan enkripsi dan dekripsi.
7. Kriptoanalisis (*cryptanalysis*) adalah kebalikan dari kriptografi, yaitu suatu ilmu untuk memecahkan mekanisme kriptografi dengan cara mendapatkan kunci dari cipherteks yang digunakan untuk mendapatkan plainteks.

Kriptografi terdiri dari 2 bagian yaitu bagian enkripsi dan bagian dekripsi. Proses menyandikan *plainteks* menjadi *cipherteks* disebut enkripsi (*encryption*). Sedangkan proses mengembalikan *cipherteks* menjadi *plainteks* disebut dekripsi (*decryption*).

Algoritma kriptogarfi disebut juga *cipher*, yaitu aturan untuk enkripsi dan dekripsi, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Konsep matematis yang mendasari algoritma kriptografi adalah relasi antara dua buah himpunan yang berisi elemen – elemen *plainteks* dan himpunan yang berisi *cipherteks*. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen - elemen antara dua himpunan tersebut. Misalkan *P* menyatakan *plainteks* dan *C* menyatakan *cipherteks*, maka fungsi enkripsi *E* memetakan *P* ke *C*.

$E(P) = C$

Dan fungsi dekripsi *D* memetakan *C* ke *P*

$D(C) = P$

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesan semula, maka kesamaan berikut harus benar,

$D(E(P)) = P$

Kriptografi mengatasi masalah keamanan data dengan menggunakan kunci, yang dalam hal ini algoritma tidak dirahasiakan lagi, tetapi kunci harus tetap dijaga kerahasiaannya. Kunci (*key*) adalah parameter yang digunakan untuk transformasi enkripsi dan dekripsi. Kunci biasanya

berupa string atau deretan bilangan. Dengan menggunakan K (kunci) , maka fungsi enkripsi dan dekripsi dapat ditulis sebagai :

EK (P) = C dan *DK (C) = P*

Dan kedua fungsi ini memenuhi

DK (EK(P)) = P

Keterangan :

P = plainteks

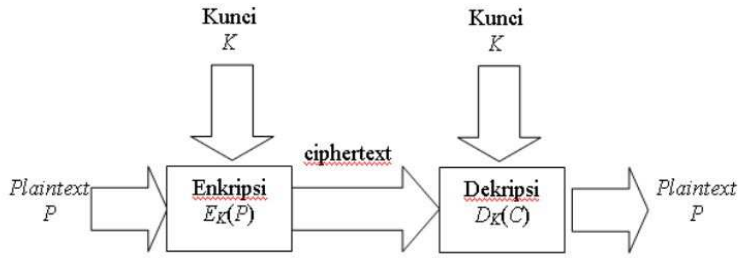
C = cipherteks

K = kunci

EK = proses enkripsi menggunakan kunci *K*

DK = proses dekripsi menggunakan kunci *K*

Skema enkripsi dan dekripsi dengan menggunakan kunci diperlihatkan pada gambar 2.12



Gambar 2.12 Skema enkripsi dan dekripsi dengan menggunakan kunci

Ada empat tujuan mendasar dari ilmu kriptografi ini yang juga merupakan aspek keamanan informasi yaitu :

1. Kerahasiaan, adalah aspek yang berhubungan dengan penjagaan isi informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka informasi yang telah dienkripsi.
2. Integritas data, adalah aspek yang berhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.

3. Autentikasi, adalah aspek yang berhubungan dengan identifikasi atau pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain.
4. *Non-repudiation* (menolak penyangkalan), adalah usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman suatu informasi oleh yang mengirimkan, atau harus dapat membuktikan bahwa suatu pesan berasal dari seseorang, apabila ia menyangkal mengirim informasi tersebut.

2.7.1 Algoritma Simetris

Algoritma simetris adalah algoritma kriptografi yang menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Algoritma ini mengharuskan pengirim dan penerima menyetujui suatu kunci tertentu sebelum mereka saling berkomunikasi. Kelebihan dan kekurangan algoritma simetris adalah :

1. Kelebihan Algoritma Simetris
   a. Kecepatan operasi lebih tinggi bila dibandingkan dengan algoritma asimetrik.
   b. Karena kecepatannya yang cukup tinggi, maka dapat digunakan pada sistem *real-time*
2. Kelemahan Algortima Simetri
   a. Untuk tiap pengiriman pesan dengan pengguna yang berbeda dibutuhkan kunci yang berbeda juga, sehingga akan terjadi kesulitan dalam manajemen kunci tersebut.
   b. Perlu adanya kesepakatan untuk jalur yang khusus untuk kunci, hal ini akan menimbulkan masalah yang baru karena tidak mudah untuk menentukan jalur yang aman untuk kunci, masalah ini sering disebut dengan "*Key Distribution Problem*".

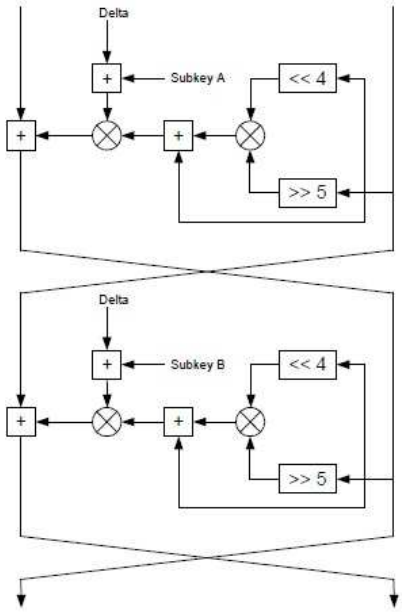
2.7.2 Algoritma Asimetris

Algoritma asimetris (*asymmetric algorithm*) adalah suatu algoritma dimana kunci enkripsi yang digunakan tidak sama dengan kunci dekripsi. Pada algoritma ini menggunakan dua kunci yakni kunci publik (*public key*) dan kunci privat (*private key*). Kunci publik disebarkan secara umum sedangkan kunci privat disimpan secara rahasia oleh si pengguna. Sifat kunci yang seperti ini membuat pengirim harus selalu memastikan bahwa jalur yang digunakan dalam pendistribusian kunci adalah jalur yang aman atau memastikan bahwa seseorang yang ditunjuk membawa kunci untuk dipertukarkan adalah orang yang dapat dipercaya. Masalahnya akan menjadi rumit apabila komunikasi dilakukan secara bersama-sama oleh sebanyak n pengguna dan setiap dua pihak yang melakukan pertukaran kunci, maka akan terdapat sebanyak (n-1)/2 kunci rahasia yang harus dipertukarkan secara aman. Kelebihan dan kekurangan algoritma asimetris adalah :

1. Kelebihan :
   a. Masalah keamanan pada distribusi kunci dapat lebih baik
   b. Masalah manajemen kunci yang lebih baik karena jumlah kunci yang lebih sedikit
2. Kelemahan :
   a. Kecepatan yang lebih rendah bila dibandingkan dengan algoritma simetris
   b. Untuk tingkat keamanan sama, kunci yang digunakan lebih panjang dibandingkan dengan algoritma simetris.

2.7.3 Algoritma XTEA (*eXtended Tiny Encryption Algorithm*)

Wheeler dan Needham menciptakan XTEA pada tahun 1997 untuk menutupi kelemahan pada TEA. Pada XTEA, pada ronde ganjil digunakan *k[sum & 3]*, sedangkan pada ronde genap digunakan *k[sum >> 11 & 3]*. Struktur penyandian XTEA terlihat pada gambar 2.13 sebagai berikut [9]:



Gambar 2.13 Algoritma XTEA

2.8 *Peak Signal to Noise Ration* (PSNR)

Mean Square Error (MSE), yaitu sigma dari jumlah error antara citra yang telah disisipi pesan dan citra asli.

$$MSE = \frac{1}{mn} \sum_{i}^{m} \sum_{j}^{n} ||I(i,j) - K(i,j)||^2$$

Keterangan :

MSE = nilai *Mean Square Error* citra

M = panjang citra (piksel)

N = lebar citra (piksel)

(i,j) = koordinat masing-masing piksel

I = nilai *bit* dari citra pada koordinat (i,j)

K = nilai *bit* dari citra yang sudah disisipi pesan pada koordinat (i,j)

Peak Signal to Noise Ratio (PSNR) merupakan nilai (rasio) yang menunjukan tingkat toleransi noise tertentu terhadap banyaknya noise pada

suatu citra. Semakin tinggi nilai PSNR dari suatu citra, maka semakin kecil tingkat kesalahan yang dimiliki citra tersebut.

Untuk menghitung nilai PSNR digunakan persamaan

$$\begin{aligned} PSNR &= 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \\ &= 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \\ &= 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}(MSE) \end{aligned}$$

Keterangan :

PSNR = Nilai PSNR citra (dalam db)

MAX = Nilai maksimum piksel (nilai maksimum piksel warna adalah 255)

MSE = Nilai MSE

Semakin besar nilai PSNR dari citra, maka citra yang telah dimanipulasi semakin mendekati citra aslinya, dengan kata lain semakin bagus kualitas citra hasil manipulasi tersebut. Sebaliknya, semakin kecil nilai PSNR semakin berkurang kualitas citra hasil manipulasi. Nilai PSNR pada umumnya berada pada rentang 20 – 40 db [12].

2.9 Pemodelan Sistem

Pemodelan (*modeling*) adalah proses merancang piranti lunak sebelum melakukan pengkodean (*coding*). Model piranti lunak dapat dianalogikan seperti pembuatan *blueprint* pada pembangunan gedung. Membuat model dari sebuah sistem yang kompleks sangatlah penting karena kita tidak dapat memahami sistem semacam itu secara menyeluruh. Semakin komplek sebuah sistem, semakin penting pula penggunaan teknik pemodelan yang baik. Dengan menggunakan model, diharapkan pengembangan piranti lunak dapat memenuhi semua kebutuhan pengguna dengan lengkap dan tepat, termasuk faktor-faktor seperti *scalability, robustness, security*, dan sebagainya.

2.9.1 UML (*Unified Modelling Language*)

Unified Modeling Language (UML) adalah sebuah standar bahasa pemodelan yang memungkinkan untuk menspesifikasi, memvisualisasi, membangun, dan mendokumentasikan sebuah sistem perangkat lunak. Tujuan dari pemodelan ini adalah untuk memodelkan sistem perangkat lunak dari segi pembangunan, produksi, kualitas, pengurangan biaya, dan juga waktu. [17]

Diagram UML diklasifikasikan menjadi dua kategori; struktural dan *behavioral*. Diagram struktural memodelkan aspek statis dari sistem dan juga fitur-fitur struktural dari sistem, sedangkan diagram *behavioral* menggambarkan perilaku dinamis sebuah sistem. Diagram struktural yang akan dibahas dalam penulisan ini adalah *Class Diagram,* dan diagram *behavioral* yang akan dibahas adalah *Use Case Diagram* dan *Sequence Diagram*. [17]

UML tepat digunakan untuk memodelkan sistem dari mulai memodelkan informasi sistem untuk perusahaan hingga aplikasi web, bahkan untuk sistem yang rumit sekalipun. UML menggunakan class dan operation dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk perancangan aplikasi prosedural dalam VB atau C. [18]

2.9.1.1 *Use Case diagram*

Use case diagram menggambarkan sebuah skenario yang menampilkan interaksi antara pengguna dan sistem. Seorang pengguna dapat berupa seseorang atau sistem lain. Sebuah *use case* mengacu pada tindakan bahwa sistem dapat dilakukan dengan berinteraksi dengan sebuah aktor. Oleh karena itu, use case diagram menunjukkan serangkaian tindakan serta aktor yang dapat melakukan tindakan tersebut [17].

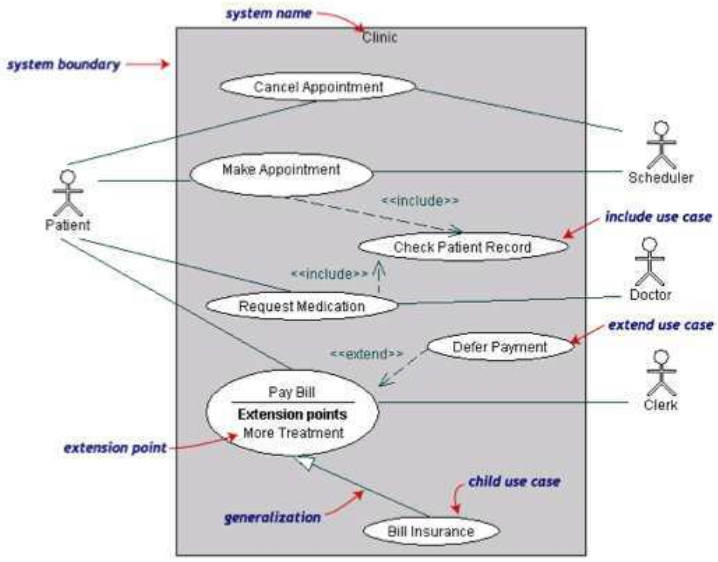
Yang ditekankan dalam *use case diagram* adalah “apa” yang diperbuat sistem, dan bukan “bagaimana” [18].

Secara umum, *use case diagram* terdiri dari tiga bagian di bawah ini:

1. Aktor, adalah sebuah peran dari *user*, *subsystem*, atau piranti yang berinteraksi dengan sistem. Aktor digambarkan dengan figur '*stick*'.
2. *Use case* menyediakan aktor rangkaian aksi yang bisa dilakukan. *Use case* direpresentasikan dengan bentuk oval yang digambarkan secara horizontal.
3. *Association* adalah hubungan dua atau lebih *classifier*, yaitu hubungan koneksi yang terdapat diantara instansi-instansi yang ada dan digambarkan dengan menggunakan garis [17].

Sebuah *use case* dapat meng-*includ*e fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*.

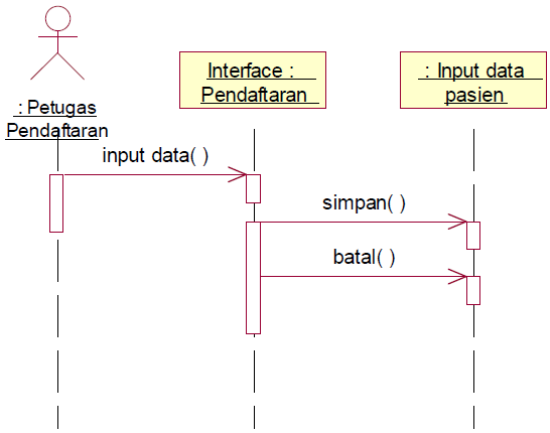
Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain [18].



Gambar 2.14 Contoh *Use Case Diagram*

2.9.1.2 Sequance Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai *respons* dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. Contoh *Sequence diagram* dapat dilihat pada gambar 2.15.



Gambar 2.15 Contoh *Sequence Diagram*

2.9.1.3 *Class Diagram*

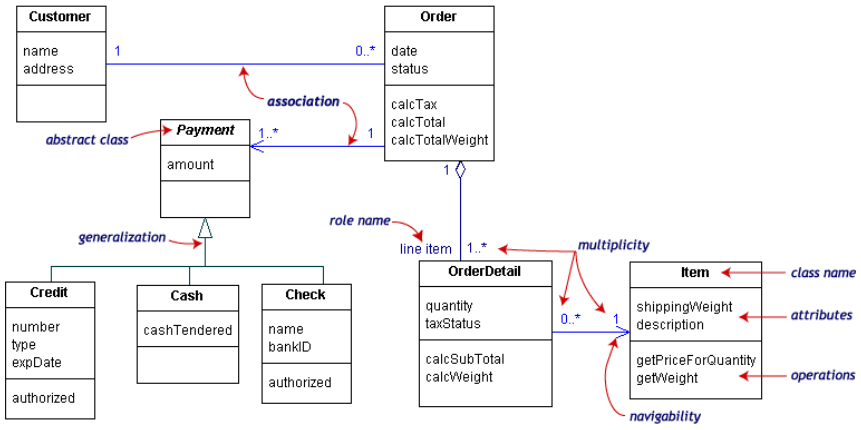
Class diagram merupakan bangunan utama dalam pemodelan berorientasi objek. Diagram ini menggambarkan sebuah pandangan dari satu aspek tertentu dari model atau keseluruhan, menggambarkan struktur elemen beserta hubungan mereka. *Class Diagram* terutama digunakan untuk membangun sebuah arsitektur sistem dengan menangkap dan mendefinisikan *class-class* dan *interface* dan hubungan antara mereka.

Sebuah *class diagram* menggambarkan hubungan antara kelas daripada hubungan antar objek [17].

Hubungan Antar *Class*

1. Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. Panah *navigability* menunjukkan arah *query* antar *class*.
2. Agregasi, yaitu hubungan yang menyatakan bagian ("terdiri atas..").
3. Pewarisan, yaitu hubungan hirarkis antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda *class* asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
4. Hubungan dinamis, yaitu rangkaian pesan (*message*) yang di-*passing* dari satu *class* kepada *class* lain. Hubungan dinamis dapat digambarkan dengan menggunakan *Sequence diagram* yang akan dijelaskan kemudian.

Contoh *class diagram* :



Gambar 2.16 Contoh *Class Diagram*

2.10 Java

Bahasa pemograman Java dikeluarkan pada tahun 1995, kemudian menjadi bahasa pemograman yang banyak digunakan pada World Wide Web. Beberapa tahun berikutnya merupakan bahasa pemograman yang

digunakan untuk bermacam-macam keperluan. Keberhasilan Java disebabkan oleh beberapa faktor, diantaranya penggunaan pada Web dalam jaringan internasional di mana komputer-komputer tidak menggunakan platform yang sama. Dengan demikian Java menjadi bahasa pemograman pilihan untuk program yang perlu dijalankan pada berbagai sistem komputer [19].

Menurut informasi dari *Sun Microsystems,* bahasa pemograman java mempunyai karakteristik sederhana, berorientasi objek, terdistribusi, *interpreter*, kokoh, aman, arsitektur netral, *performance* tinggi, *multithreaded*, dan dinamis.

1. Sederhana

   Pengembangan Java berdaasarkan bahasa pemograman C++, tetapi beberapa fitur dihilangkan. Java mempunyai perbedaan dengan C++ dan C dalam beberapa hal seperti berikut:

   a. Java tidak mendukung tipe data *struct, union* dan *pointer*.
   b. Java tidak mendukung *typedef* atau *#define*
   c. Java berbeda dalam menangani operator tertentu dan tidak mengizinkan operator *overloading.*
   d. Java tidak mendukung *multiple inherince.*
   e. Java mempunyai kelas String yang menjadi bagian dari paket java.lang.
   f. Java menangani argumen *command-line* berbeda.
   g. Java mempunyai sistem otomatis untuk mengalokasikan dan membebaskan memori, sehingga tidak perlu menggunakan fungsi alokasi dan delokasi memori.

2. Berorientasi Objek

   Java mempunyai karakteristik berorientasi objek. Pendekatan berorientasi objek dalam pemograman merupakan fitur bawaan dari program java. Seperti C++, Java dapat mendukung pendekatan berorientasi objek untuk menuliskan program. Dengan demikian, desain

berorientasi objek dapat menggunakan komponen *software* untuk digunakan kembali.

Pemograman berorientasi objek berorientasi objek berdasarkan pemodelan dari dunia nyata yang disebut objek. Objek terdiri dari data dan perilaku yang dikemas dalam objek. Objek dapat mempunyai karakteristik dari objek lain yang menurunkannya, sehingga programmer dapat membuat metode hanya sekali untuk pekerjaan tertentu.

3. Terdistribusi

Tidak seperti bahasa pemograman C++ dan C, Java secara spesifik dirancang untuk pekerjaan dalam lingkungan jaringan. Java mempunyai banyak pustaka kelas untuk komunikasi menggunakan protokol internet TCP/IP, termasuk protokol seperti HTTP dan FTP.

4. *Interpreter*

Kompiler Java menerjemahkan file sumber Java menjadi *byte code* yang dapat dijalankan pada semua komputer yang menjalankan *interpreter* Java atau *browser*. Program Java dapat dibuat oleh programmer pada platform apapun.

5. Kokoh

Perangkat lunak yang kuat tidak mudah menghentikan program karena terdapat *bug* atau kesalahan logika di dalamnya. Pemograman pada umumnya mempunyai keterbatasan termasuk tipe data dan penggunanaan *pointer*. Java tidak mendukung aritmatika *pointer* melainkan array.

6. Aman

Karena Java digunakan dalam berbagai *platform*, maka keamanan sangat diperhatikan. Java membatasi pada *pointer* sehingga *developer* tidak dapat mengakses memori jika tidak diijinkan. Aspek ini membuat Java cukup aman dalam lingkongan *software*.

7. Arsitektur Netral

Kompiler Java membuat *bytecode* yang dikirim ke *browser*dan diterjemahkan oleh *browser* yang mempunyai interpreter Java. Dengan demikian dapat dijalankan pada semua komputer dengan *platform* berbeda.

8. *Performance* Tinggi

   Kompilasi Java termasuk kebebasan menerjemahkan *bytecode* ke kode mesin pada *platform* yang ada. Hal ini memberikan efisiensi yang sama seperti kompilasi dan pemrosesan secara tradisional

9. *Multithreaded*

   Java adalah bahasa pemrograman yang dapat digunakan untuk membuat aplikasi dimana beberapa *event* berlangsung pada saat yang sama.

10. Dinamis

    Tidak seperti kode dalam C++ yang memerlukan kompilasi kembali jika suatu kelas diubah, Java menggunakan metode dari *interface* untuk menghilangkan ketergantungan tersebut. Metode dan variabel dari instance baru dapat digunakan tanpa mempengaruhi objek pada *client* [19].