BAB II
LANDASAN TEORI

Landasan teori menjelaskan dasar-dasar teori yang digunakan dalam penelitian yang dilakukan pada tugas akhir. Teori-teori yang digunakan meliputi keamanan informasi, citra digital, Pengukuran *error* pada citra, steganografi, kriptografi, *Unified Modelling Language* (UML), dan teknologi *java*.

II.1 Keamanan Informasi

Keamanan informasi merupakan salah satu aspek yang penting dalam berkomunikasi, namun keamanan seringkali dilupakan ketika melakukan komunikasi. Informasi yang jatuh ke tangan yang salah dapat menimbulkan masalah yang besar seperti penipuan, pencurian, pemerasan dan masih banyak lagi masalah yang akan ditimbulkan.

Keamanan informasi mempunyai beberapa aspek yang harus dipenuhi, agar informasi dapat terjamin keaslian dan keamanannya. Aspek-aspek umum tersebut meliputi *confidentiality, integrity, authentication, availability* dan *non repudiation*[2].

1. *Confidentiality*

Confidentiality atau kerahasiaan yaitu keamanan informasi harus dapat menjamin kerahasiaan dari informasi. Salah satu cara yang dapat dilakukan yaitu dengan membatasi hanya orang yang mempunyai hak saja yang dapat membaca atau mengubah suatu informasi.

2. *Integrity*

Integrity berhubungan dengan keaslian informasi. *Integrity* yaitu menjamin keutuhan dan keaslian informasi yang dikirimkan, agar informasi tidak dirusak atau diubah oleh orang yang tidak berhak. 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. *Authentication*

Authentication yaitu usaha atau metode untuk mengetahui keaslian dari informasi, dan memastikan bahwa informasi diterima oleh orang yang benar. Untuk menjaga otentikasi terhadap informasi, dapat digunakan *digital signature* untuk memastikan keaslian informasi.

4. *Availability*

Availability atau ketersediaan data yaitu keamanan informasi harus dapat menjamin bahwa data atau informasi harus tersedia ketika akan digunakan. Untuk menjamin ketersediaan data, sistem harus mempunyai cadangan data jika data tersebut hilang.

5. *Non Repudiation*

Non repudiation yaitu menjamin bahwa seorang pengirim informasi tidak dapat menyangkal keaslian dari informasi yang dikirimnya. Sehingga penerima informasi dapat memastikan bahwa informasi yang diterima merupakan informasi yang asli.

II.2 Citra digital

Citra secara harafiah, adalah gambar pada bidang dua dimensi (dwimatra). Ditinjau dari sudut pandang matematis, citra merupakan fungsi menerus (*continue*) dari intensitas cahaya pada bidang dwimatra. Sedangkan citra digital adalah citra yang dapat diolah oleh komputer[4].

Sebuah citra digital dapat diwakili oleh sebuah matriks yang terdiri dari M kolom dan N baris, dimana perpotongan antara kolom dan baris disebut *pixel*, yaitu elemen terkecil dari sebuah citra. Sebuah citra digital dapat ditulis dalam bentuk fungsi II.1 berikut.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & .. & f(0,M-1) \\ f(1,0) & .. & .. & f(1,M-1) \\ .. & .. & .. & .. \\ f(N-1,0) & f(N-1,1) & .. & f(N-1,M-1) \end{bmatrix} \text{.....(II,1)}$$

Terdapat banyak format file dari citra digital, format file citra ini terdiri dari beberapa jenis dan setiap jenisnya memiliki karakteristik masing- masing.

Beberapa contoh format umum, pada citra digital yaitu *Bitmap (BMP)*, *Joint Photographic Group Experts (JPEG)*, *Graphics Interchange Format (GIF)*, dan *Portable Network Graphics (PNG)*.

1. *Bitmap (BMP)*

Bitmap merupakan format baku citra pada sistem operasi *windows* dan IBM OS/2. Citra berformat BMP merupakan citra yang tidak terkompresi, sehingga pada umumnya citra berformat BMP mempunyai ukuran yang relatif lebih besar dibandingkan dengan forman citra lainnya. Intensitas pixel dari citra berformat BMP dipetakan ke sejumlah bit tertentu. Panjang setiap pixel pada *bitmap* yaitu 4 bit, 8 bit, sampai 24 bit yang merepresentasikan nilai intensitas *pixel*. Dengan demikian ada sebanyak $2^8 = 256$ derajat keabuan, mulai dari 0 sampai 255[4].

2. *Joint Photographic Group Experts (JPEG)*

Joint Photographic Group Experts (JPEG) merupakan standar kompresi file yang dikembangkan oleh *Group Joint Photographic Experts* menggunakan kombinasi DCT dan pengkodean *Huffman* untuk mengkompresikan citra. Citra JPEG merupakan citra terkompresi yang bersifat *lossy*, artinya citra tidak bisa dikembalikan ke bentuk aslinya. Citra ini memiliki ukuran yang relatif lebih kecil dibandingkan dengan citra berformat BMP karena telah terkompresi.

3. *Graphics Interchange Format (GIF)*

Graphics Interchange Format (GIF) merupakan format citra terkompresi. Ukuran dari citra berformat GIF merupakan yang terkecil dari semua format citra digital. Kombinasi warna yg tersedia sebanyak 256 warna. Citra jenis ini banyak dugunakan untuk keperluan website, untuk membuat sebuah icon, logo dan keperluan lainnya yang mengharuskan menggunakan citra dengan ukuran yang kecil. Citra jenis ini juga dapat dibuat animasi.

4. *Portable Network Graphics (PNG)*

Portable Network Graphics (PNG) adalah salah satu format penyimpanan citra yang menggunakan metode kompresi yang tidak menghilangkan bagian dari citra tersebut (*lossless compression*). Citra berformat PNG merupakan salah satu format yang baik untuk digunakan pengolahan citra, karena format ini selain tidak menghilangkan bagian dari citra yang sedang diolah.

II.3 Pengukuran *Error* Citra

Pengukuran *error* citra dilakukan untuk mengetahui tingkat kesamaan antara citra asli dengan citra yang telah dimanipulasi. Pengukuran *error* dilakukan dengan menghitung *Mean Square Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR) dari citra.

Mean Square Error (MSE) adalah tingkat kesalahan *pixel – pixel* citra hasil dari pemrosesan terhadap citra aslinya. Rumus untuk menghitung MSE pada citra digital yaitu menggunakan persamaan II.2. Sedangkan *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 II.3.

$$MSE = \frac{1}{\mathrm{MN}} \sum_{y=1}^{m} \sum_{x=1}^{n} [\, I(x,y) - I'(x,y) \,]^2 \quad \text{.....(II.2)}$$

Dimana: I(x,y) adalah nilai *pixel* di citra asli.

I’(x,y) adalah nilai *pixel* pada citra hasil kompresi.

M, N adalah dimensi citra.

$$PSNR = 10\ LOG_{10}\left(\frac{255^2}{MSE}\right) \quad \text{.....(II.3)}$$

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[6].

II.4 Steganografi

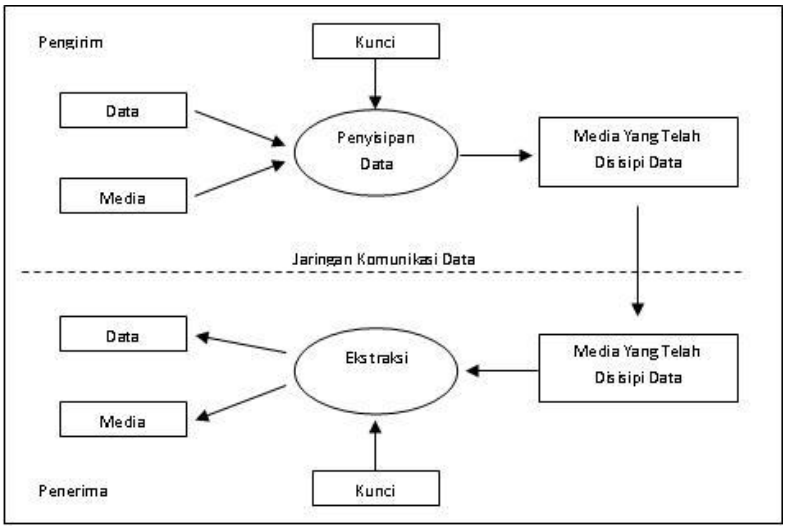
Steganografi merupakan salah satu metode yang dapat digunakan untuk mengamankan informasi. Steganografi berbeda dengan kriptografi atau metode keamanan informasi lainnya, metode ini yaitu menyembunyikan informasi atau pesan kedalam media lain seperti citra digital, teks, suara atau video sehingga tidak menimbulkan kecurigaan orang lain. Steganografi membutuhkan dua

properti, yaitu informasi dan media penampung[4]. Media penampung yang dapan digunakan yaitu citra digital, audio, teks, dan video.

Tujuan dari steganografi adalah menyembunyikan data/pesan pada suatu media. Media penampung data/pesan yang akan disembunyikan dapat berupa gambar digital, suara, video dan media lainnya. Jika pada media yang telah disisipi pesan rahasia tersebut terlihat mencurigakan, maka tujuan dari steganografi tersebut tidak tercapai.

Teknik steganografi sudah dikenal sejak jaman Yunani dan Romawi kuno. Misalnya dengan mencukur kepala budak, lalu pesan rahasia ditulis pada kulit kepalanya. Setelah rambut budak tersebut tumbuh, budak terebut dikirim untuk menyampaikan pesan rahasia tersebut.

Steganografi mempunyai dua proses utama yaitu *embed*/penyisipan dan ekstrak/pengungkapan terlihat pada gambar II.1. Proses penyisipan merupakan proses menyisipkan *hidden object* atau informasi/pesan yang akan disisipkan, ke dalam sebuah *cover object* atau media penampung, sehingga menghasilkan file baru yang telah tersisipi pesan didalamnya yang disebut dengan *stego file*. Sedangkan proses ekstrak merupakan proses pengembalian *hidden object* secara utuh setelah disisipkan ke dalam *cover object*.



Gambar II.1 Proses Steganografi

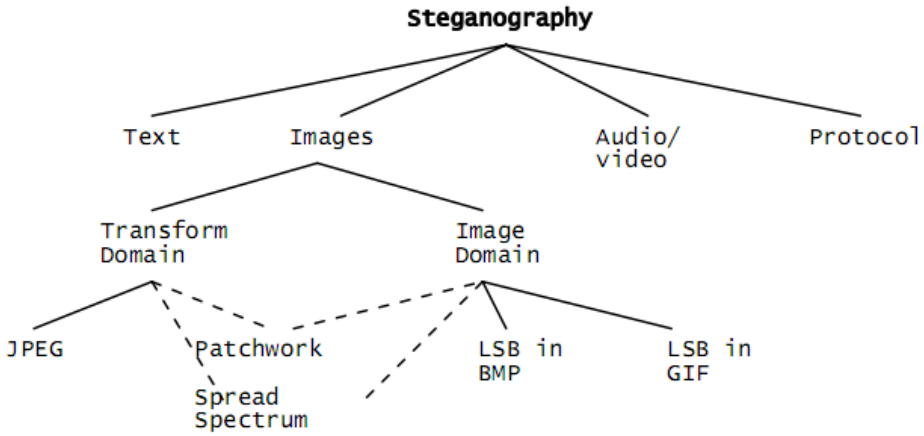
Ada beberapa criteria-kriteria yang harus dipenuhi dalam pembuatan steganografi. Kriteria – kriteria tersebut yaitu[3] :

1. *Impercepbility*, yaitu keberadaan pesan tidak dapat dipersepsi oleh indrawi. Jika pesan disisipkan ke dalam sebuah citra, citra yang telah disisipi pesan harus tidak dapat dibedakan dengan citra asli oleh mata. Begitu pula dengan suara,telinga harus mendapati perbedaan antara suara asli dan suara yang telah disisipi pesan.
2. *Fidelity*, yaitu mutu media penampung tidak berubah banyak akibat penyisipan. Perubahan yang terjadi harus tidak dapat dipersepsi oleh indrawi.
3. *Recovery*, yaitu pesan yang disembunyikan harus dapat diungkap kembali. Tujuan steganografi adalah menyembunyikan informasi, maka sewaktu-waktu informasi yang disembunyikan harus dapat diambil kembali untuk dapat digunakan lebih lanjut sesuai keperluan.

II.5 Steganografi Pada Citra Digital

Citra digital merupakan media penampung yang banyak digunakan dalam steganografi. Penggunaan citra digital sebagai media penampung mempunyai kelebihan karena indera penglihatan manusia memiliki keterbatasan terhadap warna, sehingga dengan keterbatasan tersebut manusia sulit membedakan citra digital yang asli dengan citra digital yang telah disisipi pesan rahasia.

Banyak metode yang dapat digunakan dalam pembuatan steganografi pada citra digital seperti *Least Significant Bits* (LSB), *Bit Plane Complexity Segmentation* (BPCS), *Discrete Cosine Transform* (DCT), *Discrete Wavelet Transform* (DWT), *Spread Spectrum* dan metode-metode lainnya. Dari banyaknya metode tersebut, metode steganografi pada citra digital dapat dibagi menjadi dua bagian yaitu metode pada *transform domain* dan *image domain* seperti pada gambar II.2.



Gambar II.2 Kategori Metode Steganografi Pada Citra Digital[3]

II.5.1 *Transform Domain*

Ranah transform memfokuskan penyisipan pesan ke dalam frekuensi dari *cover-object*. Ranah ini memanfaatkan area *cover-object* yang cenderung tidak akan mengalami pemrosesan digital[3]. Teknik-teknik yang dapat digunakan untuk melakukan seteganografi pada citra digital diranah transform di antaranya adalah sebagai berikut:

1. *Discrete Cosine Transform*
2. *Fourier Transform*
3. *Wavelet Transform*

II.5.2 *Image Domain*

Image Domain merupakan penyembunyian pesan pada citra digital pada ranah spasial, ranah spasial ini juga dikenal sebagai teknik subtitusi. Teknik subtitusi ini dilakukan sedemikian rupa sehingga media penampung yang disisipkan pesan/informasi tidak dapat dipersepsi oleh indrawi perubahannya. Metode steganografi yang beroperasi pada ranah spatial diantaranya yaitu *least significant bit* (LSB), *dynamics cell spreading* (DCS), dan *pixel value differencing* (PVD).

II.6 Pixel Value Differencing (PVD) Steganografi

Pixel Value Differencing merupakan salah satu metode yang dapat digunakan dalam steganografi. Metode ini beroperasi pada ranah spasial dari citra. Berdasarkan analisis terhadap sistem penglihatan manusia yang menyatakan bahwa, mata manusia tidak sensitif terhadap perubahan pada *pixel* yang memiliki kekontrasan tinggi melainkan sensitif terhadap perubahan pada *pixel* yang memiliki kekontrasan rendah. Melalui sifat tersebut maka lebih banyak *bit* data rahasia yang dapat disisipkan pada *pixel* yang memiliki nilai kekontrasan tinggi, dan sedikit *bit* yang dapat disisipkan pada *pixel* dengan kekontrasan rendah. Hal tersebut yang menjadi dasar pemikiran metode *Pixel value differencing* (PVD) pada steganografi[10].

Proses penyisipan pada metode ini dilakukan dengan cara membandingkan dua *pixel* yang bertetangga P_i dan P_{i+1} dengan menggunakan persamaan II.4.

$$d = |P_i - P_{i+1}| \quad \text{(II.4)}$$

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 II.5.

$$t = \lfloor Log_2 w_i \rfloor \quad \text{(II.5)}$$

Dimana : t : Jumlah *bit* yang dapat disisipkan.

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 II.6.

$$d_i' = l_i + b \quad \text{(II.6)}$$

Dimana : b : Nilai decimal dari jumlah *bit* disisipkan.

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

Dimana m didapat dari selisih d'_i dengan d_i menggunakan persamaan II.7.

$$m = |d'_i - d_i| \quad \text{.....(II.7)}$$

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[10].

Berdasarkan informasi tersebut dapat diketahui ukuran data rahasia yang disisipkan pada kedua *pixel* menggunakan persamaan II.5, sehingga pesan rahasia yang telah disisipkan didapatkan kembali. Proses ekstraksi ini dilakukan sampai semua data rahasia yang telah disisipkan didapatkan kembali.

II.7 Kriptografi

Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, otentikasi, dan autentifikasi data[2]. Kriptografi bukan satu-satunya cara untuk menyediakan keamanan informasi, melainkan satu set teknik yang dapat digunakan untuk mengamankan informasi.

Secara umum, kriptografi terdiri dari dua buah bagian utama yaitu bagian enkripsi dan bagian dekripsi. Enkripsi adalah proses transformasi informasi menjadi bentuk lain sehingga isi pesan yang sebenarnya tidak dapat dipahami, hal ini dimaksudkan agar informasi tetap terlindung dari pihak yang tidak berhak menerima. Sedangkan dekripsi adalah proses kebalikan enkripsi, yaitu

transformasi data terenkripsi ke data bentuk semula. Proses transformasi dari *plainteks* menjadi *cipherteks* akan dikontrol oleh kunci. Peran kunci sangatlah penting, kunci bersama-sama dengan algoritma matematisnya akan memproses *plainteks* menjadi *cipherteks* dan sebaliknya.

Kriptografi tidak memenuhi semua aspek dari keamanan informasi. Kriptografi hanya memenuhi empat aspek dalam keamanan informasi yang merupakan tujuan dari kriptografi. Keempat aspek tersebut yaitu kerahasiaan (*confidentiality*), integritas data (*integrity*), otentikasi data (*authentication*), dan *non-repudiation*[5].

1. Kerahasiaan

Kerahasiaan adalah layanan yang digunakan untuk menjaga isi dari informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka informasi yang telah disandi. Kriptografi memenuhi aspek kerahasiaan karena informasi tidak dapat secara langsung diketahui.

2. Integritas data

Integritas data adalah layanan 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. Otentikasi

Otentikasi adalah layanan yang berhubungan dengan identifikasi/ pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan melalui jaringan harus diotentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain. Untuk alasan ini aspek kriptografi biasanya dibagi menjadi dua kelas utama yaitu otentikasi entitas dan otentikasi data asal.

4. *Non-repudiation*

Non-repudiation adalah layanan yang mencegah terjadinya penyangkalan terhadap pengiriman/terciptanya suatu informasi oleh yang mengirimkan/ membuat. Sebagai contoh, satu entitas dapat mengizinkan pembelian properti oleh

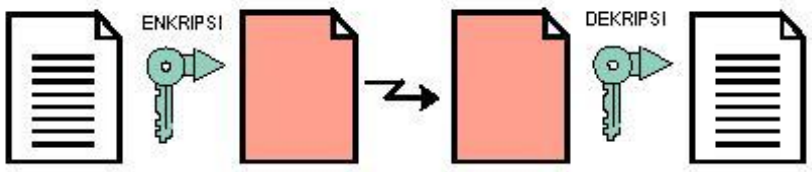
entitas lain dan kemudian berusaha menyangkal otorisasi tersebut diberikan. Sebuah prosedur yang melibatkan pihak ketiga yang terpercaya diperlukan untuk menyelesaikan sengketa tersebut.

II.7.1 Algoritma Kriptografi

Algoritma kriptografi yang handal adalah algoritma kriptografi yang kekuatannya terletak pada kunci, bukan pada kerahasiaan algoritma itu sendiri. Berdasarkan jenis kuncinya, algoritma kriptografi dibagi menjadi dua jenis yaitu algoritma simetris dan algoritma asimetris.

II.7.1.1 Algoritma Simetris

Algoritma simetris (*symmetric algorithm*) adalah suatu algoritma dimana kunci enkripsi yang digunakan sama dengan kunci dekripsi sehingga algoritma ini disebut juga sebagai *single-key algorithm*[5]. Ilustrasi penggunaan algoritma kriptografi dengan kunci simetris dapat terlihat pada gambar II.3 berikut.



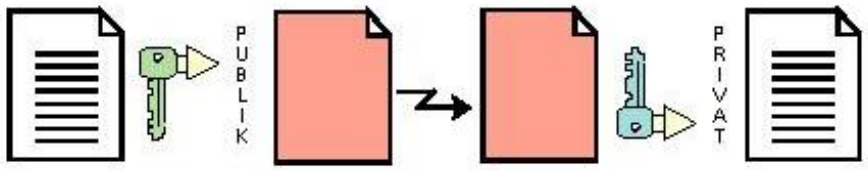
Gambar II.3 Ilustrasi Kriptografi Dengan Kunci Simetris

Algoritma simetris banyak digunakan karena lebih cepat dan lebih simpel, namun penggunaan kunci simetris juga mempunyai kekurangan, karena jika kunci dapat diketahui, maka informasi pun dapat diketahui. Beberapa algoritma kriptografi yang termasuk pada algoritma simetris yaitu DES, AES, *Rijndael*, *Blowfish*, dan IDEA.

II.7.1.2 Algoritma Asimetris

Algoritma asimetris (*asymmetric algorithm*) adalah suatu algoritma dimana kunci enkripsi yang digunakan tidak sama dengan kunci dekripsi[5]. 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 pengguna. Ilustrasi penggunaan algoritma kriptografi dengan kunci asimetris dapat terlihat pada gambar II.4 berikut.



Gambar II.4. Ilustrasi Kriptografi Dengan Kunci Asimetris.

Algoritma asimetris mempunyai keamanan yang lebih baik, karena jika *public key* diketahui, informasi belum tentu dapat diketahui karena *private key* kemungkinan berbeda. Namun akan menjadi sulit dan lama ketika implementasinya. Beberapa algoritma kriptografi yang termasuk pada algoritma asimetris yaitu Diffie – Hellman, RSA, ElGamal, dan DSA.

II.7.2 Algoritma *Rijndael*

Rijndael termasuk dalam jenis algoritma simetris dan *cipher block*. Dengan demikian algoritma ini menggunakan kunci yang sama saat enkripsi dan dekripsi serta masukan dan keluarannya berupa blok dengan jumlah *bit* tertentu.

Rijndael mendukung berbagai variasi ukuran blok dan kunci yang akan digunakan, namun *Rijndael* mempunyai ukuran blok dan kunci yang tetap sebesar 128, 192, 256 *bit*. Pemilihan ukuran blok data dan kunci akan menentukan jumlah proses yang harus dilalui untuk proses enkripsi dan dekripsi. Tabel II.3 adalah perbandingan jumlah proses yang harus dilalui untuk masing-masing masukan.

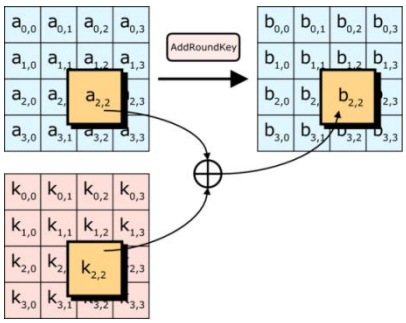
Tabel II.1 Jumlah Proses Berdasarkan Ukuran *Bit* Blok dan Kunci

Ukuran blok dan kunci	Panjang Kunci (Nk) Dalam words	Ukuran Blok Data (Nb) Dalam words	Jumlah Proses (Nr)
128 bit	4	4	10
192 bit	6	4	12
256 bit	8	4	14

Blok-blok data masukan dan kunci dioperasikan dalam bentuk *array*. Setiap anggota *array* sebelum menghasilkan keluaran *cipher text* dinamakan dengan *state*. Setiap *state* akan mengalami proses yang terdiri dari empat tahap yaitu, *Add Round Key*, *Sub Bytes*, *Shift Rows*, dan *Mix Columns*. Kecuali pada tahap *Mix Columns*, ketiga tahap lainnya akan diulang pada setiap proses sedangkan tahap *Mix Columns* tidak akan dilakukan pada tahap terakhir[8].

II.7.2.1 *Add Round Key*

Proses *Add Round Key* yaitu proses menggabungkan *subkey* dengan *state* menggunakan operasi XOR untuk setiap *byte* dari *subkey* dengan *byte* yang dari *state*[8]. Untuk setiap tahap, *subkey* dibangkitkan dari kunci utama dengan menggunakan proses *key schedule*. Proses *Add Round Key* dapat dilihat pada Gambar II.5.



Gambar II.5 Proses *Add Round Key* Algoritma *Rijndael*

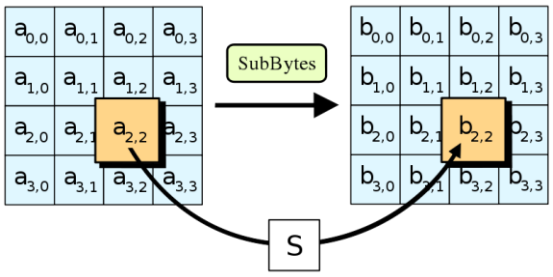
II.7.2.2 *Sub Bytes*

Proses *Sub Bytes* adalah proses substitusi dengan cara mengganti setiap *byte state* dengan *byte* pada sebuah tabel yang dinamakan tabel *S-Box* yang dapat dilihat pada Tabel II.2.

Tabel II.2 *S-Box*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	18	23	C6	E8	87	B8	01	4F	36	A6	D2	F5	79	6F	91	52
1	60	BC	9B	8E	A3	0C	7B	35	1D	E0	D7	C2	2E	4B	FE	57
2	15	77	37	E5	9F	F0	4A	CA	58	C9	29	0A	B1	A0	6B	85
3	BD	5D	10	F4	CB	3E	05	67	E4	27	41	8B	A7	7D	95	C8
4	FB	EE	7C	66	DD	17	47	9E	CA	2D	BF	07	AD	5A	83	33
5	63	02	AA	71	C8	19	49	C9	F2	E3	5B	88	9A	26	32	B0
6	E9	0F	D5	80	BE	CD	34	48	FF	7A	90	5F	20	68	1A	AE
7	B4	54	93	22	64	F1	73	12	40	08	C3	EC	DB	A1	8D	3D
8	97	00	CF	2B	76	82	D6	1B	B5	AF	6A	50	45	F3	30	EF
9	3F	55	A2	EA	65	BA	2F	C0	DE	1C	FD	4D	92	75	06	8A
A	B2	E6	0E	1F	62	D4	A8	96	F9	C5	25	59	84	72	39	4C
B	5E	78	38	8C	C1	A5	E2	61	B3	21	9C	1E	43	C7	FC	04
C	51	99	6D	0D	FA	DF	7E	24	3B	AB	CE	11	8F	4E	B7	EB
D	3C	81	94	F7	B9	13	2C	D3	E7	6E	C4	03	56	44	7F	A9
E	2A	BB	C1	53	DC	0B	9D	6C	31	74	F6	46	AC	89	14	E1
F	16	3A	69	09	70	B6	C0	ED	CC	42	98	A4	28	5C	F8	86

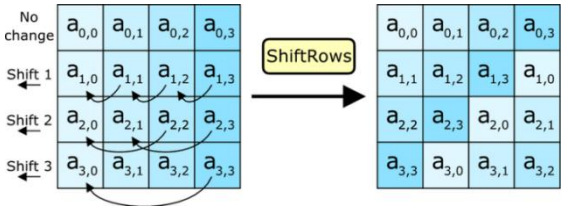
Sebuah tabel *S-Box* terdiri dari 16 baris dan 16 kolom dengan masing-masing berukuran 1 byte. Proses *Sub Bytes* dapat dilihat pada Gambar II.6.



Gambar II.6 Proses *Sub Bytes* Algoritma *Rijndael*

II.7.2.3 *Shift Rows*

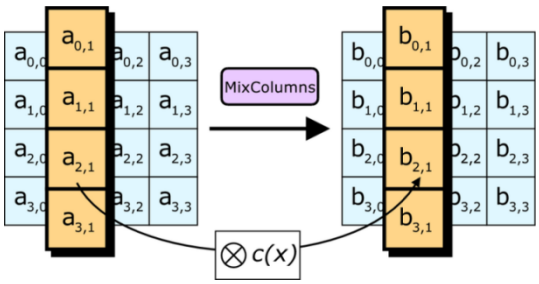
Proses *Shift Rows* adalah proses perputaran pada tiga baris terakhir dari state dengan jumlah putaran yang berbeda-beda. Baris ke-1 akan diputar sebanyak 1 kali, baris ke-2 akan diputar sebanyak 2 kali, dan baris ke-3 akan diputar sebanyak 3 kali, sedangkan baris ke-0 tidak akan diputar[8]. Proses *Shift Rows* diperlihatkan pada Gambar II.7.



Gambar II.7 Proses *Shift Rows* Algoritma *Rijndael*

II.7.2.4 *Mix Columns*

Operasi *Mix Columns* adalah proses menggabungkan 4 *bytes* dari setiap kolom dari tabel *state* dengan menggunakan transformasi linier[8]. Operasi *Mix Columns* memperlakukan setiap kolom sebagai polinomial 4 suku dalam *Galois field* dan kemudian dikalikan dengan c(x) *modulo* (x4+1), dimana c(x)=3x3+x2+x+2. Kebalikkan dari *polinomial* ini adalah c(x)=11x3+13x2+9x+14. Operasi *Mix Columns* juga dapat dipandang sebagai perkalian matrix seperti terlihat pada gambar II.8.



Gambar II.8 Proses *Mix Columns* Algoritma *Rijndael*

II.8 *Unified Modelling Language* (UML)

Unified Modelling Language (UML) adalah sebuah bahasa pemodelan standar yang memiliki sintaks dan semantic. Pemodelan ini sangat cocok digunakan untuk merancang dan memodelkan sistem berorientasi objek[9].

Diagram pada UML dibagi menjadi dua bagian yaitu *structural* diagram dan *behavior* diagram. *Structural* diagram digunakan untuk mendeskripsikan relasi antar kelas. *Tools* yang digunakan pada bagian ini yaitu *class diagram*. Sedangkan *behavior* diagram digunakan untuk mendeskripsikan interaksi antara aktor dan sebuah *use case* (bagaimana seorang aktor menggunakan sistem). *Tools* yang digunakan pada bagian ini yaitu *Use case* diagram, *Sequence* diagram, *Collaboration* diagram, *Statechart* diagram, dan *Activity* diagram.

1. Diagram *Class*

Diagram *class* digunakan untuk menggambarkan keadaan suatu sistem dengan menjelaskan keterhubungan antara suatu *class* dengan *class* yang lain yang terdapat pada sistem[9]. Sebuah *class* terdiri dari nama, atribut dan *method.* Atribut dan *method* dari sebuah kelas mempunyai visibility, Ada tiga jenis visibility yang digunakan yaitu *private*, *public*, dan *protected*. Setiap *class* pada *class* diagram mempunyai hubungan dengan *class* lainnya, ada beberapa jenis hubungan *class*, yaitu *dependency*, asosiasi, agregasi, komposisi dan generalisasi.

a. *Dependency*

Dependency merupakan hubungan terlemah antar *class*. *Dependency* bermakna satu *class* menggunakan atau memiliki pengetahuan terhadap *class* lain, namun hubungannya hanya sementara dan tidak ada batas waktu yang jelas.

b. Asosiasi

Asosiasi mempunyai hubungan yang lebih kuat dari hubungan *dependency*, dimana suatu *class* tetap berhubungan dengan *class* lain seterusnya. Hubungan asosiasi dibagi menjadi dua jenis yaitu *directional* dan *bidirectional*.

c. Agregasi

Agregasi merupakan bentuk hubungan yang mengimplikasikan kepemilikan suatu *class*. Agregasi juga merupakan bentuk yang lebih kuat dari asosiasi. Hubungan agregasi dinyatakan dengan simbol *diamond* pada pemilik kelas dan garis utuh berpanah kekelas yang dimiliki.

d. Komposisi

Komposisi merupakan bentuk hubungan antar *class* yang paling kuat. Komposisi digunakan untuk mengambil seluruh bagian dari *class* yang berhubungan. Aturan dari komposisi yaitu hanya boleh ada satu komposisi dalam satu waktu.

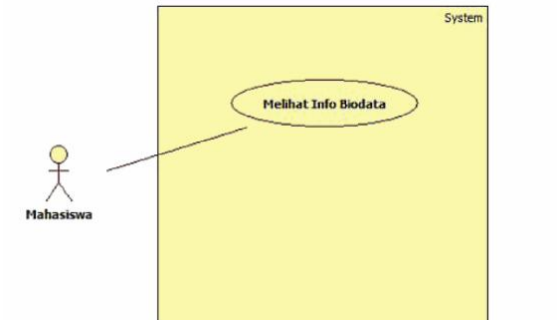
e. Generalisasi

Generalisasi merupakan bentuk hubungan antar *class*, dari *class* yang umum dengan *class* yang lebih khusus. Contoh *class* hewan memiliki hubungan generalisasi dengan *class* kucing, karena kucing merupakan *class* khusus dari *class* hewan yang lebih umum.

2. Diagram *Use Case*

Diagram *use case* digunakan untuk mendeskripsikan kejadian-kejadian apa saja yang dapat dilakukan oleh *user*/aktor dan fungsionalitas-fungsionalitas apa saja yang diharapkan dari sistem yang akan dibangun, tanpa mendeskripsikan bagaimana sistem menyelesaikannya[9]. Sebuah *use case* menggambarkan suatu urutan interaksi antara satu atau lebih aktor dan sistem. Dalam fase requirements, model *use case* mengambarkan sistem sebagai sebuah kotak hitam dan interaksi antara aktor dan sistem dalam suatu bentuk naratif, yang terdiri dari *input user* dan respon-respon sistem.

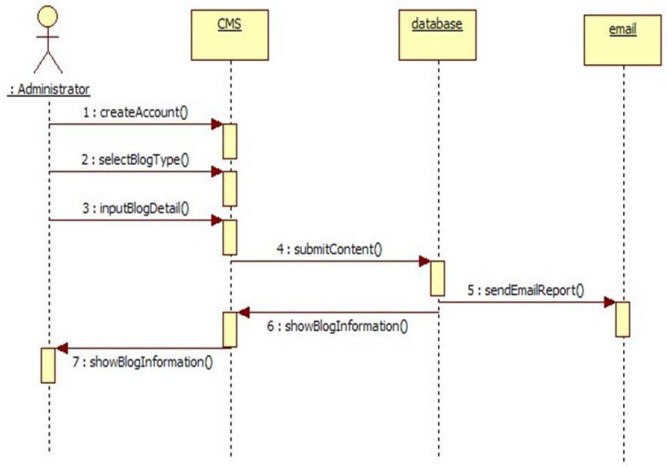
Setiap *use case* menggambarkan perilaku sejumlah aspek sistem, tanpa mengurangi struktur internalnya. Selama pembuatan model *use case* secara pararel juga harus ditetapkan obyek-obyek yang terlibat dalam setiap *use case*. Contoh *use case* diagram diperlihatkan pada gambar II.9.



Gambar II.9 Contoh *Use Case* Diagram

3. Diagram *Sequence*

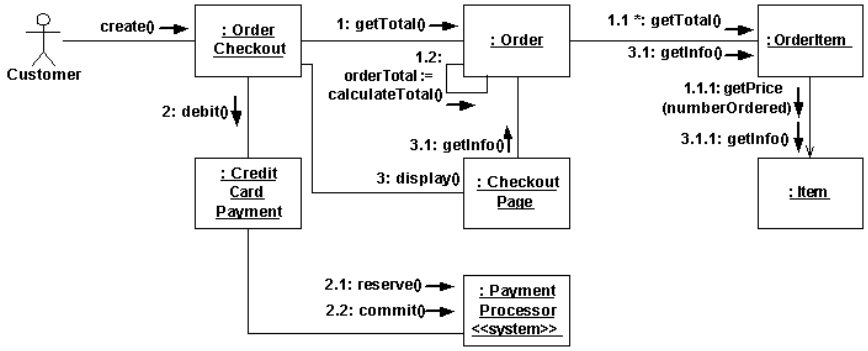
Diagram *sequence* menggambarkan interaksi antar objek di dalam dan di sekitar sistem yang menekankan pada pengiriman pesan dalam suatu waktu tertentu[9]. *Sequence* diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* 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 diperlihatkan pada gambar II.10.



Gambar II.10 Contoh *Sequence* Diagram

4. Diagram *Collaboration*

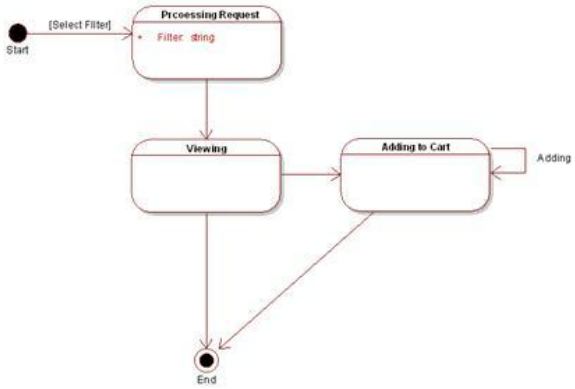
Diagram *collaboration* menunjukan informasi yang sama seperti dalam *sequence* diagram. Perbedaannya adalah dari cara mengelompokan objek *sequence* berdasarkan urutan nomor dari pesan[9]. Contoh *collaboration* diagram diperlihatkan pada gambar II.11.



Gambar II.11 Contoh *Collaboration* Diagram

5. Diagram *Statechart*

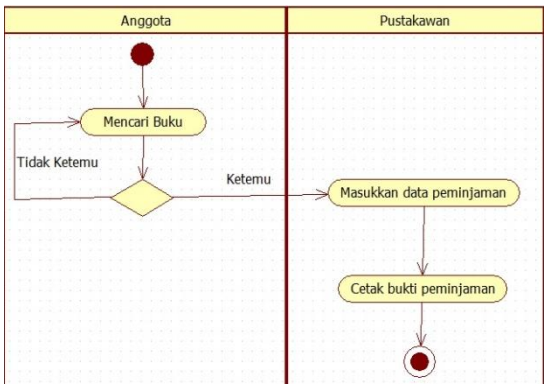
Diagram *statechart* menggambarkan keadaan-keadaan pada sistem, memuat status (*state*), transisi, kejadian serta aktifitas[9]. Contoh *Statechart* diagram diperlihatkan pada gambar II.12.



Gambar II.12 Contoh *Statechart* Diagram

6. Diagram *Activity*

Diagram *activity* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir[9]. *Activity* diagram merupakan *state* diagram khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity* diagram tidak menggambarkan *behaviour* internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Contoh *Activity* diagram diperlihatkan pada gambar II.13.



Gambar II.13 Contoh *Activity* Diagram

II.9 Teknologi *Java*

Java adalah sebuah teknologi yang diperkenalkan oleh *Sun Microsystems* pada pertengahan tahun 1990. Menurut definisi dari *Sun*, *Java* adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *stand alone* ataupun pada lingkungan jaringan. *Java* berdiri di atas sebuah mesin *interpreter* yang diberi nama *Java Virtual Machine* (JVM). JVM inilah yang akan membaca *bytecode* dalam *file .class* dari suatu program sebagai representasi langsung program yang berisi bahasa mesin.

Oleh karena itu, bahasa *Java* disebut sebagai bahasa pemrograman yang *portable* karena dapat dijalankan pada berbagai sistem operasi, asalkan pada sistem operasi tersebut terdapat JVM. *Platform Java* terdiri dari kumpulan *library*, JVM, kelas-kelas *loader* yang dipaket dalam sebuah lingkungan rutin *Java*, dan sebuah *compiler*, *debuger*, dan perangkat lain yang dipaket dalam *Java Development Kit* (JDK)[1].

Java merupakan salah satu bahasa pemrograman yang menggunakan paradigma pemrograman berbasis objek. Paradigma ini yaitu menggunakan objek untuk membungkus atribut dan operasi yang mungkin pada objek tersebut. Java mempunyai kelebihan dan kekurangan yaitu:

A. Kelebihan Java

Beberapa kelebihan dari java yaitu sebagai berikut:

1. Multiplatform

   Kelebihan utama dari *java* ialah dapat dijalankan di beberapa *platform*/sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan dimana saja. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas sistem operasi *linux* tetapi dijalankan dengan baik di atas *Microsoft Windows*.

2. OOP (*Object Oriented Programming*)

   Java merupakan salah satu bahasan pemrograman berbasis objek secara murni. Semua tipe data diturunkan dari kelas dasar yang disebut objek. Hal ini sangat memudahkan pemrogram untuk mendesain, membuat, mengembangkan, dan mengalokasi kesalahan sebuah program dengan basis *java* secara cepat, tepat, mudah dan terorganisir.

3. *Library* yang lengkap

   Java terkenal dengan kelengkapan *library*/perpustakaan (kumpulan program-program yang disertakan dalam pemrogram *java*) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4. Bergaya C++

   Java memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik banyak pemrogram C++ untuk pindah ke *java*. Saat ini pengguna *java* sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke *java*.

5. Pengumpulan sampah otomatis

   Java memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas) [1].

B. Kekurangan Java

Beberapa kekurangan Java antara lain :

1. Mudah didekompilasi

   Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini memungkinkan karena kode jadi *java* merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi seperti nama-nama kelas, metode dan tipe data.

2. Penggunaan memori yang banyak

   Penggunaan memori untuk program berbasis *java* jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan *Pascal* (lebih spesifik lagi *delphi* dan Objek *Pascal*) [1].