BAB II

STUDI PUSTAKA

2.1. Tinjauan Pustaka

Penelitian yang sebelumnya pernah dilakukan yaitu mengenai Implementasi Pengamanan Data dan Informasi dengan Metode Steganografi LSB dan Algoritma Kriptografi AES, memiliki kolerasi yang searah dengan penelitian yang dibahas antara lain:

1. Meneruskan penelitian sebelumnya yang telah dilakukan oleh orang lain. Sehingga dengan adanya studi literatur, penelitian yang akan dilakukan dapat membangun fitur untuk penyandian pesan teks menggunakan algoritma *Advanced Encryption Standard* (AES) dan Metode Steganografi *Least Significant Bit* (LSB) untuk penyisipan pesan kedalam gambar.

1.1.1 Stat Of the Art

Tabel 2. 1 State Of The Art

No	Peneliti	Judul	Masalah	Metode	Data	Hasil
1	Chyquitha Danu putri,dkk	*Data Security Using LSB Steganography and Vigenere Chiper in an Android Evironment*	Keamanandata/informasi dalam pengiriman pesan berbasis android	LSB *Steganography and Vigenere Cipher*	.txt, pdf, .xls. docx .gif, jpeg, .png, bmp	Aplikasi dapat mengirim pesan rahasia melalui semua media *online* dan *Bluetooth*. Perangkat lunak prototipe harus berjalan di android untuk diproses enkripsi, kompresi, dekompresi, *checksum* dan

						steganografi membutuhkan waktu dan memori ekstra.

Tabel 2.1 State Of The Art (Lanjutan)

2	Syamsul Anwar	Implementasi Pengamanan Data dan Informasi dengan Metode Steganografi LSB dan Kriptografi AES	Untuk melihat aspek imperceptibility dan aspek recovery pada metode Modified LSB.	Algoritma AES dan LSB	Text, Bmp	Metode Modified LSB memenuhi aspek imperceptibility, dimana pesan rahasia pada citra digital sulit diprediksi oleh inderawi. Dan berbasis Desktop
3	Gede Wisnu Bhaudhyana, I Made Widiartha	IMPLEMENTASI ALGORITMA KRIPTOGRASFI AES 256 DAN STEGANOGRAFI LSB PADA GAMBAR BITMAP	Pentingnya kemanan terhadapat informasi yang dikirim maupun diterima dengan cara kriptografi dan steganografi.	Kirptografi Algoritma AES 256 dan LSB	Text, Bmp	Algoritma kriptografi AES 256 dan Metode Steganografi LSB baik untuk diimplementasikan dalam mengamankan file gambar yang kerahasiaanya sangat dijaga.
4	Fresly Nandar Pabokory, dkk	Implementasi Kriptografi Pengamanan Data Pada Pesan Teks, Isi File Dokumen, dan File Dokumen Menggunakan Algoritma Advanced Encryption Standard (AES)	Mengimplementasikan kriptografi pada pesan teks, isi file dokumen dan file dokumen dengan melakukan perhitungan algoritma AES.	Algoritma Kriptografi AES	Teks, File dokumen	Pengguna dapat mengenkripsi pesan, file dokumen dan isi file dokumen.
5	Muhamad hasan nurhidayat, Yana aditia gerhana, dkk	Kombinasi Algoritma Kriptografi Vigener Cipher dan Hill Cipher untuk	Untuk mengamankan pesan rahasia, dan mengembangkan penelitian sebelumnya yang hanya	Kriptografi Vigener Cipher, Hill Cipher, Steganografi LSB	Text, Jpg, Png	Aplikasi yang dapat menyembunyikan pesan rahasia kedalam cover image berformat

		Penyediaan Pesan Rahasia pada Metode Steganografi	menggunakan Vigener Cipher dan Steganografi LSB.			jpg dan png. Dan berbasis Mobile

Chyquitha Danuputri, Teddy Mantoro, Mardi Hardjianto dengan judul Data *Security Using Least Significant* Bit *Steganography* and *Vigenere Cipher in an* Android *Environment.* Penelitian ini mengusulkan penggunaan hybrid antara dua pendekatan, yaitu steganografi *Least Significant* Bit dan *Vigenere Cipher*, untuk validasi keamanan data di lingkungan Android. Dengan prototipe ini, sebagai bukti konsep, pesan rahasia dapat dikirim melalui semua media online dan *bluetooth Share* di android, steganografi *Least Significant* Bit dan metode *Vigenere Cipher* di integrasikan dalam digunakan untuk validasi keamanan data. Pendekatan ini menggunakan metode *Arithmetic Coding* untuk data Kompresi dan dekompresi data, untuk menjaga keaslian *file* data, fungsi *hash* (SHA 256)[5]**.**

Pada tahun 2017 Syamsul Anwar, dalam jurnalnya yang berjudul "Implementasi Pengamanan Data dan Informasi dengan Metode Steganografi LSB dan Kriptografi AES" pada penelitian ini dilakukan pengujian aspek *imperceptibility* dan aspek *recovery* pada metode Modified LSB dan mengkombinasikan dengan algoritma AES untuk meningkatkan kemanan data[3].

Pada tahun 2015 Gede Wisnu B, dkk dalam jurnalnya yang berjudul "Implementasi Algoritma Kriptografi AES 256 dan Metode Steganografi LSB pada Gambar Bitmap". Pertukaran informasi yang tak terbatas berdampak terhadap adanya tindakan kejahatan dunia maya berupa penyadapan, pembajakan dan pentingnya sebuah kemanan informasi yang dikirim maupun yang diterima dengan cara kriptografi dan steganografi[6].

Pada tahun 2015 Fresly Nandar Pabokory, dkk dalam jurnalnya yang berjudul "Implementasi Kriptografi Pengamanan Data Pada Pesan Teks, Isi File Dokumen, dan File Dokumen Menggunakan Algoritma Advanced Encryption Standard (AES)". Hasil dari penelitian yaitu pengguna dapat mengenkripsi pesan teks kemudian disimpan menjadi sebuah file dokumen dan isi file dokumen tersebut dienkripsi lagi selanjutnya hasil enkripsi isi file citra (gambar) agar kemanan data informasi tersebut dapat terjaga kemanannya[7].

Pada tahun 2017 Muhamad Hasan N, dkk dalam jurnalnya yang berjudul "Kombinasi Algoritma Kriptografi Vigener Cipher dan Hill Cipher untuk Penyediaan Pesan Rahasia pada Metode Steganografi" dimana pada penelitiannya menggunakan Metode Steganografi LSB dan Kriptografi Vigener Cipher, Hill Cipher. Untuk mengamankan data kedalam gambar berformat jpg dan png.[8]

2.2. Landasan Teori

2.2.1. Multimedia

Multimedia berasal dari dua kata, yaitu multi dan media. *Multi* berarti banyak dan *media* biasa diartikan untuk menyampaikan atau membuat sesuatu, perantaraan, alat pengantar, suatu bentuk komunikasi, seperti surat kabar, majalah, atau televisi. Multimedia ialah penggunaan komputer untuk menyajikan dan menggabungkan teks, suara, gambar, animasi, audio dan video dengan alat bantu (*tool*) dan koneksi (*link*) sehingga pengguna dapat melakukan navigasi, berinteraksi, berkarya dan berkomunikasi[9].

2.2.1.1. Objek Multimedia

- Teks
- Gambar

- Suara
- Animasi
- Video

a. Teks

Teks merupakan bentuk media paling umum digunakan dalam menyajikan informasi, baik yang menggunakan model baris perintah GUI. Teks dapat disajikan dalam berbagai bentuk *font* maupun ukuran[9].

b. Gambar Statis

Banyak format gambar yang telah dimplementasikan dalam sistem komputer. Beberapa format terkenal dapat dilihat pada Tabel 2.2

Tabel 2. 2 Format berbagai gambar pada komputer

Format	**Keterangan**
BMP	Kepanjangannya adalah BitMap Graphics. Format yang biasa digunakan pada DOS dan Windows. Ekstensi : .bmp
CDR	Format gambar yang dihasilkan oleh CorelDraw. Ekstensi : .cdr
DXF	DXF (*Drawing eXchange Format*) adalah format gambar yang dihasilkan oleh program autoCAD. Ekstensi: .dxf
EPS	Kepanjangannya adalah Encapsulated PostScript. Format yang dapat digunakan untuk teks dan gambar. Ekstensi: .eps

Tabel 2.2 Format berbagai gambar pada komputer

GIF	Kepanjangannya adalah *Graphics Interchange Format.* Dikembangkan oleh Compuserve pada tahun 1987, hanya dapat

	menangani 256 warna. Ekstensi: .gif
HPG	Format dari Hewlett Packard (*Hewlett Packard Graphics Language*). Ekstensi: .hpg
JPG	Kepanjangannya adalah Joint Photographic Expert Group. Tingkat kompresinya sangat tinggi. Ekstensi: .jpg, .jpeg, atau .jpe
PCX	Format yang digunakan oleh perangkat lunak Paintbrush. Ekstensi: .pcx
PNG	PNG (*Portable Network Graphics*) dirancang oleh W3C (*World Wide Web Consortium*) untuk menggantikan GIF dan JPEG. Formatnya didesain supaya tidak tergantung pada mesin, sehingga dapat ditangani oleh sembarang jenis komputer dan sistem operasi. Ekstensi: .png
WPG	Format gambar yang dihasilkan oleh DrawPerfect. Ekstensi: .wpg

2.2.2. Kriptografi

Secara etimologi kata kriptografi (*Cryptography*) berasal dari bahasa Yunani, yaitu "*cryptos*" yang artinya yang tersembunyi dan "*graphein"* yang artinya tulisan**.** Awal mula kriptografi dipahami sebagai ilmu tentang menyembunyikan pesan**,** tetapi seiring perkembangan zaman hingga saat ini pengertian kriptografi berkembang menjadi ilmu tentang teknik matematis yang digunakan untuk menyelesaikan persoalan keamanan berupa privasi dan otentikasi[1]**.**

Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya.

Dalam ilmu kriptografi, terdapat dua buah proses yaitu melakukan enkripsi dan dekripsi. Pesan yang akan dienkripsi disebut sebagai plaintext (teks biasa). Disebut demikian karena informasi ini dengan mudah dapat dibaca dan dipahami oleh siapa saja. Algoritma yang dipakai untuk mengenkripsi dan mendekripsi sebuah plaintext melibatkan penggunaan suatu bentuk kunci. Pesan plaintext yang telah dienkripsi (atau dikodekan) dikenal sebagai ciphertext (teks sandi).

Di dalam kriptografi kita akan sering menemukan berbagai istilah atau terminology. Beberapa istilah yang harus diketahui yaitu :

1. Pesan, Plainteks, dan Cipherteks

Pesan (*message*) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah (*plaintext*) atau teks jelas (*cleartext*).

2. Pengirim dan Penerima

Komunikasi data melibatkan pertukaran pesan antara dua entitas. Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (*receiver*) adalah entitas yang menerima pesan.

3. Enkripsi dan dekripsi

Proses menyandikan plainteks menjadi cipherteks disebut enkripsi (*encryption*) atau enciphering (standard nama menurut ISO 7498-2). Sedangkan proses mengembalikan cipherteks menjadi plainteks semula disebut dekripsi (*decryption*) atau deciphering (standard nama menurut ISO 7498-2).

4. Cipher dan kunci

Algoritma kriptogarfi disebut juga cipher, yaitu aturan untuk enkripsi dan dekripsi, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa cipher memerlukan algoritma yang berbeda 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 :

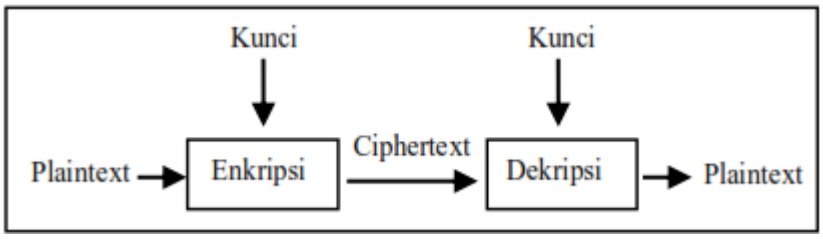
E(P) = C □ fungsi enkripsi E memetakan P ke C

D(C) = P □ fungsi dekripsi D memetakan C ke P

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesan asal, maka persamaan D(E(P)) = P harus benar.

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 kunci K, maka fungsi enkripsi dan dekripsi dapat ditulis sebagai skema diperlihatkan pada Gambar 2.1



2.2.3. Algoritma *Advanced Encryption Standard* (AES)

2.2.3.1 Sejarah *Advanced Encryption Standard* (AES)

AES (Advanced Encryption Standards) dipublikasikan oleh NIST (National Institute Of Standard and Technology) pada tahun 2001 yang digunakan untuk menggantikan algoritma DES yang semakin lama semakin mudah untuk dibobol. AES diperoleh dari hasil kompetisi yang diadakan NIST tahun 1997. Pada tahap pertama 15 peserta dari 21 peserta lolos ke tahap berikutnya berdasarkan penilaian tingkat keamanan, harga, algoritma dan karakteristik implementasi. Sepuluh dari 15 peserta tersebut gugur pada tahap berikutnya karena diangap kurang aman dan efektif. Pada agustus 1999 dipilih lima kandidat dari tahap seleksi akhir, yaitu MARS (IBM, Amerika Serikat), RSA (RSA corp., Amerika Serikat), Rijndael (Belgia), Serpent (Israel, Norwegia, Inggris), dan Twofish (Counterpane., Amerika Serikat). Pada tahap ini NIST memberikan penilaian pada general security, implementasi software dan hardware, ruang lingkup, implementasi atas serangan, enkripsi dan dekripsi, kemampuan kunci, dan kemampuan lain seperti fleksibilitas dan kepotensial

untuk tingkat instruksi pararel. Akhirnya, 2 Oktober 2000 terpilih algoritma Rijndael yang dibuat oleh Dr. Vincent Rijment dan Dr. Joan Daemen sebagai pemenang[1].

Algoritma ini termasuk jenis simetri yang disebut juga sebagai algoritma konvensional, yaitu algoritma yang menggunakan kunci enkripsi dan dekripsi yang sama. AES menggunakan sandi blok kunci simetrik dengan ukuran kunci bervariasi, yaitu 128 bit, 192 bit, dan 256 bit. Pemerintah Amerika Serikat telah mengadopsi AES sebagai standar enkripsi. Standar ini terdiri dari 3 blok chiper, yaitu AES-128, AES-192, AES-256 yang diadopsi dari koleksi yang lebih besar yang awalnya diterbitkan sebagai Rijndael. AES telah dianalisis secara luas dan sekarang digunkan diseluruh dunia, seperti halnya dengan DES (Data Encryption Standard)[10].

2.2.3.2 Deskripsi AES

AES merupakan sistem penyandian blok yang bersifat *non-Feistel* karena AES menggunakan komponen yang selalu memiliki invers dengan panjang blok 128 bit. Kunci AES dapat memiliki panjang kunci bit 128, 192, 256 bit. Penyandia AES meggunakan proses yang berulang yang disebut dengan ronde. Jumlah ronde yang digunakan AES tergantung denngan panjang kunci yang digunakan. Setiap ronde membutuhkan kunci ronde dan masukan yang diberikan. Relasi antara jumlah ronde dan panjang kunci diberikan Tabel 2.3.

Tabel 2. 3 Hubungan antara jumlah ronde dan panjang kunci AES

Pajang Kunci AES (bit)	Jumlah Ronde(Nr)
128	10
192	12
256	14

2.2.3.3 Unit data AES

AES menggunakan 5 unit ukuran data: *bit*, *byte*, *word*, *blok* dan *state. Bit* merupakan satuan data terkecil, yaitu nilai digit sistem biner. Sedangkan *byte* berukuran 8 *bit*, word berukuran 4 *byte* (32 *bit*0, blok berukuran 16 *byte*(128 *bit*) sedangkan *state* adalah blok yang ditata sebagai matriks *byte* berukuran 4x4 lihat Gambar 2.1.

b = [b0 b1 b2 b3 b4 b5 b6 b7]

Byte

word

b b b b b b b b b b b1 b1 b1 b1 b1 b1

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

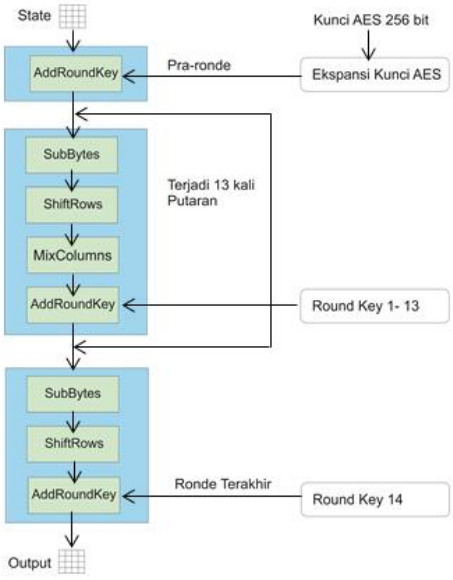
blok

$$\begin{bmatrix} S0,0 = b0 & S0,1 = b4 & S0,2 = b8 & S0,3 = b12 \\ S1,0 = b1 & S1,1 = b5 & S1,2 = b9 & S1,3 = b13 \\ S2,0 = b2 & S2,1 = b6 & S1,2 = b10 & S2,3 = b14 \\ S3,0 = b3 & S3,1 = b7 & S3,2 = b11 & S3,3 = b15 \end{bmatrix}$$

2.2.3.4 Struktur Enkripsi AES

Proses didalam AES merupakan transformasi terhadap *state*. Sebuah teks asli dalam blok (128 *bit*) terlebih dahulu diorganisir sebagai *states.* Enkripsi AES adalah transformasi terhadap *state* secara berulang dalam beberapa ronde. *States* yang menjadi keluaran ronde k menjadi masukan untuk ronde ke-k +1.

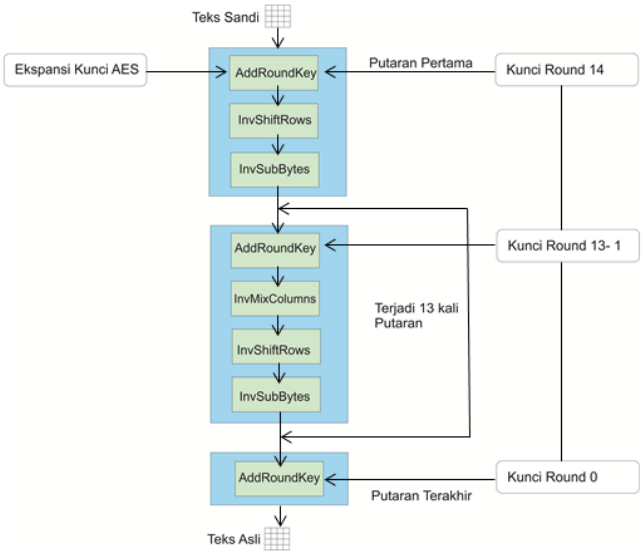
Pada awalnya teks asli direorganisasi sebagai sebuah *state*, kemudian sebelum ronde 1 dimulai blok teks asli dicampur dengan kunci ronde ke-0 (transformasi ini disebut *AddRoundKey*). Setelah itu, ronde ke-1 sampai ronde ke-(Nr-1) dengan Nr adalah jumlah ronde menggunakan 4 jenis transformasi, yaitu *SubBytes, ShiftRows,MixColoumns* dan *AddRoundKey*. Pada ronde terakhir, yaitu ronde ke-Nr dilakukan transformasi serupa dengan ronde lain namun tanpa transformasi *MixColoumns*. Secara garis besar enkripsi AES seperti pada Gambar 2.2.



Gambar 2. 2 Struktur Enkripsi AES

2.2.3.5 Struktur Dekripsi AES

Secara ringkas algoritma dekripsi merupakan kebalikan algoritma enkripsi AES. Algoritma dekripsi AES menggunakan transformasi *invers* semua transformasi dasar yang digunakan pada algoritma enkripsi AES. Setiap transformasi dasar AES memiliki transformasi invers, yaitu : *InvSubBytes, InvShiftRows, dan InvMixColoumns. AddRoundKey* merupakan transformasi yang bersifat *self-invers* dengan syarat menggunakan kunci yang sama. Algoritma dekripsi AES dapat dilihat pada Gambar 2.3.



Gambar 2. 3 Struktur Dekripsi AES

2.2.3.6 Transformasi-transformasi AES

Algoritma enkripsi AES menggunakan 4 transformasi. Substitusi yang disebut dengan *SubBytes*, permutasi yang disebut dengan *ShiftRows*, pencampuran yang disebut dengan *MixColoumns*, dan penambahan yang

disebut dengan *AddRoundKey*. Berikut adalah penjelasan dari ke 4 transformasi enkripsi pada algoritma AES :

a. AddRoundKey

Dalam initial *round*, transformasi *AddRoundKey()* dilakukan terhadap kunci utama. Sedangkan dalam 10 *round* yang lain, proses *AddRoundKey* dilakukan terhadap kunci putaran *(round key).* Proses *AddRoundKey* didefinisikan sebagai operasi XOR antara array state dengan *round key*. Operasi XOR dilakukan pada masing-masing *byte* dalam array sehingga menghasilkan nilai baru pada array hasil dengan ukuran array hasil sama dengan ukuran array state awal dan array *key*, yaitu sebesar 4x4. Hasil untuk masing-masing baris dan kolom pada array state hasil diperoleh dari hasil operasi XOR antara array state awal dengan array key untuk baris dan kolom yang sama.

b. SubBytes

Transformasi *SubBytes()* memetakan setiap byte dari array state dengan menggunakan tabel substitusi S-Box. Tabel S-Box dapat dilihat pada Gambar 2.4 berikut :

Hex		Y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
X	0	63	7C	77	7B	F2	6B	6F	C5	30	1	67	2B	FE	D7	AB	76
	1	Ca	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	4	C7	23	C3	18	96	5	9A	7	12	80	E2	EB	27	B2	75
	4	9	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	0	ED	20	FC	B1	5B	6	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	2	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	OB	DB
	A	E0	E0	32	3A	0A	6	24	5C	C2	D3	AC	62	91	95	A4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	8
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	3	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

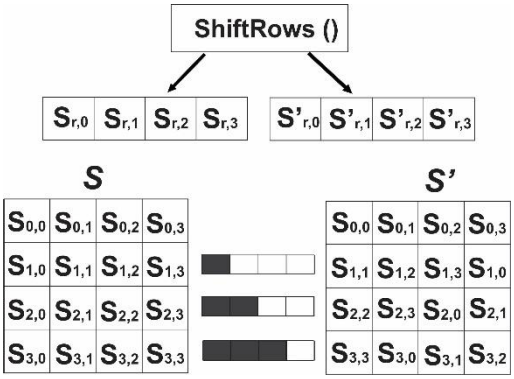
Gambar 2. 4 S-BOX

Cara pensubstitusian adalah sebagai berikut:

untuk setiap *byte* pada array, misalkan S[r,c] = xy, yang dalam hal ini xy adalah digit heksadesimal dari nilai S[r,c], maka nilai substitusinya, yang dinyatakan dengan S'[r,c], adalah elemen di dalam S-Box yang merupakan perpotongan baris x dengan kolom y.

c. ShiftRows

Transformasi *ShiftRows()* melakukan pergeseran secara *wrapping* (siklik) pada 3 baris terakhir dari array state. Jumlah pergeseran bergantung pada nilai baris (r). Baris r = 1 digeser sejauh 1 *byte*, baris r = 2 digeser sejauh 2 *byte*, dan baris r = 3 digeser sejauh 3 *byte*. Baris r = 0 tidak digeser. Gambar *ShiftRows* ditunjukkan pada Gambar 2.5



Gambar 2. 5 Transformasi ShiftRows

d. MixColumns

Transformasi *MixColumns()* dilakukan setelah transformasi *ShiftRows*, merupakan sumber utama dari difusi pada algoritma AES. Difusi merupakan prinsip yang menyebarkan pengaruh satu bit *plaintext* atau kunci ke sebanyak mungkin *ciphertext*. Transformasi MixColumns() mengalikan setiap kolom dari array state dengan polinom a(x) mod (x4 + 1). Setiap kolom diperlakukan sebagai polinom 4 suku pada GF (28). Polinom a(x) yang ditetapkan pada persamaan 1

a (x) = {03}x3 + {01} x2 + {01} x + {02} (1)

Transformasi ini dinyatakan sebagai perkalian matriks seperti pada persamaan 2

s'(x) = a(x) ⊗ s (x) (2)

$$\begin{bmatrix} s'0,c \\ s'1,c \\ s'2,c \\ s'3,c \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_0,c \\ S_1,c \\ S_2,c \\ S_3,c \end{bmatrix}$$

Hasil dari perkalian matriks tersebut, setiap *byte* dalam kolom *array state* akan

digantikan dengan nilai baru. Persamaan matematis untuk setiap *byte* tersebut pada persamaan 3

$s'0,c = (\{02\} \cdot s0,c) \oplus (\{03\} \cdot s1,c) \oplus s2,c \oplus s3,c$

$s'1,c = s0,c \oplus (\{02\} \cdot s1,c) \oplus (\{03\} \cdot s2,c) \oplus s3,c$

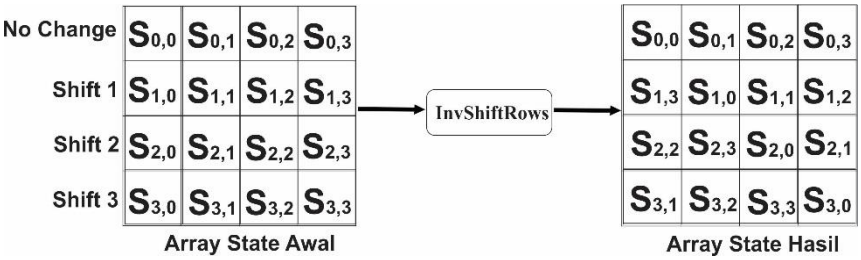
$s'2,c = s0,c \oplus s1,c \oplus (\{02\} \cdot s2,c) \oplus (\{03\} \cdot s3,c)$

$s'3,c = (\{03\} \cdot s0,c) \oplus s1,c \oplus s2,c \oplus (*02\} \cdot s3,c)$ (3)

Transformasi cipher dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES. Transformasi byte yang digunakan pada invers cipher adalah *InvShiftRows, InvSubBytes, InvMixColumns,* dan *AddRoundKey*. Algoritma dekripsi dapat dilihat pada skema berikut ini :

a. InvShiftRows

InvShiftRows adalah transformasi byte yang berkebalikan dengan transformasi *ShiftRows*. Pada transformasi *InvShiftRows,* dilakukan pergeseran bit ke kanan sedangkan pada *ShiftRows* dilakukan pergeseran bit ke kiri. Ilustrasi transformasi *InvShiftRows* terdapat pada gambar berikut.



Gambar 2. 6 Transformasi InvShiftRows

b. InvSubBytes

InvSubBytes juga merupakan transformasi *bytes* yang berkebalikan dengan transformasi *SubBytes.* Pada *InvSubBytes*, tiap elemen pada state dipetakan dengan menggunakan tabel *Inverse S-Box*. Tabel Inverse S-Box akan ditunjukkan dalam Gambar 2.7 berikut.

Hex		Y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
X	0	63	7C	77	7B	F2	6B	6F	C5	30	1	67	2B	FE	D7	AB	76
	1	Ca	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	4	C7	23	C3	18	96	5	9A	7	12	80	E2	EB	27	B2	75
	4	9	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	0	ED	20	FC	B1	5B	6	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	2	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	OB	DB
	A	E0	E0	32	3A	0A	6	24	5C	C2	D3	AC	62	91	95	A4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	8
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	3	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Gambar 2. 7 Tabel Inverse S-Box

c. InvMixColumns

Setiap kolom dalam state dikalikan dengan matrik perkalian dalam AES. Perkalian dalam matrik dapat dituliskan :

$$\begin{bmatrix} s'_0,c \\ s'_1,c \\ s'_2,c \\ s'_3,c \end{bmatrix} = \begin{bmatrix} 0B & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0 \end{bmatrix} \begin{bmatrix} S_0,c \\ S_1,c \\ S_2,c \\ S_3,c \end{bmatrix}$$

Hasil dari perkalian matriks tersebut, setiap byte dalam kolom array state akan

digantikan dengan nilai baru. Persamaan matematis untuk setiap byte tersebut pada persamaan 2.9

s'0,c =({0E}• s0,c)⊕({0B}• s1,c)⊕({0D}•s2,c) ⊕({09}• s3,c)s'1,c =

({09}•s0,c)⊕ ({0E} • s1,c) ⊕ ({0B} • s2,c) ⊕ ({0D}•s3,c)s'2,c =

({0D}•s0,c)⊕ ({09}•s1,c)⊕ ({0E}• s2,c)⊕ ({0B}• s3,c)s'3,c =

({0B}•s0,c) ⊕ ({0D}•s1,c)⊕ ({09}•s2,c) ⊕(*0E} • s3,c) (2.9)

2.2.3.7 Keamanan Sandi AES

Sandi AES ampai saat ini masih dianggap aman untuk digunakan. Banyak sistem komunikasi menggunakan AES sebagai dasar sistem sandinya karena efisien dan aman. Keamanan sistem sandi AES salah satunya disebabkan oleh penggunaan kunci yang besar (128, 192, dan 256 bit) bandingkan dengan sistem sandi DES yang hanya menggunakan 64 bit. Jadi *brute attack* terhadap sistem sandi AES 256 bit memiliki kunci 2^{256} yang merupakan nilai yang sangat besar.

Selain itu, sandi AES mencapai karakteristik difusi dan confusion sehingga mendekati sistem sani ideal. Karakter difusi dan confusion dapat dibuktikan dengan adanya efek *acalanche* pada kunci lemah atau setengah lemah sehingga apapun kunci yang digunakan keamanan AES tetap aman[11].

2.2.4. Steganografi

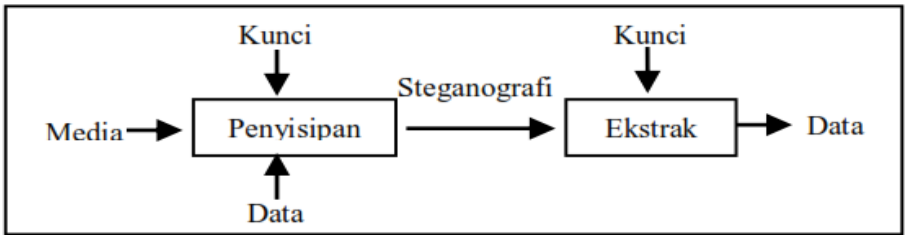
Steganografi merupakan seni komunikasi rahasia dengan menyembunyikan pesan pada objek yang tampaknya tidak berbahaya. Keberadaan pesan steganografi adalah rahasia. Istilah Yunani ini berasal dari kata *Steganos*, yang berarti tertutup dan Graphia, yang berarti menulis[6]**.** Istilah steganografi berasal dari bahasa Yunani, yaitu *steganos* yang berarti penyamaran atau penyembunyian dan *graphein* yang berarti tulisan. Jadi, steganografi bisa diartikan sebagai seni menyembunyikan pesan dalam data lain tanpa mengubah data yang ditumpanginya tersebut sehingga data yang ditumpanginya sebelum dan setelah proses penyembunyian hampir terlihat sama[3].

Secara umum, terdapat dua proses utama didalam steganografi. Yaitu proses penyisipan (*Embedding/encoding)* untuk menyembunyikan pesan dan ekstraksi (*extraction/decoding*) untuk mengekstraksi pesan yang disembunyikan. Pesan dapat berupa *plaintext, chipertext,* citra atau apapun yang dapat ditempelkan ke dalam *bit-strem.*

Embedding adalah proses menyisipkan pesan ke dalam *file* yang belum dimodifikasi, yang disebut media *cover* (*cover object*). Kemudian media *cover* dan pesan yang ditempelkan membuat media *stego* (*stego object*).

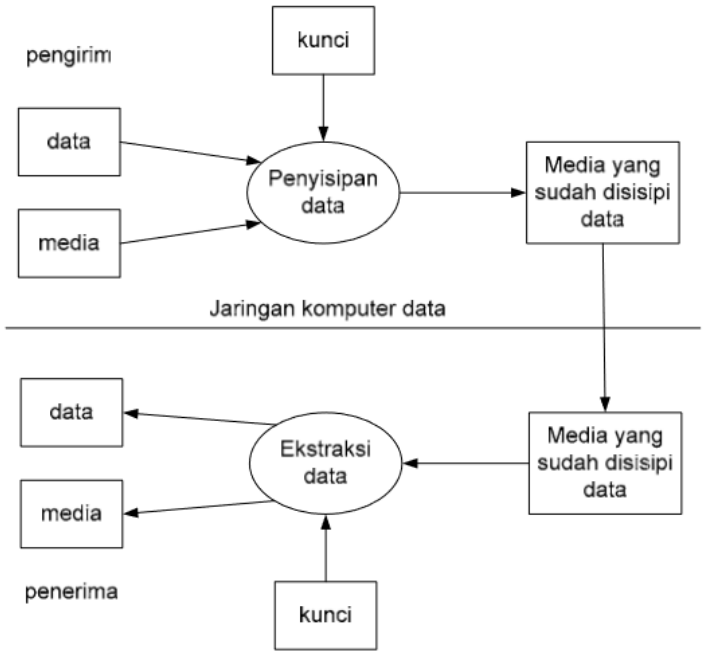
Extraction adalah proses menguraikan pesan yang tersembunyi dalam media *stego*. Suatu kunci khusus (*stego key*) juga dapat digunakan secara tersembunyi, pada saat penguraian selanjutnya dari pesan. Ringkasnya steganografi adalah teknik menanamkan *embedded message* pada suatu *cover object*, dimana hasilnya berupa *stego object*. Pihak yang terkait dengan steganografi antara lain

embeddor, *extractor*, dan *stegoanalyst*. Skema penyisipan dan ekstraksi dalam steganografi dapat dilihat dalam ilustrasi gambar 2.8



Gambar 2. 8 Skema penyisipan dan ekstraksi dalam steganografi

Steganografi digital menggunakan media digital wadah penampungan, misalnya citra, suara, teks, dan video. Sedangkan data rahasia yang disembunyikan dapat berupa pesan atau file berkas apapun. Media yang telah disisipi data rahasia disebut stegomessage. Proses penyembunyian data ke dalam media disebut penyisipan (*embedding*), sedangkan proses sebaliknya disebut ekstraksi. Proses tersebut dapat dilihat pada ilustrasi gambar 2.9.



2.2.4.1 Metode Steganografi Least Significant Bit (LSB)

Least Significant Bit adalah salah satu metode untuk menyembunyikan pesan dalam media digital dengan cara menyisipkan pesan tersebut pada satu bit paling kanan ke *pixel file* obyek. Proses ekstrasi dilakukan dengan 2 tahap, pertama untuk memperoleh *ciphertext* diambil bit-bit paling belakang dari *stego image*. Kedua, untuk memperoleh *cover image*, tambahkan satu bit paling belakang pada *pixel-pixel* sisa tahap pertama dengan bit yang sama dengan bit paling belakang *cover image.* Untuk LSB (*Least Significant Bit*) *Embedding Process Berikut ini* penyisipan data pada file citra bitmap greyscale 8 bit per pixel dengan skala 0 sampai 255, atau dengan format biner 00000000 sampai 11111111. Misalnya pixel-pixel citra yang akan digunakan sebagai wadah (cover image) adalah : (01001101 00101110 10101110 10001010 10101111 10100010 00101011 10101011) pixel-pixel cover image tersebut akan berubah menjadi : (0100110<u>0</u> 0010111<u>1</u> 1010111<u>0</u> 1000101010101110 1010001<u>0</u> 0010101<u>0</u> 1010101<u>1</u>) Perubahan yang tidak significant ini tidak akan terdeteksi oleh mata manusia.

Untuk proses LSB *(Least Significant Bit) Extracing Process* Proses ekstraksi dilakukan dengan 2 tahap. Pertama untuk memperoleh *ciphertext* diambil bit-bit paling belakang dari stego *image*. Kedua, untuk memperoleh *cover image*, tambahkan satu bit paling belakang pada *pixel-pixel* sisa tahap pertama dengan bit yang sama dengan bit paling belakang *cover image.*

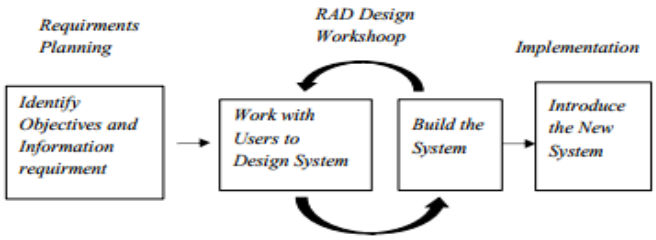
Dari proses penyisipan karakter ‘A’ diperoleh stego *image* (0100110<u>0</u> 0010111<u>1</u> 1010111<u>0</u> 1000101<u>0</u> 1010111<u>0</u> 1010001<u>0</u> 0010101<u>0</u> 1010101<u>1</u>) dengan mengambil bit-bit paling belakang dari stego image tersebut maka

diperoleh karakter ‘A’ (01000001) dan pixel-pixel sisa tahap pertama cover image (0100110? 0010111? 1010111? 1000101? 1010111? 1010001? 0010101? 1010101?). Menggunakan tahap kedua diperoleh cover image berikut: (01001101 00101110 10101110 10001010 10101111 10100010 00101011 10101011)[12].

2.2.5. Rapid Aplication Development (RAD)

Rapid Application Development (RAD) adalah salah satu metode pengembangan sistem informasi dengan waktu yang relatif singkat. Untuk pengembangan suatu sistem informasi yang normal membutuhkan waktu minimal 180 hari, akan tetapi dengan menggunakan metode RAD suatu sistem dapat diselesaikan hanya dalam waktu 30-90 hari. Tujuan utama dari semua metode sistem development adalah memberikan suatu sistem yang dapat memenuhi harapan dari para pemakai, maka para pemakai bisa menjadi bagian dari keseluruhan proses pengembangan sistem dengan bertindak sebagai pengambil keputusan pada setiap tahapan pengembangan[4]. Dengan menggunakan RAD, maka keterlibatan pengguna menjadi semakin meningkat yang akhirnya dapat meningkatkan kepuasan penggua terhadap sistem yang dikembangkan dan menghasilkan suatu sistem dengan cepat.

2.2.4.1. Tahapan-Tahapan Pada RAD



a. Rencana Kebutuhan *(Requirement Planning)* Pada tahapan ini, *user* dan *analyst* melakukan identifikasi tujan dari aplikasi ataupun sistem dan melakukan identifikasi kebutuhan yang diperlukan.

b. Proses Desain *(Design Workshop)* pada tahapan ini adalah melakukan proses desain dan perbaikan-perbaikan apabila masih terdapat ketidak sesuaian desain antara *user* dan *analyst.* Proses desain menggunakan UML (*unifled modelling language*).

c. Implementasi *(Implementation)* Setelah desain dari sistem yang akan dibuat sudah disetujui baik itu oleh *user* dan *analyst.* Maka pada tahapan ini *programmer* mengembangkan desain menjadi suatu program.

2.2.6. Aplikasi Pendukung

2.2.6.1. Android Studio

Android Studio merupakan sebuah *Integrated Development Environment* (IDE) untuk platform Android. Android Studio ini diumumkan pada tanggal 16 Mei 2013 pada Konferensi Google I/O oleh Produk Manajer Google, Ellie Powers. Android studio bersifa *free* dibawah *Apache License* 2.0. Android Studio awalnya dimulai dengan versi 0.1 pada bulan mei 2013, Kemudian dibuat versi beta 0.8 yang dirilis pada bulan juni 2014. Yang paling stabil dirilis pada bulan Desember 2014, dimulai dari versi 1.0. Berbasiskan JetBrainns' IntelliJ IDEA, Studio di desain khusus untuk Android *Development.* Ini sudah bisa di download untuk Windows, Mac OS X, dan Linux[13].

2.2.6.2. Java

Java memiliki beberapa keunggulan bila dibandingkan dengan bahasa pemrograan lainnya, diantaranya adalah sebagai berikut :

a) *Java* bersifat sederhana dan relatif mudah

Java di modelkan sebagian dari bahasa C++, namun dengan memperbaiki beberapa karakteristik C++, seperti mengurangi kopleksitas beberapa fitur, penambahan fungsionalitas, serta penghilangan beberapa aspek pemicu ketidak stabilan sistem pada C++. Sebagai contoh *java* menggantikan konsep pewarisan lebih dari satu (*multiple inheritance*) dengan *interface*, menghilangkan konsep *pointer* yang sering membingungkan, otomatisasi sistem alokasi memori dan sebagainya. Ini membuat *java* menjadi relatif sederhana dan mudah dipelajari dibandingkan dengan bahasa pemrograman lainnya.

b) *Java* berorientasi pada objek (*object oriented*)

Java adalah bahasa pemrograman yang beorientasi objek (OOP), bukan seperti pascal, basic atau C yang berbasis prosedural. Dalam memecahkan masalah, *java* membagi program menjadi objek – objek, kemudian memodelkan sifat dan tingkah laku masing – masing. Selanjutnya, *java* mengatur dan menentukan interaksi antara objek yang satu dengan yang lainnya.

c) *Java* bersifat terdistribusi

Pada dekade awal perkembangan PC (*Personal Conputer*), komputer hanya bersifat *workstation* tunggal, tidak terhubung satu sama lain. Saat ini, sistem

terkomputerisasi cenderung terdistribusi, mulai dari *workstation clien, email server, database sever, web server, proxy server* dan sebagainya.

d) *Java* bersifat *multi-flatform*

Pada umumnya, program yang dibuat dan di kompile disuatu *flatform* tersebut. *Java* bersifat *multi-latform,* yakni dapat diterjemahkan oleh *java interpreter* pada berbagai sistem operasi.

e) *Java* bersifat *multi-thread*

Thread adalah proses yang dapat dikerjakan oleh program dalam satu waktu. *Java* bersifat *multi-thread,* artinya dapat mengerjakan beberapa proses dalam waktu hampir bersamaan[14]**.**

2.2.7. UML (*Unified Modeling Language*)

UML Merupkan singkatan dari *Unified Modeling Language* yang berarti bahasa standar. UML memiliki sintaks dan semantik. Ketika membuat model menggunakan konsep UML ada aturan – aturan yang harus diikuti. UML merupakan metodologi yang paling sering digunakan saat ini untuk analisa dan perancangan sistem dengan metodologi berorientasi objek mengadaptasi maraknya penggunaan bahasa "pemrograman berorientasi objek" (OOP)[14]**.** UML bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk ;

a. Merancang perangkat lunak.
b. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
c. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.

d. Mendokumentasi sistem yang ada, proses – proses dan organisasinya.

2.2.6.1. Use Case Diagram

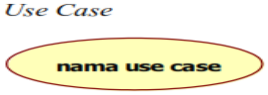
Use case merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi – fungsi itu.

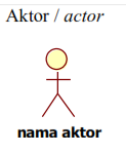
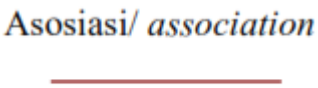
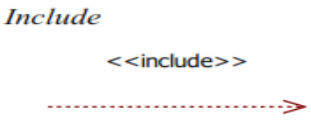
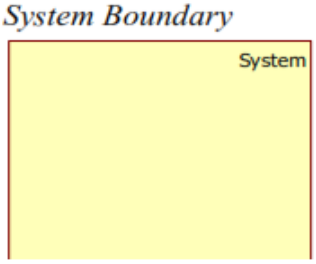
Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*[15].

a. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
b. *Use case* merupakan fungsional yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau aktor.

Berikut adalah tabel simbol-simbol yang ada pada *use case diagram*.

Tabel 2. 4 Simbol - simbol *use case diagram*

Simbol	Deskripsi
	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal fase nama use case

	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri.
	Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan aktor.
	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dujalankannya use case ini
	Disimbolkan dalam bentuk kotak yang mewadahi use case, sebagai representasi dari ruang lingkup sistem yang akan dikembangkan. Biasanya digunakan apabila terdapat beberapa alternatif sistem, yang dapat dijadikan pilihan

2.2.6.2. Class Diagram

Class diagram menggambarkan struktur aplikasi berorientasi objek dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun aplikasi. Kelas memiliki apa yang disebut atribut metode atau operasi. Atribut merupakan variabel – variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi – fungsi yang dimiliki oleh suatu kelas. Kelas – kelas yang ada pada struktur sistem harus dapat melakukan fungsi sesuai dengan kebutuhan sistem.

Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis – jenis kelas seperti berikut[15] :

a. Kelas *main*

Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

b. Kelas yang menangani tampilan sistem

Kelas yang mendefinisikan dan mengatur tampilan ke pemakai

c. Kelas yang diambil dari pendefinisian *use case*

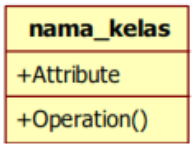
Kelas yang menangani fungsi – fungsi yang harus ada diambil dari pendefinisian *use case* tersebut.

d. Kelas yang diambil dari pendefinisian data

Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data Jenis – jenis kelas diatas juga dapat digabungkan satu sama lain sesuai dengan pertimbangan yang dianggap baik asalkan fungsi – fungsi yang sebaiknya ada pada struktur kelas tetap ada. Susunan kelas juga dapat ditambahkan kelas utilitas seperti koneksi ke *database*, membaca *file* teks, dan lain sebagainya sesuai dengna kebutuhan.

Berikut adalah tabel simbol-simbol yang ada pada pada diagram kelas.

Tabel 2. 5 *Simbol - Simbol* Class Diagram

Simbol	Deskripsi
Kelas 	Kelas pada struktur sistem
Asosiasi/ *association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity
Public 	sebuah class dapat dipanggil oleh siapa saja

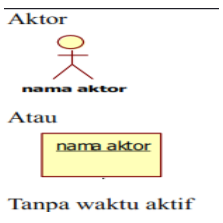
Private 	sebuah class tidak dapat digunakan/ dipanggil dari luar class yang bersangkutan
Protected 	hanya dapat dipanggil oleh kelas yang mewarisinya.

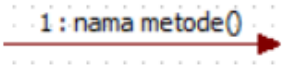
2.2.6.3. Squence Diagram

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek – objek yang terlibat dalam sebuah *use case* beserta metode – metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Banyaknya diagram sekuen yang harus digambar adalah sebanyak pendefinisian *ue case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *user case* yang didefinisikan maka diagram sekuen yang harus didefinisikan semakin banyak[15].

Berikut adalah tabel simbol-simbol yang ada pada squence diagram.

Tabel 2. 6 Simbol - simbol squence diagram

Simbol	Deskripsi
	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya

	dinyatakan dengan kata benda di awal frase nama aktor
Garis hidup/ *lifeline*	Menyatakan kehidupan suatu objek
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/ metode yang ada pada objek lain atau dirinya sendiri, arah panah mengarah pada objek yang memiliki operasi/ metode. Karena ini memanggil operasi/ metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi

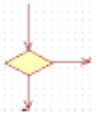
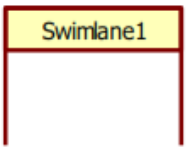
2.2.6.4. Activity Diagram

Menggambarkan *workflow* (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktifitas menggambarkan aktifitas sistem bukan apa yang dilakukan oleh aktor. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal – hal berikut[15] :

a. Rancangan proses bisnis dimana setiap urutan aktifitas yang digambarkan merupakan proses bisnis sistem yang diidentifikasikan.
b. Urutan atau pengelompokan tampilan dari sistem/ *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antar muka tampilan.
c. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.

Berikut adalah tabel simbol-simbol yang ada pada diagram activity diagram.

Tabel 2. 7 Simbol simbol activity diagram

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memilik sebuah status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan/ *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimline 	Memisahkan organisasi bisnis yang bertanggunag jawab terhadap aktivitas yang terjadi

2.2.8. Black-box Testing

Pengujian *Black-Box* berfokus pada persyaratan fungsional perangkat lunak[16]. Dengan demikian, pengujian *Black-Box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Pengujian *Black-Box* bukan merupakan alternatif dari teknik *White-Box* tetapi merupakan

pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan dari metode *White-Box*.

Pengujian *Black-Box* berusaha menemukan kesalahan dalam kategori sebagai berikut :

a. Fungsi-fungsi yang tidak benar atau hilang.
b. Kesalahan *interface*.
c. Kesalahan dalam struktur data atau akses database eksternal.
d. Kesalahan kinerja.
e. Inisialisasi dan kesalahan terminasi.

Pengujian *Black-Box* ini merupakan pengujian berdasarkan fungsi dari program. Tujuan dari *Black-Box* ini adalah untuk menemukan kesalahan fungsi pada program.