BAB 2
LANDASAN TEORI

2.1 Kriptografi

Kriptografi merupakan ilmu dan seni untuk menjaga keamanan pesan, kriptografi juga merupakan ilmu yang mempelajari teknik-teknik matematika yang menghubungkan dengan aspek keamanan informasi seperti kerahasiaan, integritas data serta otentikasi (Munir, 2006). Berikut ini merupakan empat tujuan mendasar dari ilmu kriptografi ini yang merupakan aspek keamanan informasi yaitu:

1. Kerahasiaan

   Kerahasiaan adalah layanan yang digunakan untuk menjaga isi dari informasi dari siapapun kecuali yang dimiliki otoritas atau kunci rahasia untuk membuka/mengupas informasi yang telah disandi.

2. Integritas data

   Integirias data adalah berhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, diantaranya pennyisipan, penghapusan, dan pensubsitusian data lain kedalam data data asli atau data sebenarnya.

3. *Autentikasi*

   Autentikasi adalah berhubungan dengan identifikasi atau pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirim melalui kanal harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lainya.

2.1.1 Algoritma Kriptografi

Definisi terminologinya Algoritma adalah urutan langkah-langkah logis untuk menyelesaikan masalah yang disusun secara matematis. Sedangkan algoritma kriptografi merupakan langkah langkah logis bagaimana menyembunyikan dari orang yang tidak berhak atas pesan tersebut (Munir, 2006). Algoritma kriptografi terdiri dari tiga fungsi yaitu:

1. Enkripsi

   Enkripsi merupakan hal yang sangat penting dalam kriptografi yang merupakan pengamanan data yang dikirimkan terjaga rahasianya. Pesan asli disebut *Plaintext* yang disebut menjadi kode-kode tidak dimengerti. Enkripsi bisa diartikan dengan *chiper* atau kode. Sama halnya dengan kita

tidak mengerti akan sebuah kata, maka kita akan melihanya didalam kamus-kamus atau daftar istilah-istilah. Beda halnya dengan enkripsi, untuk mengubah *plaintext* kebentuk *ciphertext* kita menggunakan algoritma yang dapat mengkodekan data yang kita ingini.

2. Dekripsi

   Dekripsi merupakan kebaikan dari enkripsi pesan yang telah di enkripsi dikembalikan kebentuk asalnya (*Plaintext*) disebut dengan dekripsi pesan. Algoritma yang digunakan untuk dekripsi tentu berbeda dengan yang digunakan untuk enkripsi.

3. Kunci

   Keamanan dari kriptografi modern hanya dengan merahasiakan kunci yang dimiliki orang lain, tanpa harus merahasiakan algoritma itu sendiri. kunci berfungsi sama dengan *password*. Jika keseluruhan dari keamanan algoritma tergantung pada kunci yang dipakai maka, algoritma ini bisa di publikasikan dan dianalisis oleh orang lain. jika algoritma ini bisa dipecahkan dalam waktu singkat oleh orang lain maka, algoritma tersebut belum aman untuk digunakan (Munir, 2006).

2.1.2 Macam-macam Algoritma Kriptografi

Algoritma dibagi menjadi 3 bagian berdasarkan dari kunci yang dipakai, yaitu:

1. Algoritma Simetri

   Munir (2006) dalam bukunya menyatakan algoritma ini juga disebut algoritma klasik karena menggunakan kunci yang sama untuk kegiatan enkripsi dan dekripsinya. Mengirim pesan dengan algoritma ini, sipenerima harus diberitahu kunci dari pesan tersebut untuk bisa mendekripsi dari pesan yang dikirim. Algoritma yang memakai kunci asimetri diantaranya adalah:

   (a) *Data Encryption Standart* (DES)
   (b) RC2, RC4, RC5, RC6
   (c) *International Data Encription Algoritma* (IDEA)
   (d) *Advance Encription Standart* (AES)
   (e) *One Time Pad* (OTP)
   (f) Dan lain sebagainya

2. Algoritma Asimetri

   Algoritma asimetri sering disebut juga algoritma kunci public, dengan arti lain kunci yang digunakan untuk melakukan enkripsi dan dekripsinya berbeda (Munir, 2006). Pada algoritma asimetri kunci dibagi dua bagian:

(a) Kunci Umum (*Public Key*): Kunci yang boleh semua orang tau (dipublikasikan).

(b) Kunci Pribadi (*Private Key*): kunci yang dirahasiakan (hanya diketahui oleh satu orang).

Menurut Munir (2006), dalam bukunya menyatakan Kunci kunci tersebut saling berhubungan satu dengan yang lainya. Dengan kunci public orang dapat mengenkripsi pesan tapi tidak bisa mendeskripsikanya, hanya orang yang memiliki kunci pribadi yang dapat mendeskripsi pesan tersebut. Algoritma yang memiliki kunci publik diantaranya adalah:

(a) *Digital Signature Algorithm (DSA)*

(b) *RSA*

(c) *Diffie-Hellman(DH)*

(d) *Elliptic Curve Criyptography (ECC)*

3. *Hash Function* (Fungsi hash)

Menurut Munir (2006), Fungsi *hash* sering disebut fungsi hash atau satu arah (*one-way function*), *message digest*, *fingerprint*, fungsi kompresi dan *message authentication code* (MAC), hal ini merupakan satu fungsi matematika yang mengambil input panjang *variable* dan mengubahnya menjadi urutan biner dengan panjang yang tetap. Fungsi has biasaya diperlukan jika ingin membuat sidik jari dari suatu pesan. Sidik jari pada pesan merupakan suatu tanda yang menandakan bahwa pesan tersebut benar-benar dari orang yang di inginkan.

2.2 *Advanced Encryption Standart* (AES)

Menurut Munir (2006), menyatakan *Advanced Encryption Standart* (AES) adalah algoritma enkripsi yang digunakan untuk melindungi data elektronik. Algoritma dari AES merupakan blok *cipher* yang dapat melalukan enkripsi dan dekripsi pada informasi atau data. Data yang diproses pada AES memiliki panjang 128-bit dan dapat diproses menggunakan tiga macam kunci yaitu kunci sepanjang 128-*bit* atau dikenal dengan AES-128 dan 192-*bit* atau dikenal dengan AES-192, dan 256-bit atau dikenal dengan AES-256. Table berikut meresumekan perbedaan ketiga versi AES tersebut dapat dilihat pada Tabel 2.1.

Tabel 2.1. Versi AES

Type	**Panjang Kunci**	**Ukuran Blok**	**Jumlah Putaran**
AES-128	4	4	10
AES-192	6	4	12

Tabel 2.1 Versi AES (Tabel lanjutan...)

No	Simbol	DeskripsiHasil	
AES-256	8	4	14

1 word = 32 *bit*, Secara *de-facto*, ada dua varian AES, yaitu AES-128 dan AES-256, karena akan sangat jarang pengguna yang menggunakan kunci 192 *bit*. Karena AES mempunyai panjang kunci paling sedikit 128 *bit*, maka AES tahan terhadap serangan *exhaustive key search* dengan teknologi saat ini. Dengan panjang kunci 128-*bit*, maka terdapat sebanyak 2128 =3,4 x 1038 kemungkinan kunci. Jika digunakan komputer tercepat yang dapat mencoba 1 juta kunci setiap detik, maka akan dibutuhkan waktu selama 5,4 x 1018 tahun untuk mencoba seluruh kemungkinan kunci (Munir, 2006). Kriptografi AES memiliki beberapa kelebihan diantaranya adalah sebagai berikut:

1. Tidak ada serangan yang diketahui yang bisa memecahkanya.
2. Menggunakan *sbox non linear*.
3. Mempunyai suatu security margin yang cukup.
4. *Enkripsi* dan *dekripsi* sangat baik untuk semua *platform* yang ada, yang meliputi 8 *bit* dan 64 *bit platform*.
5. Tidak memakan banyak sumber daya *processor*, Karena aes memiliki *performance* yang baik.
6. Setup dalam waktu yang singkat.
7. *Enkripsi* menggunakan memnggunakan sumber daya RAM dan ROM yang sedikit.
8. *Schedule* untuk *enkripsi* dan *dekripsi* terpisah.
9. Ukuran kunci 192 dan 256 tidak mempengaruhi kecepatan dalam setup
10. Sumber daya *hardware* yang sedikit.
11. Mempunyai sistem pertahanan yang baik.
12. Teknik dari algoritma AES jika terkena serangan tidak menyebabakan kerusakan yang berarti.
13. Mempunyai potensial yang sempurna untuk *single block* enkripsi *parallel*.

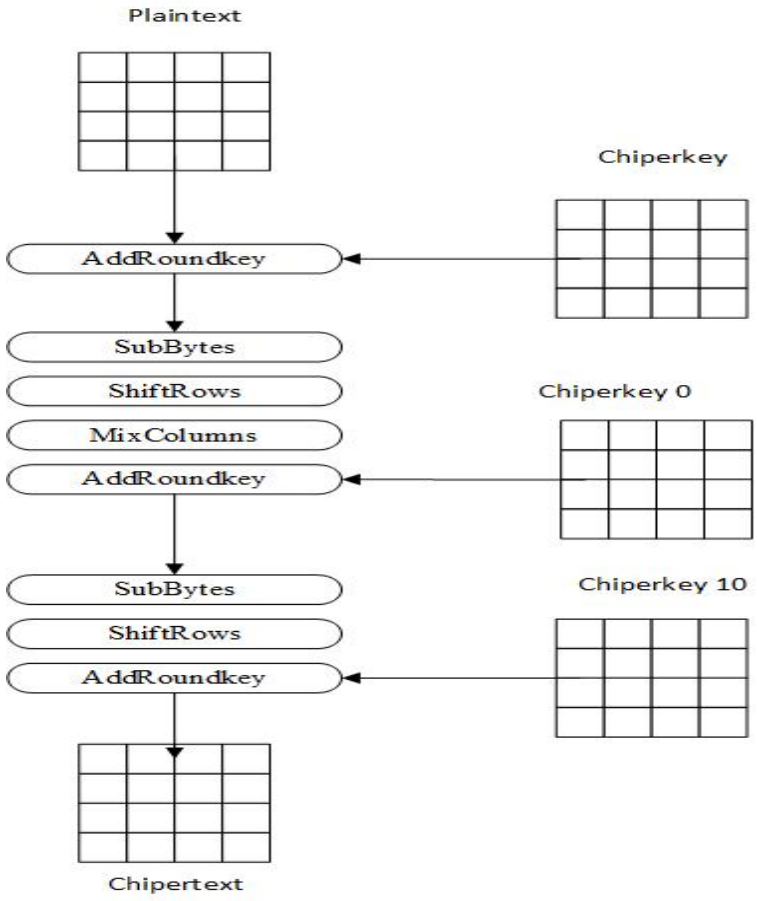
2.3 Proses *Enkripsi Advanced Encryption Standart*

Menurut Munir (2006), *Advanced Encryption Standart* menggunakan subtitusi dan permutasi, dan sejumlah putaran (*cipher* berulang) – setiap putaran menggunakan kunci internal ulang berbeda (kunci setiap putaran *roundkey*). Tetapi tidak sepserti DES yang berorientasi bit, AES beroperasi dalam *orientasi byte* (untuk memangkuskan implementasi algoritma kedalam *software* dan *hardware*). Algoritma

AES mempunyai 3 parameter:

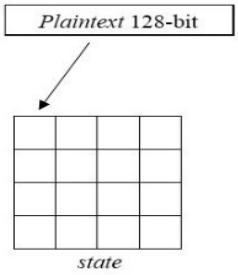
1. *Plaintext*: *array* yang berukuran 16-*byte*, yang berisi data masukan.
2. *Chipertext*: *array* yang berukuran 16-*byte*, berisi hasil enkripsi.
3. *Key*: *array* yang berukuran 16-*byte*, yang berisi kunci *ciphering* (disebut juga *cipher key*).

Ilustrasi proses enkripsi dapat digambarkan seperti pada Gambar 2.1.



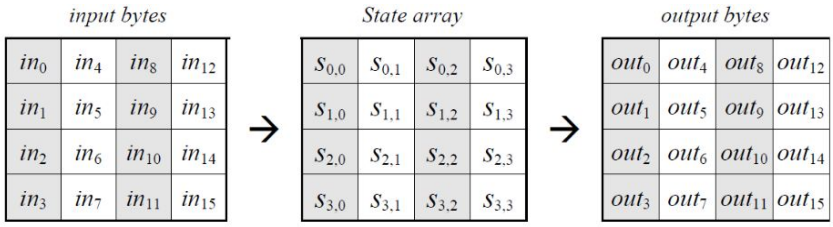
Gambar 2.1. Ilustrasi proses *enkripsi* AES (Munir, 2006)

Menurut Munir (2006) dalam bukunya menyatakan Dengan 16 *byte*, maka blok data dan kunci yang berukuran 128-bit dapat disimpan di dalam ketiga *array* tersebut (128 = 16 x 8). Selama kalkulasi *plaintext* menjadi *ciperteks*, status sekarang dari data disimpan didalam *array of byte* dua dimensi, *state*, yang berukuran NROWS x NCOLS. Untuk blok data 128-*bit*, ukuran *state* adalah 4 x 4. Elemen *array state* diacu sebagai S[r,c] Pada AES-128, NB = 128/32 = 4) berikut ini merupakan ilustrasi penerapan *state* dapat dilihat pada Gambar 2.2.



Gambar 2.2. *State* (Munir, 2006)

Pada awal enkripsi, 16-*byte* data masukan disalin kedalam *array state* (direalisasikan oleh fungsi *Copy Palintext To State* seperti di ilustrasikan pada Gambar 2.3.



Gambar 2.3. Proses *input bytes, state array*, dan *output bytes* (Munir, 2006)

Enkripsi merupakan proses mengubah suatu pesan asli (*plaintext*) menjadi suatu pesan dalam bahasa sandi (*ciphertext*).

C = E (M) (1) dimana

M = Pesan asli (*Plaintext*)

E = Proses enkripsi dengan *Key Private*

C = *Chipertext* (*Plaintext* yang terenkripsi AES)

Garis besar Algoritma AES Rijndael yang beroperasi pada blok 128-*bit* dengan kunci 128-*bit* adalah sebagai berikut (di luar proses pembangkitan *round key*).

1. *AddRoundKey*: melakukan XOR antara *state* awal (*plainteks*) dengan *cipherkey*. Tahap ini disebut juga *initial round*.
2. *Round*: Putaran sebanyak Nr – 1 kali. Proses yang dilakukan pada setiap putaran adalah:
   (a) *SubBytes*: *substitusi byte* dengan menggunakan tabel substitusi (S-box).
   (b) *ShiftRows*: pergeseran baris-baris *array state* secara *wrapping*.
   (c) *MixColumns*: mengacak data di masing-masing kolom *array state*.
   (d) *AddRoundKey*: melakukan XOR antara *state* sekarang *round key*.

3. *Final round*: proses untuk putaran terakhir:
   (a) *SubBytes*
   (b) *ShiftRows*
   (c) *AddRoundKey*

2.3.1 *AddRoundKey*

Menurut Munir (2006), Pada proses enkripsi dan dekripsi AES proses *AddRoundKey* sama, sebuah *round key* ditambahkan pada *state* dengan operasi XOR setiap *byte* dari *blok chiper* awal dikombinasikan dengan blok kunci ronde dengan operasi *bitwise* XOR.

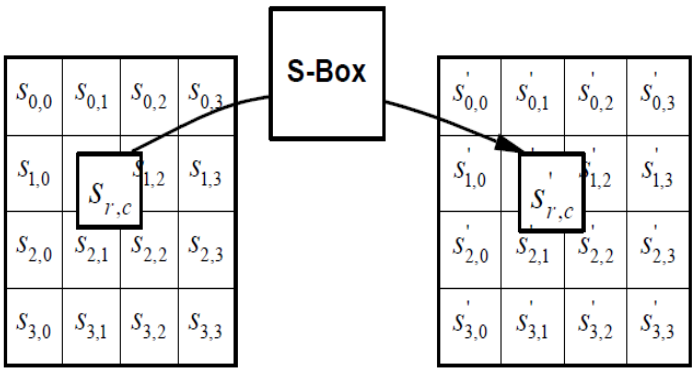
2.3.2 *SubBytes*

Menurut Munir (2006), SubBytes merupakan transformasi *byte* dimana setiap elemen pada state akan dipetakan dengan menggunakan sebuah tabel substitusi (*S-Box*). Untuk lebih jelasnya dapat dilihat pada Gambar 2.4.

		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	01	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	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	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	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	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. Rijndael s-box (Munir, 2006)

Untuk setiap *byte* pada *array state*, misalkan S[r, c] = xy, yang dalam hal ini xy adalah digit heksadesimal dari nilai *state* S[r, c], maka nilai substitusinya adalah elemen di dalam tabel subtitusi yang merupakan perpotongan dari baris x dengan kolom y, ilustrasi langkah *subbyte* dapat dilihat pada Gambar 2.5.

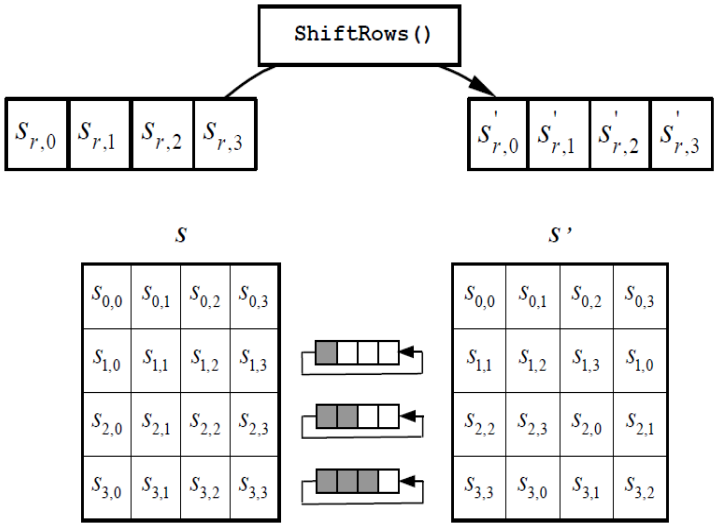


Gambar 2.5. Langkah *subbytes* (Munir, 2006)

Gambar 2.5 mengilustrasikan pengaruh pemetaan *byte* pada setiap *byte* dalam *state*.

2.3.3 *ShiftRows*

Menurut Munir (2006) langkah *ShiftRows* yaitu operasi menggeser pada setiap elemen blok yang dilakukan per barisnya. Untuk baris pertama tidak dilakukan operasi pergeseran blok, baris kedua dilakukan pergeseran 1 *byte*, baris ketiga dilakukan pergeseran 2 *byte*, dan baris keempat dilakukan pergeseran 3 *byte*. Pergeseran yang dilakukan adalah pergeseran kekiri dan apabila melewati batas kiri blok maka kembali ke posisi kanan dari blok tersebut. ilustrasi *Shiftrow* dapat dilihat pada Gambar 2.6.

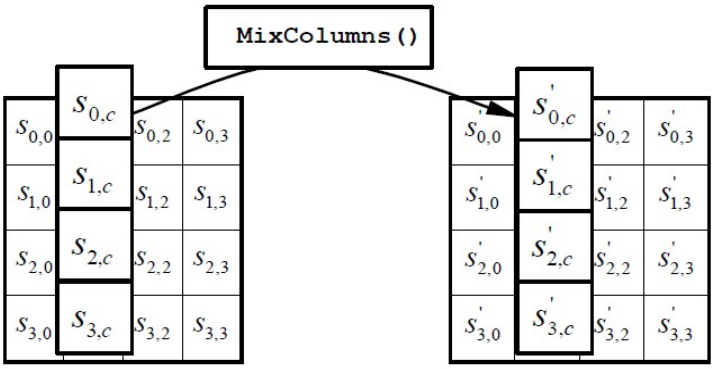


Gambar 2.6. Langkah *shiftrows* (Munir, 2006)

2.3.4 *MixColumn*

Blok *cipher* dari langkah *SwiftRow* ini menuju ke langkah *MixColumn*. Pada langkah *MixColumn* dilakukan operasi perkalian *bitwise* XOR tiap tiap elemen dari

blok *cipher*. Perkalian yang dilakukan seperti perkalian *dot product* matriks pada umumnya lalu hasilnya dimasukan pada blok *cipher* yang baru. Langkah *MixColumn* ini dilakukan sebanyak n-1 (n=10 untuk 128-*bit* atau 10 putaran, n= 12 untuk 192-*bit* 12 putaran, dan n = 14 untuk 256-*bit* 14 putaran) dikarenakan pada awal proses AES dilakukan langkah *AddRoundKey* sehingga pada akhir proses enkripsi AES dilakukan langkah *MIXColumn* agar sesuai dengan langkah deskripsinya (Munir, 2006). Ilustrasi mixclumn dapat dilihat pada Gambar 2.7.



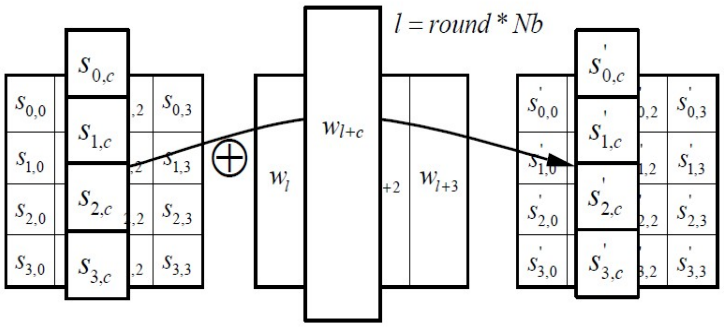
Gambar 2.7. Langkah *mixcolumn* (Munir, 2006)

MixColumns mengoperasikan setiap elemen yang berada dalam satu kolom pada *state*.Secara lebih jelas, transformasi *mixcolumns* dapat dilihat pada Gambar 2.8.

$$\begin{bmatrix} S'_{0,C} \\ S'_{1,C} \\ S'_{2,C} \\ S'_{3,C} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,C} \\ S_{1,C} \\ S_{2,C} \\ S_{3,C} \end{bmatrix}$$

Gambar 2.8. Langkah perkalian *mixcolumn* (Munir, 2006)

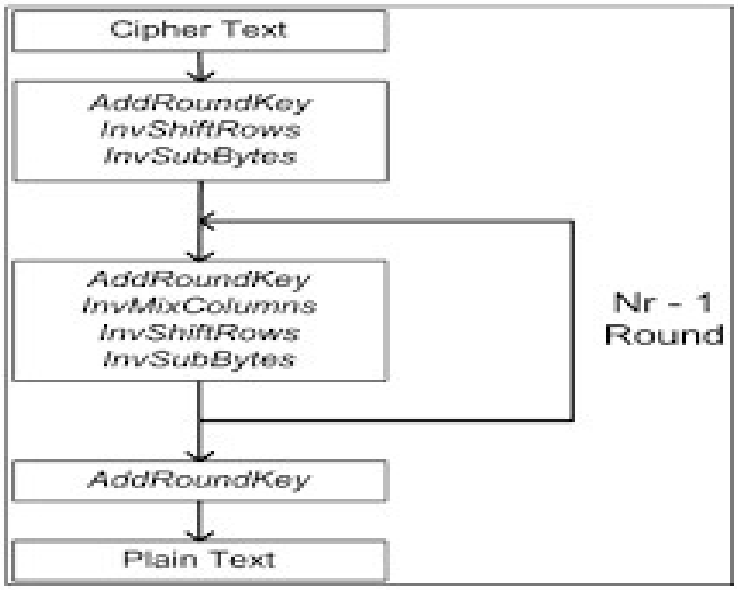
Setelah langkah *Mix Column* selesai dilakukan maka selanjutnya kembali ke langkah *AddRoundKey* seperti pada langkah inisiasi, yaitu dengan blok pada kunci ronde dikalikan dengan blok *cipher* saat ini menggunakan *bitwise* XOR, sehingga menghasilkan kunci ronde yang baru (Munir, 2006). Langkah *AddRoundKey* dapat dilihat pada Gambar 2.9.



Gambar 2.9. Langkah *addroundkey* (Munir, 2006)

2.4 Proses Dekripsi *Advanced Encryption Standart*

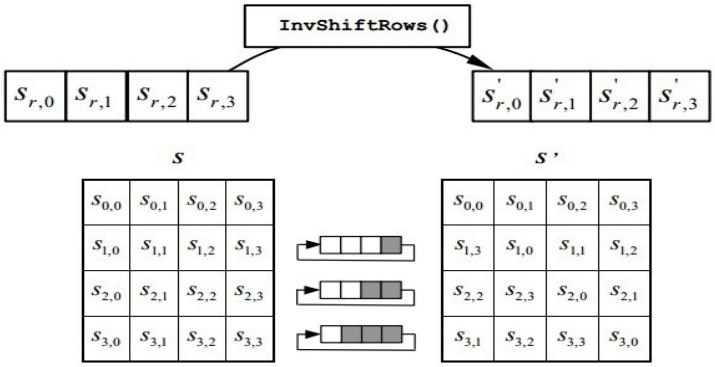
Menurut Ariyus (2006)), Transformasi *cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES. Transformasi *byte* yang digunakan pada *invers cipher* adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*. Algoritma dekripsi dapat dilihat pada Gambar 2.10.



Gambar 2.10. Ilustrasi proses *dekripsi* AES (Ariyus, 2006)

2.4.1 *InvShiftRows*

Menurut Ariyus (2006) *InvShiftRows* adalah transformasi *byte* yang berkebalikan dengan transformasi *ShiftRows* pada tahapan enkripsi. Pada transformasi *InvShiftRows*, dilakukan pergeseran *bit* ke kanan sedangkan pada *ShiftRows* dilakukan pergeseran bit ke kiri. Ilustrasi transformasi *InvShiftRows* terdapat pada Gambar 2.11.



Gambar 2.11. Langkah *invshiftrows* (Ariyus, 2006)

2.4.2 *InvSubBytes*

InvSubBytes juga merupakan transformasi *bytes* yang berkebalikan dengan transformasi *SubBytes* pada tahapan enkripsi. Pada *InvSubBytes*, tiap elemen pada *state* dipetakan dengan menggunakan tabel *Inverse S-Box* yang berbeda dengan *SubBytes* (Ariyus, 2006). Tabel *Inverse S-Box* akan ditunjukkan dalam Gambar 2.12.

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Gambar 2.12. Langkah *invsubbytes* (Ariyus, 2006)

2.4.3 *InvMixColumns*

Menurut Ariyus (2006), Setiap kolom dalam state dikalikan dengan matrik perkalian dalam AES. Perkalian dalam matrik dapat dilihat pada Gambar 2.13.

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Gambar 2.13. Langkah *invmixcolumn* (Ariyus, 2006)

2.5 *Institutional Repository*

Menurut Pendit (2008), istilah *Institutional Repository* atau "Simpanan Kelembagaan" merujuk ke sebuah kegiatan menghimpun dan melestarikan koleksi digital yang merupakan hasil karya intelektual dari sebuah komunitas tertentu atau civitas akademik. Menurut Pendit (2008) adalah pengirim materi untuk disimpan bukanlah hanya si pembuat, tetapi juga pemilik karya (misalnya penerbit yang sudah membeli hak cipta dari penulis) dan pihak ketiga (misalnya pustakawan). Selain karya, disimpan pula meta data dari karya tersebut, dan ini dimungkinkan karena perangkat lunaknya memang sudah dilengkapi dengan borang untuk mengisi metadata secara mudah.

Pada umumnya tersedia mekanisme sederhana untuk meletakkan, mengambil ataupun mencari dokumen, karena mengendalikan inisiatif dari pihak pengirim, maka sebuah simpanan kelembagaan perlu mendapatkan kepercayaan dan dukungan. Karakteristik setiap simpanan kelembagaan tertentu saja sangat ditentukan oleh lembaga tempatnya berada, selain oleh jenis koleksinya, yang terutama merupakan basil penelitiannya. Menurut Hikmah (2014), ada empat hal yang menjadi perhatian utama bagi eksistensi *institutional repository* sebuah perguruan tinggi, diantaranya sebagai berikut:

1. Untuk mengumpulkan konten dalam satu lokasi sehingga mudah untuk ditemukan kembali.
2. Untuk menyimpan dan melestarikan aset intelektual sepanjang waktu.
3. Untuk menyediakan akses terbuka terhadap karya intelektual institusi kepada khalayak umum.
4. Untuk menciptakan visibilitas global bagi hasil karya ilmiah institusi.

Berdasarkan beberapa pendapat di atas dapat disimpulkan bahwa perpustakaan perguruan tinggi sebagai pendukung Tri Dharma Perguruan Tinggi, mempunyai tugas mengelola koleksi untuk memenuhi kebutuhan pemustaka. Salah satu jenis koleksi di perpustakaan perguruan tinggi adalah koleksi karya ilmiah mahasiswa seperti skripsi, tesis dan disertasi yang merupakan koleksi lokal baik dalam bentuk tercetak maupun digital. Koleksi lokal dalam bentuk digital dari suatu in-

stitusi yang dikelola dan disimpan dalam suatu tempat serta didistribusikan melalui jaringan komputer agar dapat digunakan lagi oleh masyarakat institusi, organisai, dosen, mahasiswa maupun masyarakat umum disebut sebagai *institutional repository*. Salah satu *software* untuk mengelola koleksi *repository* adalah *Eprints*. Dengan adanya *institutional repository* diharapkan koleksi *repository* dapat diakses dengan mudah dari mana saja dan kapan saja.

2.6 *Object Oriented Analysis Design* (OOAD)

Menurut Sholiq (2006) metode berorientasi objek atau *object oriented* merupakan paradigma baru dalam rekayasa perangkat lunak yang memandang sistem sebagai kumpulan objek-objek diskrit yang saling berinteraksi. Sedangkan yang dimaksud dengan berorientasi objek yaitu mengorganisasikan perangkat lunak sebagai kumpulan objek-objek diskrit yang bekerja sama antara informasi atau struktur data dan perilakau (*behavior*) yang mengaturnya.

Menurut B. Nugroho (2008), *Object Oriented Programming*(OOP) atau pemrograman berorientasi objek merupakan suatu cara baru dalam berpikir serta berlogika dalam menghadapi masalah-masalah yang akan dicoba ataupun di atasi dengan menggunakan bantuan komputer. Filosofi dari OOP yaitu menciptakan sinergi luar biasa sepanjang siklus pengembangan perangkat lunak (perencanaan, analisis, perancangan, serta implementasi) sehingga dapat diterapkan pada perancangan sistem secara umum menyangkut perangkat lunak, perangkat keras, serta sistem informasi secara keseluruhan.

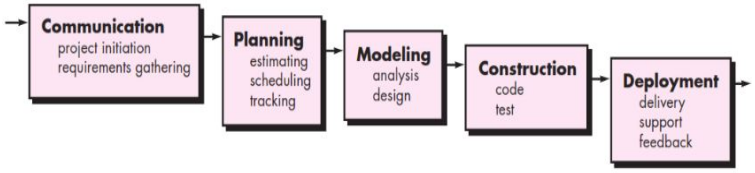
2.7 *Unifield Modelling Language* (UML)

Unified modeling language (UML) merupkan sebuah alat bantu yang digunakan para pengembang perangkat lunak dalam merancang sebuah sistem yang sesuai dengan keinginan dan kebutuhan (B. Nugroho, 2008). UML menyediakan beberapa diagram yang dapat menunjukkan berbagai aspek dalam sistem. Adapun diagram yang disediakan dalam UML antara lain:

1. Diagram *use case* (*use case* diagram)
2. Diagram aktivitas (*activity* diagram)
3. Diagram sekuensial (*sequence* diagram)
4. Diagram kolaborasi (*collaboration* diagram)
5. Diagram kelas (*class* diagram)
6. Diagram *statechart* (*statechart* diagram)
7. Diagram komponen (*component* diagram)
8. Diagram *deployment* (*deployment* diagram)

2.8 Metode *Waterfall*

Menurut Pressman (2012), model *waterfall* adalah model klasik yang bersifat sistematis, yang sifatnya berurutan dalam membangun *software*. Nama model ini sebenarnya adalah "*Linear Sequential Model*". Nama lain dari model ini sering disebut juga dengan "*classic life cycle*" atau metode *waterfall*. Model ini termasuk kedalam model *generic* pada rekayasa perangkat lunak dan *waterfall* pertama kali diperkenalkan oleh Winston Royce sekitar tahun 1970 sehingga sering dianggap kuno, tetapi merupakan model yang paling banyak dipakai dalam *Software Engineering* (SE). Model ini melakukan pendekatan secara sistematis dan berurutan. Disebut dengan *waterfall* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Langkah-langkah pengembangan system *waterfall* dapat dilihat pada Gambar 2.14.



Gambar 2.14. Struktur model *waterfall* (Pressman, 2012)

1. *Communication* (*Project Initiation* dan *Requirements Gathering*)
   Sebelum memulai pekerjaan yang bersifat teknis, sangan diperlukan adanya komunikasi dengan *costumer* untuk dapat lebih memahami dan mencapai tujuan yang ingin dicapai. Hasil dari komunikasi tersebut adalah inisialisasi proyek, seperti menganalisis permasalahan yang dihadapi dan mengumpulkan data-data yang diperlukan, serta membantu mendefenisikan fitur dan fungsi *software*.
2. *Planning (Estimating, Scheduling, Tracking)*
   Tahap berikutnya adalah tahapan perencanaan yang menjelaskan tentang estimasi tugas-tugas teknis yang akan dilakukan, resiko-resiko yang dapat terjadi sumber daya yang diperlukan dalam membuat sistem, produk kerja yang ingin dihasilkan, penjadwalan kerja yang adakan dilaksanakan, dan tracking proses pengerjaan sistem.
3. *Modelling (Analysis dan Design)*
   Tahapan ini adalah tahap perancangan dan permodelan arsitektur sistem yang berfokus pada perancangan struktur data, arsitektur *software*, tampilan *interface* dan algoritma program. Tujuannya adalah untuk lebih memahami gambaran besar dari apa yang akan dikerjakan.

4. *Construction (Code dan Test)*

   Tahapan *Construction* ini merupakan proses penerjemahan bentuk desain menjadi kode atau bentuk bahasa yang dapat dbaca oleh mesin. Setelah pengkodean selesai, tahapan selanjutnya dilakukan pengujian terhadap system dan juga kode yang sudah dibuat. Tujuannya adalah untuk menemukan kesalahan - kesalahan dalam penulisan kode yang mungkin terjadi untuk nantinya diperbaiki.

5. *Deployment (Delivery, Support, Feedback)*

   Tahapan *Deployment* merupakan tahapan implementasi *software ke customer*, pemeliharaan *software* secara berkala, perbaikan *software*, evaluasi *software*, dan pengembangan *software* berdasarkan umpan balik dari pengguna agar sistem dapat tetap berjalan dan berkembang sesuai dengan fungsinya.

Menurut Pressman (2012) metode *Waterfall* memiliki beberapa kelebihan dan juga kekurangan diantaranya, yaitu:

1. Kelebihan Metode *Waterfall* Adapun urutan proses pengerjaan sistem menggunakan metode ini menjadi lebih teratur dari satu tahap ke tahap yang selanjutnya. Dari sisi *user* juga lebih menguntungkan karena dapat merencanakan dan menyiapkan seluruh kebutuhan data dan proses yang akan diperlukan secara maksimal. Jadwa setiap proses dapat ditentukan secara pasti. Sehingga dapat dilihat jelas target penyelesaian pengembagan program. Dan dengan adanya urutan yang pasti, dapat dilihat pula progress untuk setiap tahapan pengembangan.
2. Kekurangan Metode *Waterfall* Adapun kekurangan metode *waterfall* sifatnya kaku, sehingga sulit dalam melakukan perubahan di tengah proses. Jika terdapat kekurangan proses atau prosedur pada tahapan sebelumnya, maka tahapan pengembangan harus dilakukan mulai dari awal. Hal ini menyebabkan dapat memakan waktu yang cukup lama. Selain itu membutuhkan daftar kebutuhan system yang lengkap di awal pengembangan. Untuk menghindari pengulangan tahap dari awal, *user* harus memberikan seluruh prosedur, data dan laporan yang di inginkan di mulai dari tahap awal pengembangan (Pressman, 2012).

2.9 *Hypertext Preprocessor* (PHP)

Menurut Peranginangin (2006). PHP merupakan singkatan dari PHP: *Hypertext Preprocessor* yang digunakan sebagai bahasa *script server side* yang digunakan untuk pengembangan web yang disisipkan dalam dokumen HTML. Peng-

gunaan PHP memungkinkan *website* dapat dinamis sehingga perawatan situs web tersebut menjadi lebih mudah dan efisien. PHP merupakan *software Open-source* yang telah disebarkan dan dilisensikan secara gratis dan juga dapat di *download* secara bebas dari situs resminya.

Kelenihan lain dari PHP adalah bisa digunakan untuk mengakses berbagai macam *database* diantaranya *Access, Oracel, MySQL*, dan lain-lain. PHP masih banyak dipakai untuk memrogram situs *Web* yang dinamis, walaupun tidak tertutup kemungkinan digunakan untuk pemakaian lain. Menurut Peranginangin (2006) adapun kelebihan - kelebhan PHP dari bahasa pemrograman lain adalah:

1. Bahasa pemrograman php adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaanya.
2. *Web Server* yang mendukung php dapat ditemukan dimana–mana dari mulai IIS sampai dengan *apache*, dengan configurasi yang juga relative mudah.
3. Dalam sisi pengembangan juga lebih mudah, karena banyaknya milis–milis dan *developer* yang siap membantu dalam pengembangan.
4. Dalam sisi pemahamanan, PHP adalah bahasa *scripting* yang paling mudah karena mendapatkan referensi yang banyak.
5. PHP adalah bahasa *open source* yang dapat digunakan diberbagai mesin (*linux, unix, windows*) dan juga dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

2.10 MySQL

Menurut Sunarfrihantono (2002), MySQL merupakan *multiuser database* yang menggunakan bahasa *Structured Query Language* (SQL). MySQL dalam operasi *client server* melibatkan *server daemon* MySQL disisi server dan berbagai macam program serta *library* yang berjalan disisi client. MySQL dapat menangani data yang cukup besar. Menurut perusahaan yang mengembangkan MySQL yaitu TEX, MysQL mampu menyimpan data lebih dari 40 database, 10.000 tabel, dan sekitar 7.000.000 baris totalnya kurang lebih 100 *Giga byte* data.

Sebagai sebuah program penghasil database, MySQL tidak dapat berjalan sendiri tanpa adanya sebuah aplikasi pengguna (*interface*) yang berguna sebagai program aplikasi pengakses database yang dihasilkan. MySQL dapat didukung oleh hampir semua program aplikasi baik yang *Open Source* seperti *PHP, framework php, java, phyton* maupun yang tidak *Open Source*yang ada pada *platform windows* seperti *Visual Basic, Delphi* dan lainnya (A. Nugroho, 2002).

2.11 *Blackbox Testing*

Blackbox testing merupakan cara yang dapat digunakan untuk mepresentasikan sistem yang cara kerja didalamnya tidak tersedia untuk diinspeksi. Teknik pengujian *blackbox* juga digunakan untuk pengujian berbasis skenario, dimana isi dalam sistem mungkin tidak tersedia untuk diinspeksi tapi masukan dan keluaran yang didefinisikan oleh *use case* dan informasi analisis yang lain (Hariyanto, 2004). *Blackbox testing* berusaha untuk menemukan kesalahan dalam kategori - kategori sebagai berikut:

1. Fungsi yang tidak benar atau fungsi yang hilang.
2. Kesalahan antarmuka.
3. Kesalahan dalam struktur data atau akses *database eksternal*.
4. Kesalahan perilaku (behavior) atau kesalahan kinerja.
5. Berikut ini merupakan rumus perhitungan dari hasil pengujian dengan menggunakan metode *black box testing* dapat dilihat pada Persamaan 2.1.

$$Persentasi\_keberhasilan = \frac{jawaban\_berhasil}{Jumlah\_prtanyaan} X100\% \tag{2.1}$$

2.12 *User Acceptance Test* (UAT)

User Acceptance Test (UAT) merupakan pengujian yang dibentuk dengan metode sederhana untuk memutuskan apakah program yang telah di buat sudah layak untuk digunakan. Program yang tidak lolos semua pengujian ini menujukan program tersebut tidak memenuhi semua spesifikasi (Hariyanto, 2004). Berikut ini merupakan rumus perhitungan dari hasil pengujian denganmenggunakan metode *User Acceptance Testing* dapat dilihat pada Persamaan 2.2.

$$\widehat{PersentaseResponden} = \frac{JumlahJawabannxbobotn + n}{JumlahJawabannxbobot} X100\% \tag{2.2}$$

Adapun perhitungan persentase pertanyaan dapat dilihat pada Persamaan 2.3.

$$\widehat{PersentasePertanyaan} = \frac{JumlahJawabannxbobotn + n}{JumlahJawabannxbobot} X100\% \tag{2.3}$$

2.13 Perpustakaan Universitas Islam Negeri Sultan Syarif

kasim Riau Perpustakaan IAIN Susqa berdiri bersamaan dengan berdirinya Institut Agama Islam Negeri Sulthan Syarif Qasim berdasarkan SK Menteri Agama RI No. 194 tahun 1970 tanggal 9 September 1970 yang terdiri dari 3 fakultas, yaitu Fakultas Tarbiyah di Pekanbaru, Fakultas Syariah di Tembilahan dan Fakultas

Ushuluddin di Pekanbaru.

Pada tahun 1970 itu juga masyarakat dan pemuka agama di Riau meminta kepada gubernur Riau Kolonel Arifin Ahmad untuk mendirikan kampus di Jl. Pelajar (sekarang Jl. Ahmad Dahlan) seluas tanah 3,5 hektar di kecamatan Sukajadi Pekanbaru. Pada tahun 1973 pembangunan ruang kuliah selesai dibangun. Dengan satu lokal diantaranya dipakai sebagai perpustakaan. Tahun 1978 proyek Pusat Departemen Agama RI membangun gedung perpustakaan tersendiri dengan luas 8 X 25 meter, dengan koleksi buku sebanyak 12.897 exp. Yang terdiri dari 1.744 judul buku.

Pada tahun 1985 ruang Perpustakaan diperluas menjadi 450 m^2 dengan koleksi buku 30307 exp, dengan 2.920 judul. Pada awalnya Perpustakaan IAIN SUSQA menggunakan sistem tertutup, dimana pemustaka tidak dapat mengambil buku yang diinginkan secara langsung ke rak, tapi harus melalui petugas dengan terlebih dahulu melakukan penelusuran melalui katalog. Melihat animo pengguna / pemustaka cukup banyak, maka pada tahun 1999 setalah dilakukan perluasan, digunakanlah sistem terbuka, dimana pemustaka dapat langsung melakukan browsing dan mengambil buku yang diinginkan.

Sedangkan untuk melayani kebutuhan informasi pemustaka yang berada di Kampus II Panam, pada tahun 2000 dibangun sebuah perpustakaan cabang seluas 7 X 16 mtr. Pada tahun 2005, seiring dengan perubahan status IAIN menjadi Universitas Islam Negeri Sulthan Syarif Kasim Riau (UIN SUSKA Riau) yang berimbas kepada perpustakaan. Perpustakaan dituntut untuk lebih optimal dalam melakukan pelayanan berorientasi kepada otomasi perpustakaan yang selama ini masih bersifat konvensional. Pada tahun 2006 Departemen Agama RI melakukan program otomasi perpustakaan di 10 Perguruan Tinggi Islam, UIN SUSKA Riau termasuk ke-10 PTI tersebut dan mendapatkan *software* SIMPus (Sistem Informasi Manajemen Perpustakaan).

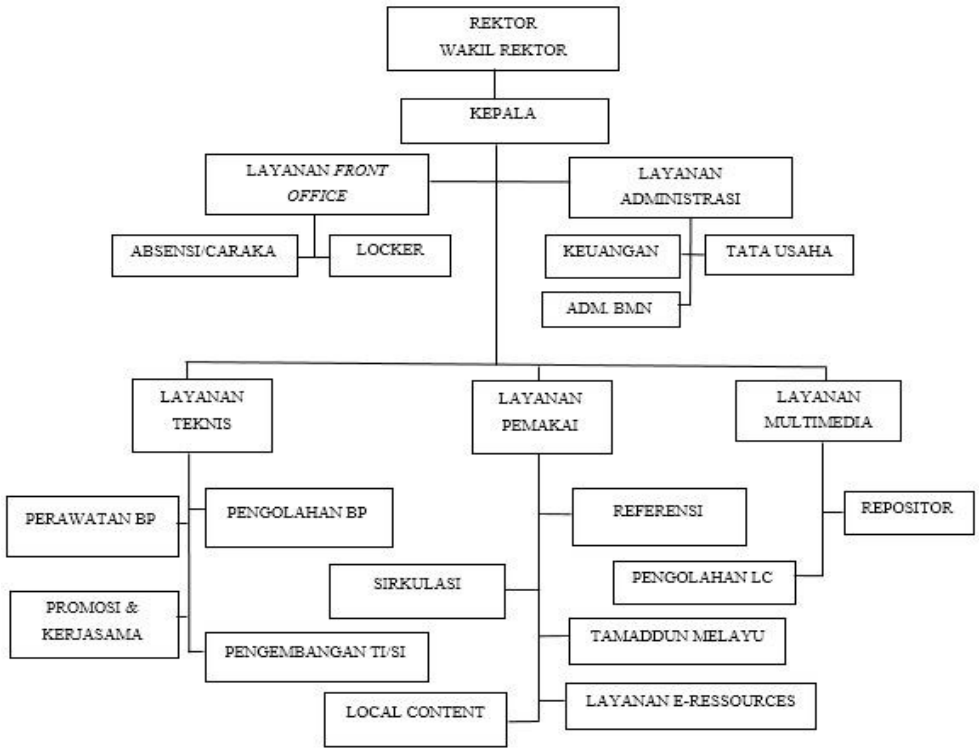
Dengan kerja keras dari seluruh staff perpustakaan selama 7 bulan mulai dari pengenalan program, training, persiapan input data seluruh koleksi yang ada di perpustakaan UIN SUSKA Riau sampai dengan pemustakaan otomasi secara menyeluruh termasuk pergantian kartu anggota, Alhamdulillah berkat dukungan pimpinan Universitas dalam memenuhi seluruh sarana yang diperlukan dalam menjalankan otomasi, maka pada bulan September 2006 seiring dengan dimulainya tahun ajaran baru 2006/2007 sistem otomasi perpustakaan baik mulai dari pengolahan, pendataan, transaksi peminjaman dan pengembalian, semua yang ada pada program SIMPus tersebut mulai diterapkan.

Tahun ajaran 2008/2009 Perpustakaan menempati gedung baru 4 lantai

dengan luas 4000 m2 di Kampus Ali Haji Panam Pekanbaru. Ketersediaan sarana dan fasilitas di perpustakaan tersebut menuntut dilakukannya inovasi teknologi informasi dalam hal otomatisasi perpustakaan. Untuk itu maka pada tahun 2009 sistem informasi perpustakaan yang sebelumnya berbasis *Desktop* dengan menggunakan software SIMPus, beralih ke *software OpenBiblio* yang sudah berbasi *web*. Perpustakaan UIN SUSKA Riau mulai awal berdiri sampai sekarang telah dipimpin oleh 8 orang kepala yaitu:

1. Drs. Basyiran S. Alam Th. 1973 - 1976
2. Drs. Zul Asyri, LA Th. 1976 - 1977
3. Drs. Noor Aini, HA Th. 1977 - 1978
4. Drs. Darwis Tanjung Th. 1979 - 1994
5. Drs. Mahyuni Said Th. 1994 - 2005
6. Dra. Hj. Azwinar Aziz (Plt. kepala) Th. 2005 - 2006
7. Drs. Suhaimi D, M.Si Th. 2006 - 2014
8. Dr. Suriani, S.Ag, SS, M.Si Th. 2014 – Sekarang.

(Sumber: http://lib.uin-suska.ac.id/, diakses pada 12 Maret 2018). Dibawah ini merupakan struktur organisasi perpustakaan UIN SUSKA Riau dapat dilihat pada Gambar 2.15.



Gambar 2.15. Struktur organisasi pada perpustakaan UIN SUSKA riau

2.14 Studi Pustaka Penelitian Terdahulu

Berikut ini merupakan penelitian – penelitian yang telah dilakukan oleh peneliti terdahulu yang terkait dengan penelitian yang akan dibuat dapat dilihat pada Tabel 2.2.

Tabel 2.2. Penelitian terdahulu

No	Simbol	Deskripsi	Hasil
1	Sianturi (2013)	Perancangan Aplikasi Pengamanan Data Dengan Kriptografi *Advanced Standard* (AES)	Aplikasi ini dibuat untuk menjamin keamanan dan keutuhan dari suatu data yang dimanana enkripsi dilakukan saat data akan dikirim. Kemudian proses dekripsi dilakukan oleh penerima data. Data rahasia yang diterima akan diubah kembali menjadi data asal atau plainteks.
2	Latif (2015)	Implementasi Kriptografi Menggunakan Metode *Advance Encryption Standard* (AES) Untuk Pengamanan Data Teks	Dengan aplikasi yang dibuat pesan dapat di enkripsi dan dekripsi dengan baik, hal ini terbukti dari pesan yang dihasilkan tidak dapat di baca oleh pengguna, kemudian setelah pesan tersebut di dekripsi, maka akan kembali seperti pesan asli dan dapat di baca pesan tersebut.
3	Gumira, Erlansharsi, dkk. (2016)	Implementasi Metode *Advance Encryption standard* (AES) dan *Message Digest* 5 (MD5)Pada enkripsi dokumen (Studi Kasus LPSE UNIB)	Hasil penelitian ini menunjukkan bahwa algoritma AES 256 *bit* telah berhasil mengamankan file. Sedangkan algoritma MD5 digunakan sebagai penghasil nilai hash yang akan digunakan sebagai sandi untuk mengenkripsi dan mendekripsikan kembali dokumen.
4	Munawir, Zulfan, Yanti, dan Mudianto (2017)	Teknik Pengamanan File Dokumen Berbasis Text Menggunakan Metode AES	Tujuan dari penelitian ini adalah untuk merancang Aplikasi Kriptografi AES untuk Keamanan File Dokumen. Setelah melakukan penelitian, aplikasi kriptografi ini berhasil dilakukan dan diimplementasikan untuk mengamankan file dokumen.