

## BAB II

### KAJIAN LITERATUR

#### 2.1 Tinjauan Pustaka

##### 2.1.1 The State of The Art

Beberapa penelitian telah dilakukan sebelumnya untuk mengamankan pesan rahasia dengan menggunakan teknik kriptografi. Dalam upaya untuk mengembangkan sistem yang serupa dengan hal tersebut maka diperlukan studi literatur sebagai tahapan pada metode penelitian yang akan dilakukan. Berikut merupakan penelitian yang telah dilakukan sebelumnya dan memiliki kecocokan dengan topik yang peneliti bahas, antara lain:

- a. **Muhammad Faqih, dkk.** Penelitian yang dilakukan dengan judul Aplikasi Kompresi Untuk Pengiriman Data Menggunakan Metode LZW (*Lempel Ziv Welch*). Hasil penelitian ini adalah aplikasi kompresi data ini bisa mengkompresi data dengan baik, dengan hasil kompresi yang sesuai algoritma LZW dimana data yang lebih besar hasil kompresinya bisa lebih kecil dengan isi data karakter lebih banyak yang sama begitu sebaliknya. [7].
- b. **Heri Haryanto, dkk.** Penelitian yang dilakukan dengan judul Implementasi Kombinasi Algoritma Enkripsi Aes 128 Dan Algoritma Kompresi Shannon-Fano. Hasil penelitian ini adalah algoritma Shannon-Fano tidak cocok digunakan untuk file teks berformat docx dan pdf karena format file tersebut lebih kompleks dibandingkan dengan file dengan format txt [9].
- c. **Jonathan Marcel.** Penelitian yang dilakukan dengan judul Studi Perbandingan Cipher Blok Algoritma Blowfish dan Algoritma Camellia menganalisis tentang kelemahan yang ada pada blok data dan putaran pada algoritma blowfish. Hasil penelitian ini adalah panjang algoritma *Blowfish* terlalu pendek dan tidak aman karena akan melemahkan informasi plainteks ketika enkripsi lebih dari  $2^{32}$  blok data dan juga mudah diserang dengan *birthday attack* [10].
- d. **Ako Muhammad Abdullah.** Penelitian yang dilakukan dengan judul *Advanced Encryption Standard (AES) Algorithm to Encrypt and Decrypt*

Data bertujuan untuk memberikan gambaran dari algoritma AES dan menjelaskan beberapa fitur penting dari algoritma ini secara rinci serta membandingkan algoritma tersebut dengan algoritma lain seperti DES, 3DES dll. Hasil penelitian ini adalah AES memiliki kinerja atau kemampuan untuk memberikan keamanan yang lebih banyak dibandingkan dengan algoritma lain seperti DES, 3DES dll. [8].

- e. **Resianta Perangin-angin, dkk.** Penelitian yang dilakukan dengan judul Analisis Alokasi Memori dan Kecepatan Kriptografi Simetris dalam Enkripsi dan Deskripsi bertujuan untuk memberi informasi tentang perbandingan rasio kecepatan dan alokasi memori yang digunakan antara algoritma AES (*Advanced Encryption Standard*) dan algoritma Blowfish. Hasil penelitian ini adalah rasio kecepatan dan alokasi memory menggunakan AES (*Advanced Encryption Standard*) lebih membutuhkan ruang penyimpanan yang lebih besar bila dibandingkan dengan algoritma blowfish. [11].

Berbagai penelitian sebelumnya yang telah disebutkan di atas dirangkum pula pada Tabel 2.1 State of the art.

Tabel 2.1 State of The Art

No	Peneliti	Tahun	Metode	Tujuan
1.	Muhammad Faqih, Imam Marzuki, dan Dyah Ariyanti	2017	<i>Lempel Ziv Welch</i> (LZW)	Bertujuan untuk kompresi berkas yang akan dikirim menggunakan algoritma LZW ( <i>Lempel Ziv Welch</i> )
2.	Heri Haryanto, Romi Wiryadinata, dan Muhammad Afif	2018	Algoritma AES 128 dan Algoritma Shannon-Fano	Bertujuan untuk melakukan pengujian kombinasi enkripsi-kompresi dengan menggunakan gabungan Algoritma enkripsi AES dan algoritma kompresi Shannon-Fano
3.	Jonathan Marcel T	2017	Cipher Blok Algoritma Blowfish dan Algoritma Camellia	Bertujuan untuk menganalisis tentang kelemahan yang ada pada blok data dan putaran pada algoritma blowfish
4.	Ako Muhammad Abdullah	2017	Advanced Encryption Standard (AES)	Bertujuan untuk membandingkan algoritma tersebut dengan algoritma lain seperti DES, 3DES dll

Tabel 2.2 State of The Art (lanjutan)

No	Peneliti	Tahun	Metode	Tujuan
5.	Resianta P, Indra Kelana J, Benget Rumahorbo, Berlian Juni R. Marpaung	2019	<i>Advanced Encryption Standard</i> (AES) dan algoritma Blowfish	Bertujuan untuk membandingkan rasio kecepatan dan alokasi memori yang digunakan antara algoritma AES dan algoritma Blowfish
6.	Pirman Abdurohman	2022	AES ( <i>Advanced Encryption Standard</i> ) dan LZW ( <i>Lempel Ziv Welch</i> )	Bertujuan untuk mengamankan data berupa teks dan file menggunakan algoritma AES untuk enkripsi, dekripsi dan algoritma LZW untuk kompresi dan dekompresi

Dari semua penelitian yang telah dilakukan diatas, dapat disimpulkan bahwa perlunya proses pengamanan pada data. Sebagai pembeda dari penelitian sebelumnya, penelitian ini menggunakan algoritma AES dan LZW, dimana kedua algoritma ini di kombinasikan sehingga mendapatkan hasil yang lebih baik. Algoritma AES digunakan untuk enkripsi dan dekripsi data, sedangkan algoritma LZW digunakan untuk kompresi data sebelum di unggah ke *cloud storage*.

Algoritma AES yang digunakan dalam penelitian ini adalah AES-128, dimana proses enkripsi dan dekripsi menjadi lebih cepat karena jumlah round yang terjadi lebih sedikit dibandingkan dengan panjang kunci AES yang lainnya.

Panjang kunci algoritma AES yang lainnya yaitu AES-192 dengan jumlah round sebanyak 12 dan AES-256 dengan jumlah round sebanyak 14. Sehingga dapat disimpulkan bahwa semakin besar kunci yang dipakai semakin besar juga round yang dikerjakan, hal tersebut berakibat pada lamanya proses enkripsi dan dekripsi.

Pada penelitian ini juga algoritma LZW digunakan untuk kompresi file hasil enkripsi sehingga ukuran file tidak begitu besar ketika di *download*.

## 2.2 Landasan Teori

### 2.2.1 Pengertian Kriptografi

Menurut pengertian etimologi bahwa Menurut pengertian etimologi bahwa kata kriptografi (*Cryptography*) berasal dari bahasa Yunani, yaitu “*cryptos*” yang maknanya rahasia atau tersembunyi dan “*graphein*” yang maknanya tulisan, jadi *Cryptography* bisa diartikan sebagai tulisan rahasia. Asalnya kriptografi dimengerti sebagai suatu ilmu atau disiplin tentang cara menyembunyikan pesan, tetapi seiring kemajuan dan berkembangnya zaman sampai saat ini, pengertian kriptografi dikembangkan menjadi suatu ilmu tentang teknik matematis dalam menyelesaikan persoalan di bidang keamanan berupa otentikasi dan privasi [12].

Kriptografi Merupakan ilmu yang mempelajari tentang cara atau teknik menjaga data atau pesan supaya tetap aman saat dikirimkan dari pengirim ke penerima pesan tanpa adanya gangguan dari pihak ketiga. Perlu diketahui bahwa dalam kriptografi kita akan sering mendapati berbagai istilah. Berikut berbagai istilah yang ada dalam kriptografi yang harus diketahui diantaranya :

a. *Message, plaintext*, dan *ciphertext*

*Message* atau pesan yakni suatu data atau informasi yang bisa dibaca, dipahami dan dimengerti maknanya atau maksudnya. Pesan juga bisa disebut *plaintext* atau teks asli atau bisa juga disebut teks jelas (*cleartext*).

b. *Sender* (Pengirim) dan *Receiver* (Penerima)

Dalam mekanisme komunikasi data pasti akan melibatkan pertukaran pesan antara dua entitas yaitu pengirim dan penerima. Pengirim (*sender*) merupakan entitas atau pihak yang mengirimkan pesan kepada entitas lainnya. Adapun penerima (*receiver*) merupakan entitas atau pihak yang menerima pesan yang dikirim oleh *sender*.

c. Enkripsi (*encryption*) dan dekripsi (*decryption*)

Mekanisme penyandian pesan asli (*plaintext*) yang diubah menjadi *ciphertext* disebut enkripsi (*encryption*) (nama standar menurut ISO 7498-2). Adapun mekanisme untuk mengembalikan *ciphertext* menjadi *plaintext* atau pesan asli seperti semula disebut dekripsi (*decryption*) (nama standar menurut ISO 7498- 2).

d. *Key* dan *Cipher*

Metode kriptografi atau bisa disebut juga dengan *cipher*, yakni suatu aturan dalam proses enkripsi dan dekripsi pesan, atau suatu fungsi

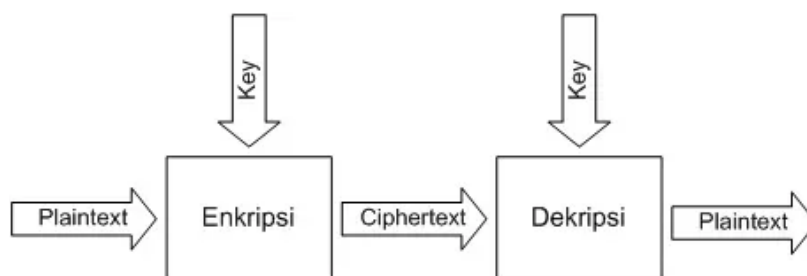
matematika yang dipakai dalam proses enkripsi dan dekripsi pesan. Beberapa *cipher* membutuhkan algoritma atau metode yang berbeda dalam proses enkripsi dan dekripsi. Konsep matematis yang didasari atas metode kriptografi merupakan hubungan antara dua buah himpunan yang berisikan elemen-elemen *plaintext* dan himpunan yang berisikan *ciphertext*. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antara dua himpunan tersebut. Diumpamakan  $Pl$  dinyatakan sebagai plainteks dan  $Cp$  dinyatakan sebagai cipherteks, maka :

$En(Pl) = C \rightarrow En$  sebagai fungsi enkripsi untuk memetakan  $Pl$  ke  $Cp$

$De(Cp) = Pl \rightarrow De$  sebagai fungsi dekripsi untuk memetakan  $Cp$  ke  $Pl$

Dikarenakan proses enkripsi dilanjutkan dengan proses dekripsi pesan (*ciphertext*) ke pesan asli (*plaintext*) , maka persamaan  $De(En(Pl)) = Pl$  harus benar atau hasilnya harus sama seperti plainteks awal [13].

Kriptografi dapat digunakan untuk mengatasi masalah keamanan data dengan memakai kunci (key) tertentu, hal ini menjadikan algoritma yang digunakan tidak bersifat rahasia lagi, akan tetapi kunci (key) kerahasiaannya harus tetap dijaga. Kunci (key) merupakan parameter yang dipakai untuk melakukan transformasi enkripsi dan dekripsi pada data. Biasanya kunci yang dipakai berupa deretan bilangan atau string. Pada dasarnya kriptografi memiliki dua proses utama yaitu enkripsi dan dekripsi. Sebagaimana yang telah diterangkan diatas proses enkripsi yakni proses pengubahan plainteks menjadi cipherteks dengan kunci (key) tertentu, sehingga informasi asli atau isi pada pesan tersebut sulit dibaca dan dipahami. Dengan menggunakan key ( $K$ ), maka fungsi enkripsi dan dekripsi dapat ditulis dalam skema yang diperlihatkan pada gambar 2.1 berikut.



Gambar 2.1 Skema proses Kriptografi [14]

Disamping algoritma yang digunakan, kunci menjadi peranan yang sangat penting dalam proses enkripsi dan dekripsi untuk membongkar kerahasiaan dari pesan (*plaintext*) yang telah dienkripsi, sehingga isi informasi dari pesan dapat diketahui.

### **2.2.2 Pengertian Kompresi**

Kompresi data adalah ilmu atau seni merepresentasikan informasi dalam bentuk yang lebih compact. David Salomon mengatakan bahwa data kompresi adalah proses mengkonversikan sebuah input data stream (stream sumber, atau 20 data mentah asli) menjadi data stream lainnya (bit stream hasil, atau stream yang telah terkompresi) yang berukuran lebih kecil. Pemampatan merupakan salah satu dari bidang teori informasi yang bertujuan untuk menghilangkan redundansi dari sumber. Pemampatan bermanfaat dalam membantu mengurangi konsumsi sumber daya yang mahal, seperti ruang hard disk atau perpindahan data melalui internet [15].

Tujuan dari kompresi data adalah untuk merepresentasikan suatu data digital dengan sesedikit mungkin bit, tetapi tetap mempertahankan kebutuhan minimum untuk membentuk kembali data aslinya. Data digital ini dapat berupa text, gambar, suara, dan kombinasi dari ketiganya, seperti video. Untuk membuat suatu data menjadi lebih kecil ukurannya daripada data asli, diperlukan tahapan-tahapan (algoritma) untuk mengolah data tersebut. Dalam algoritma kompresi data, tidak ada algoritma yang cocok untuk semua jenis data. Hal ini disebabkan karena data yang akan dimampatkan harus dianalisis terlebih dahulu, dan berharap menemukan pola tertentu yang dapat digunakan untuk memperoleh data dalam ukuran yang lebih kecil. Karena itu, muncul banyak algoritma-algoritma kompresi data.

### **2.2.3 Advanced Encryption Standards (AES)**

*Advanced Encryption Standard* atau disingkat AES pertama kali diumumkan oleh *National Institute Of Standard and Technology* (NIST) pada



tahun 2001 sebagai pengganti algoritma DES yang semakin mudah untuk diretas. AES merupakan hasil dari kompetisi yang disediakan oleh NIST tahun 1997. Pada tahap pertama berdasarkan penilaian tingkat keamanan, algoritma, harga dan karakteristik implementasi telah lolos 15 peserta dari 21 peserta ke tahap berikutnya. Akan tetapi karena dianggap kurang aman dan efektif sepuluh dari 15 peserta tersebut dinyatakan gugur pada tahap berikutnya.

Pada tahap seleksi akhir di bulan agustus 1999 telah dipilih lima kandidat, yaitu MARS (IBM, Amerika Serikat), *Rijndael* (Belgia), RSA (RSA corp., Amerika Serikat), *Twofish* (Counterpane., Amerika Serikat) dan *Serpent* (Israel, Norwegia, Inggris). Pada tahap ini NIST melakukan penilaian berdasarkan pada implementasi *software* dan *hardware*, implementasi atas serangan, *general security*, enkripsi dan dekripsi, kemampuan kunci, ruang lingkup, dan kinerja lain seperti fleksibilitas dan potensial untuk tingkat instruksi paralel. Hasil akhir dari kompetisi tersebut, Algoritma *Rijndael* yang dibuat oleh Dr. Vincent Rijmen dan Dr. Joan Daemen terpilih sebagai pemenang pada tahun 2 Oktober 2000 [16].

AES termasuk algoritma simetris yang dapat memakai kunci yang sama untuk enkripsi dan dekripsi pesan. AES mempunyai tiga pilihan tipe kunci yakni AES-128 bit, AES-192 bit dan AES-256 bit. Pada masing-masing tipe digunakan kunci internal yang berbeda yaitu *round key* dalam setiap proses putaran. Untuk memperjelas hubungan *key* dan *RoundKey* akan diperlihatkan dalam sebuah tabel 2.2 berikut ini.

Tabel 2.3 Hubungan Panjang key AES dan jumlah round key

Panjang Kunci AES (bit)	Jumlah <i>Round Key</i> (Nr)
128	10
192	12
256	14

#### 2.2.4 Struktur Proses Enkripsi AES 128 bit

Pada dasarnya AES merupakan proses transformasi terhadap *state*. Pada teks asli (*plaintext*) dalam blok (128 bit) terlebih dahulu didefinisikan sebagai *states*. Penyandian AES merupakan proses transformasi pada *state* secara

berulang dalam beberapa ronde atau putaran. *States* yang menjadi keluaran ronde  $k$  dijadikan masukan untuk ronde  $k+1$ .

Tahap awal teks asli (plainteks) direorganisasi menjadi sebuah *state*, kemudian sebelum dilakukan ronde 1 dimulai blok teks asli dicampur dengan kunci ronde ke-0 (transformasi ini disebut *AddRoundKey*).

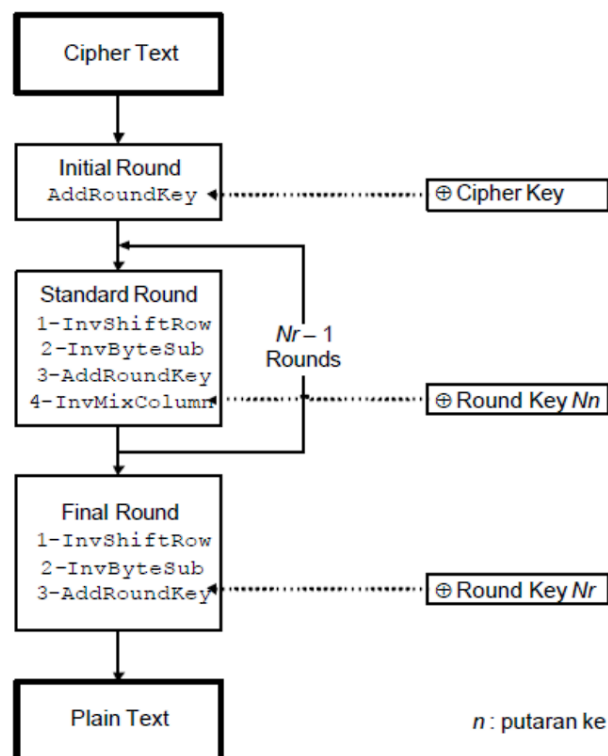
Setelah itu, ronde ke-1 sampai ronde ke- $(N_r-1)$  dengan  $N_r$  adalah jumlah ronde menggunakan 4 jenis transformasi, diantaranya yaitu *SubBytes*, *ShiftRows*, *MixColumns* dan *AddRoundKey*. Dan pada ronde terakhir, yakni ronde ke- $N_r$  dilakukan proses transformasi serupa dengan ronde lainnya, namun tanpa melakukan proses transformasi *MixColumns*.

#### **2.2.5 Struktur Proses Dekripsi AES 128 bit**

Perlu diketahui bahwa secara ringkas dekripsi AES yakni kebalikan dari proses enkripsi AES. Pada dasarnya dekripsi AES menggunakan proses transformasi *invers* pada semua transformasi dasar yang telah digunakan untuk melakukan enkripsi AES.

Untuk setiap proses transformasi dasar AES mempunyai transformasi *invers*-nya masing-masing, diantaranya: *InvSubBytes*, *InvShiftRows*, dan *InvMixColumns*. Adapun untuk *AddRoundKey* termasuk transformasi bersifat *self-invers* dengan syarat kunci yang digunakan harus sama. Untuk skema dekripsi AES diperlihatkan seperti gambar 2.2 berikut.





Gambar 2.2 Skema Dekripsi AES [14]

## 2.2.6 Transformasi – transformasi algoritma AES

Pada dasarnya algoritma penyandian AES memakai 4 proses transformasi, diantaranya substitusi atau disebut juga *SubBytes*, permutasi (*ShiftRows*), pencampuran (*MixColoumns*) dan penambahan (*AddRoundKey*). Berikut merupakan penjelasan dari masing-masing transformasi enkripsi pada algoritma AES :

### A. *AddRoundKey*

*AddRoundKey()* yakni transformasi yang dilakukan pada kunci utama. Sedangkan pada 10 *round* lainnya, proses transformasi *AddRoundKey* dilakukan terhadap kunci putaran (*round key*). Proses *AddRoundKey* merupakan operasi XOR antara *state* awal (*plaintext*) dengan *cipher key*, tahap ini juga bisa disebut dengan *initial round*. Setiap *byte* dalam *array* dilakukan operasi XOR, sehingga dapat menghasilkan nilai baru pada *array* hasil, dengan ukuran *array* hasil sama dengan ukuran *array state* awal dan *array*

key, yaitu sebesar 4x4. Pada *array state* hasil, untuk hasil dari masing-masing baris dan kolomnya merupakan hasil yang diperoleh operasi XOR antara *array state* awal dengan *array key* untuk baris dan kolom yang sama.

### B. SubBytes

*SubBytes()* yakni transformasi yang berfungsi untuk melakukan pemetaan pada setiap bytes dari *array state* menggunakan bantuan tabel substitusi S-Box. Adapun untuk substitusi S-Box diperlihatkan pada gambar 2.3 berikut.

		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 2.3 Tabel substitusi S-Box [14]

Cara untuk melakukan proses substitusi dengan menggunakan tabel S-Box yaitu:

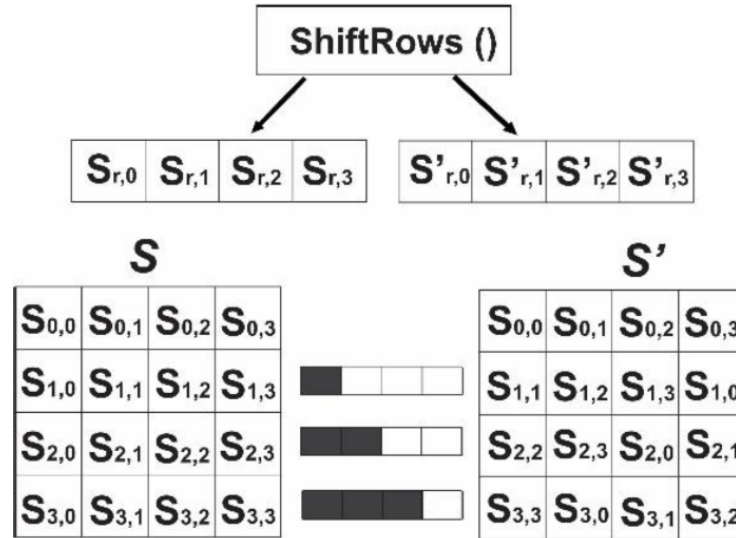
Dimisalkan  $S[r,c] = xy$  untuk setiap *byte* pada *array*, dalam hal ini *xy* merupakan angka (digit) heksadesimal dari nilai  $S[r,c]$ , maka untuk hasil nilai dari substitusinya, dinyatakan sebagai  $S'[r,c]$ , merupakan elemen di dalam S-Box yang berupa perpotongan baris *x* dengan kolom *y*.

**Contoh :  $S[r,c] = 17$  maka  $S'[0,0] = F0$**

### C. ShiftRows

*ShiftRows()* merupakan transformasi untuk melakukan proses pergeseran secara *wrapping* pada 3 baris terakhir dari *array state*. Untuk jumlah pergeseran tergantung pada nilai baris (*r*). Jika baris  $r = 1$  maka digeser sejauh 1 *byte*, dan jika baris  $r = 2$  maka digeser sejauh 2 *byte*, dan jika baris  $r = 3$

maka digeser sejauh 3 *byte*, serta jika baris  $r = 0$  maka tidak digeser. Skema Gambar 2.4 memperlihatkan dengan jelas proses *ShiftRows()*.



Gambar 2.4 Proses ShiftRows [14]

#### D. MixColumns

Setelah proses transformasi *ShiftRows* selesai dilakukan, maka selanjutnya dilakukan transformasi *MixColumns()*, yang merupakan sumber utama dari difusi (percampuran) pada algoritma AES. Difusi merupakan prinsip yang menyebarkan pengaruh satu bit *plaintext* atau kunci ke sebanyak mungkin *ciphertext*. Transformasi *MixColumns()* berfungsi untuk mengalikan setiap kolom dari *array state* dengan *polinom*  $a(x) \bmod (x^4 + 1)$ . Setiap kolom diperlakukan sebagai *polinom* 4 suku pada GF (28). *Polinom*  $a(x)$  yang ditetapkan pada persamaan 1.

Rumus 2.1 Persamaan 1

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$$

Transformasi Ini dinyatakan sebagai perkalian matriks seperti pada persamaan 2.

Rumus 2.2 Persamaan 2

$$\begin{bmatrix} s'_{0,c} & s'_{1,c} & s'_{2,c} & s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 & 01 & 02 & 03 & 01 & 01 & 01 & 02 & 03 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

$$s'_{3,c} = 03 \ 01 \ 01 \ 02 \ s_{3,c}$$

$$s'(x) = a(x) \otimes s(x)$$

Hasil dari perkalian matriks tersebut adalah setiap byte dalam kolom array state akan digantikan dengan nilai baru. Persamaan matematis untuk setiap byte tersebut pada persamaan 3.

Rumus 2.3 Persamaan 3

$$s'_{0,c} = (\{02\} \cdot s_{0,c}) \otimes (\{03\} \cdot s_{1,c}) \otimes s_{2,c} \otimes s_{3,c}$$

$$s'_{1,c} = s_{0,c} \otimes (\{02\} \cdot s_{1,c}) \otimes (\{03\} \cdot s_{2,c}) \otimes s_{3,c}$$

$$s'_{2,c} = s_{0,c} \otimes s_{1,c} \otimes (\{02\} \cdot s_{1,c}) \otimes (\{03\} \cdot s_{3,c})$$

$$s'_{3,c} = (\{03\} \cdot s_{0,c}) \otimes s_{0,c} \otimes s_{1,c} \otimes (\{02\} \cdot s_{3,c})$$

Transformasi *cipher* dapat dilakukan proses pembalikan dan diterapkan dalam arah yang berlawanan untuk menghasilkan *invers cipher* yang mudah dipahami untuk algoritma AES. Proses Transformasi *byte* yang digunakan pada *inverse cipher* yaitu *InvShiftRows*, *InvSubBytes*, *Inv MixColumns* dan *AddRoundKey*. Berikut penjelasan proses transformasi dekripsi AES :

#### E. *InvSubBytes*

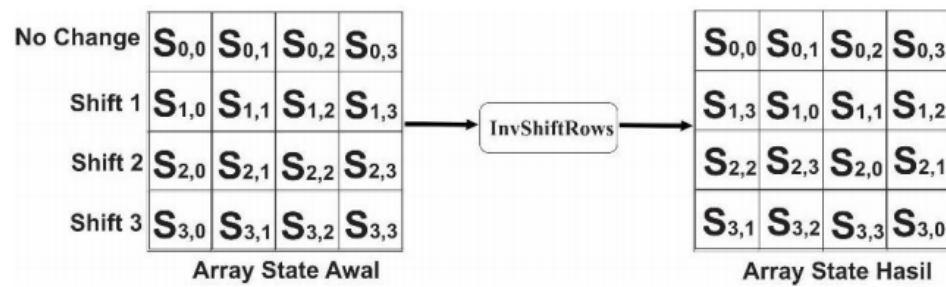
*InvSubBytes* adalah transformasi yang bersifat berkebalikan dengan transformasi *SubBytes*. Pada *InvSubBytes*, setiap elemen pada state dipetakan menggunakan tabel Inverse S-Box. Berikut ini merupakan gambar 2.5 tabel Inverse S-Box.

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

Gambar 2.5 Tabel inverse S-Box [14]

#### F. *InvShiftRows*

*InvShiftRows* merupakan transformasi *byte* yang bersifat berkebalikan dengan transformasi *ShiftRows*. Pada tahap proses transformasi *InvShiftRows*, dilakukan pergeseran bit ke kanan sedangkan pada transformasi *ShiftRows* dilakukan pergeseran bit ke kiri. Skema Gambar 2.6 menunjukkan ilustrasi dari proses transformasi *InvShiftRows*.



Gambar 2.6 Ilustrasi Transformasi InvShiftRows [14]

#### G. *InvMix Columns*

Tahap ini dilakukan perkalian antara setiap kolom dalam *state* dengan matriks perkalian dalam AES. Perkalian dalam matriks dapat dituliskan sebagai berikut: 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. Untuk contoh *InvMix Columns* diperlihatkan pada gambar 2.7 berikut.

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

$$s'_{2,c} = 0B \ 0D \ 09 \ 0 \ s_{2,c}$$

Gambar 2.7 Ilustrasi Persamaan [14]

#### 2.2.7 Rumus Akurasi Algoritma

Setelah proses pengujian selesai, maka diperlukan proses untuk menghitung akurasi dari sebuah algoritma untuk mengetahui bagaimana kinerja algoritma tersebut terhadap aplikasi yang dibuat. Untuk rumus akurasi adalah sebagai berikut:

Rumus 2.4 Rumus Akurasi

$$Akurasi = \frac{Jumlah\ Total\ Berhasil}{Jumlah\ total\ berhasil + gagal} \times 100\%$$

Rumus diatas digunakan untuk menghitung akurasi atau kinerja algoritma, dalam kasus ini adalah algoritma AES dan LZW yang digunakan untuk enkripsi, dekripsi, kompresi dan dekompresi data. Pada rumus diatas jumlah total berhasil



adalah jumlah total pengujian dari metode enkripsi, dekripsi, kompresi dan dekompresi semuanya berhasil, apabila ada salah satu yang gagal maka pengujian dinyatakan gagal. Rumus tersebut digunakan setelah melakukan percobaan terhadap data-data yang akan di enkripsi dan dekripsi.

### **2.3 Algoritma Lempel-ziv-Welch (LZW)**

Lempel-ziv-Welch (LZW) adalah algoritma kompresi lossless universal yang diciptakan Abraham Lempel, Jacob Ziv, dan Terry Welch. Algoritma LZW menggunakan teknik adaptif dan berbasis “kamus”. Pendahuluan LZW adalah LZ77 dan LZ78 yang berkembang oleh Jacob Ziv dan Abraham Lempel pada 1977 dan 1978. Terry Welch mengembangkan teknik tersebut pada 1984 [5].

Algoritma ini melakukan kompresi dengan menggunakan dictionary, dimana fragmen-fragmen teks diganti dengan indeks yang diperoleh oleh sebuah “kamus”.

Prinsip sejenis juga digunakan dalam kode Braille, dimana kode-kode khusus digunakan untuk merepresentasikan kata-kata yang ada. Pendekatan ini bersifat adaptif dan efektif karena banyak karakter dapat dikodekan dengan mengacu pada string yang telah muncul sebelumnya dalam teks [17].

### **2.4 Model Prototyping**

Prototype merupakan versi awal dari sistem *software* atau perangkat lunak untuk mendemonstrasikan percobaan rancangan, konsep-konsep yang digunakan serta memungkinkan untuk menemukan lebih banyak masalah dan solusi.

Tujuan penggunaan metode prototype pada penelitian ini supaya peneliti mendapat gambaran terhadap aplikasi yang akan dibangun melalui tahap pembangunan aplikasi prototype, yang kemudian akan dievaluasi oleh pengguna [18].

Untuk aplikasi prototype yang sudah dievaluasi oleh client (pengguna) akan dijadikan sebagai output hasil akhir dari pengembangan suatu sistem atau aplikasi. Berikut tahapan pengembangan sistem menggunakan metode prototyping:

1. Analisa Kebutuhan

Tahap ini dilakukan untuk mengetahui komponen apa saja yang dibutuhkan oleh sistem. Berikut Kebutuhan sistem yang didefinisikan pada tahap analisa kebutuhan [19]:

- a. Masukan (input) sistem.
- b. Keluaran (output) sistem.
- c. Proses Yang Berjalan dalam sistem.
- d. Basis Data Yang digunakan.

## 2. Desain Sistem

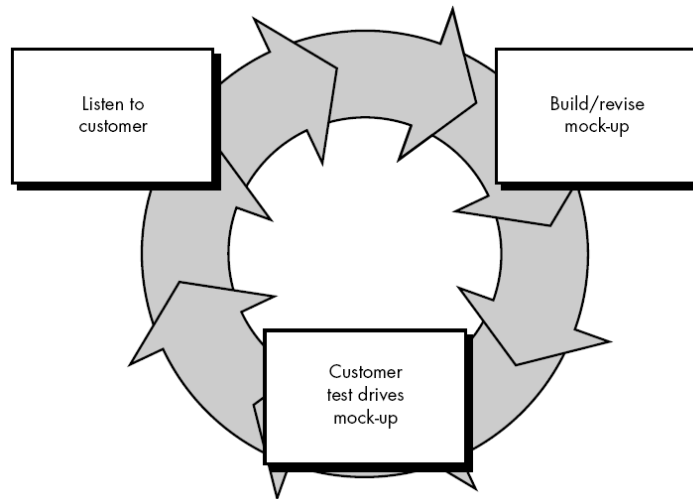
Pada tahap ini dilakukan perancangan relasi dan skema basis data. Biasanya relasional skema basis data dikembangkan dari domain class diagram tiap class yang diidentifikasi secara terpisah. Tujuan desain sistem untuk mengetahui bagaimana sistem akan memenuhi kebutuhan dan tujuan yang telah diciptakan atau dibuat. Tahap desain sistem terdiri dari proses mendesain yang hasilnya berupa spesifikasi dari sistem tersebut. Bagian dari tahap ini bisa berupa konsep desain proses, antarmuka (interface) serta data yang bertujuan untuk menghasilkan spesifikasi system yang cocok atau sesuai dengan kebutuhan yang telah ditentukan.

## 3. Prototype Sistem

Tahap ini merupakan pembuatan rancangan sementara yang difokuskan untuk memberikan gambaran kepada client tentang sistem yang sedang dikembangkan.

## 4. Perbaikan / Evaluasi Prototype

Tahap evaluasi atau perbaikan dilakukan oleh client. Pada tahap ini client memutuskan apakah prototype sudah sesuai dengan kebutuhan pengguna atau harus direvisi lagi. Life cycle dari pengembangan prototype dapat dipaparkan gambar 2.8 di bawah ini.



Gambar 2.8 Life Cycle [19]

## 2.5 Aplikasi Pendukung

### 2.5.1 Pemrograman PHP

Personal Home Page atau yang sering kita kenal PHP adalah bahasa standar yang sering dipakai dalam dunia website. Dasarnya PHP yakni suatu bahasa pemrograman dalam bentuk script yang diletakkan di dalam web server, PHP juga bisa diartikan dengan Hypertext Preprocessor [20].

PHP juga dapat berjalan pada berbagai web server seperti IIS (Internet Information Server), PWS (Personal Web Server), Apache, Xitami. PHP juga mampu berjalan di banyak sistem operasi yang beredar saat ini, diantaranya: Sistem Operasi Microsoft Windows (semua versi), Linux, Mac Os, Solaris. PHP dapat dibangun sebagai modul web server Apache dan sebagai binary yang dapat berjalan sebagai CGI (Common Gateway Interface).

PHP dapat mengirim HTTP header, dapat mengatur cookies, mengatur authentication dan redirect user. Salah satu keunggulan yang dimiliki PHP adalah kemampuannya untuk melakukan koneksi ke berbagai macam software sistem manajemen basis data atau Database Management System (DBMS), sehingga dapat menciptakan suatu halaman web dinamis.

PHP mempunyai koneksitas yang baik dengan beberapa DBMS seperti Oracle, Sybase, mySQL, MySQL, Microsoft SQL Server, Solid, PostgreSQL,

Adabas, FilePro, Velocis, dBase, Unix dbm, dan tidak terkecuali semua database ber-interface ODBC.

Hampir seluruh aplikasi berbasis web dapat dibuat dengan PHP. Namun kekuatan utama adalah konektivitas basis data dengan web. Dengan kemampuan ini kita akan mempunyai suatu sistem basis data yang dapat diakses [21].

### **2.5.2 MySQL**

MySQL adalah sebuah DBMS (Database Management System) menggunakan perintah SQL (Structured Query Language) yang banyak digunakan saat ini dalam pembuatan aplikasi berbasis website. MySQL dibagi menjadi dua lisensi, pertama adalah Free Software dimana perangkat lunak dapat diakses oleh siapa saja. Dan kedua adalah Shareware dimana perangkat lunak berpemilik memiliki batasan dalam penggunaannya.

MySQL termasuk ke dalam RDBMS (Relational Database Management System). Sehingga, menggunakan tabel, kolom, baris, di dalam struktur database-nya. Jadi, dalam proses pengambilan data menggunakan metode relational database. Dan juga menjadi penghubung antara perangkat lunak dan database server [22].

### **2.5.3 Laravel Framework**

Framework Laravel dibuat oleh Taylor Otwell, proyek Laravel dimulai pada April 2011. Awal mula proyek ini dibuat karena Otwell sendiri tidak menemukan framework yang up-to-date dengan versi PHP. Mengembangkan framework yang sudah ada juga bukan merupakan ide yang bagus karena keterbatasan sumber daya. Dikarenakan beberapa keterbatasan tersebut, Otwell membuat sendiri framework dengan nama Laravel. Oleh karena itu Laravel mengisyaratkan PHP versi 5.3 keatas.

Laravel membuat proses development yang menyenangkan bagi pengembang tanpa mengurangi fungsionalitas aplikasi. Dengan harapan, pengembang dapat membuat rangkaian kode-kode terbaik. Laravel berusaha untuk menggabungkan yang terbaik dari apa ada dalam framework web lain, termasuk framework yang menggunakan bahasa lain, seperti Ruby on Rails,

ASP.NET MVC, dan Sinatra. Laravel merupakan framework yang mudah diakses, powerful dan menyediakan tools yang diperlukan untuk skala aplikasi besar. Laravel juga merupakan sebuah aplikasi luar biasa dari sebuah kumpulan program kontrol, sistem migrasi yang ekspresif dan dukungan tools yang Anda butuhkan dalam menguji aplikasi Anda yang terintegrasi dengan beberapa aplikasi lainnya [23].

#### **2.5.4 UML**

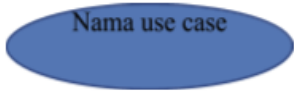




Pada kemajuan serta pengembangan teknik pemrograman berbasis orientasi objek, telah muncul sebuah standarisasi bahasa pemrograman untuk membangun perangkat lunak dengan menggunakan teknik pemrograman berorientasi objek, yaitu UML (Unified Modelling Language). Di Karena adanya kebutuhan dalam pemodelan visual untuk membangun, menggambarkan dan menspesifikasikan serta mendokumentasikan sistem perangkat lunak, maka hal tersebut menjadi dasar kemunculan UML. UML adalah bahasa visual yang digunakan untuk melakukan komunikasi dan pemodelan terkait sebuah sistem dengan menggunakan diagram dan juga teks-teks pendukungnya. Fungsi utama UML hanya untuk melakukan pemodelan. Sehingga penggunaan UML tidak dibatasi oleh metodologi tertentu, walaupun pada kenyataannya UML sering digunakan pada metodologi berorientasi objek [24].

#### **2.5.5 Use Case Diagram**

Use case yakni suatu pemodelan untuk kelakuan (behavior) sistem informasi yang akan dibangun dan dibuat. Use case menjalankan sebuah interaksi aktor antara dengan sistem informasi dibangun. Atau secara ringkas, penggunaan use case bertujuan agar dapat mengetahui fungsi apa saja yang ada pada sebuah sistem informasi serta siapa saja yang memiliki hak untuk menggunakan fungsi – fungsi tersebut. Untuk syarat penamaan pada use case yakni nama dibuat sesimpel mungkin dan dapat dimengerti serta dipahami. Aktor dan use case merupakan dua hal utama yang ada pada pendefinisian use case [24].

Berikut adalah tabel 2.3 yang berisi simbol-simbol dan deskripsi yang digunakan untuk membuat sebuah use case diagram :

Tabel 2.4 Use Case

Simbol	Deskripsi
<b>Use Case</b> 	Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan menggunakan kata kerja di awal fase nama use case.
<b>Aktor</b> 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat dan dibangun sendiri.
<b>Nama aktor</b> 	
<b>Include</b> 	Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankannya use case ini.
<b>System Boundary</b> 	Disimbolkan dalam bentuk kotak yang mewadahi use case, sebagai representasi dari ruang lingkup sistem yang akan dikembangkan. Biasanya digunakan apabila terdapat beberapa alternative sistem, yang dapat dijadikan pilihan.

### 2.5.6 Activity Diagram

Diagram aktivitas yang terdapat pada UML dibuat untuk melengkapi *use case* yang telah dibuat sebelumnya dengan memberikan representasi grafis dari aliran-aliran interaksi di dalam suatu skenario yang bersifat spesifik. Pada diagram aktivitas dijelaskan alur kerja dari sebuah sistem. Mirip dengan *flowchart*, suatu diagram aktivitas menggunakan sebuah kotak berisi lengkung yang menggambarkan fungsi tertentu yang terdapat pada perangkat lunak/sistem yang akan dikembangkan. Pada dasarnya diagram ini berguna juga untuk menganalisa sebuah use case dengan cara menggambarkan kapan aksi-aksi tersebut diperlukan dan dijalankan, memodelkan sejumlah perangkat lunak dengan proses paralel serta menjelaskan urutan algoritma yang kompleks [25].

Berikut adalah tabel 2.4 yang berisi simbol-simbol dan deskripsi yang digunakan untuk membuat sebuah activity diagram :



Tabel 2.5 Activity Diagram [26]

Symbol	Nama	Keterangan
	Swimline	Memisahkan pelaku yang bertanggung jawab terhadap aktifitas yang terjadi.
	Titik awal	Status awal aktifitas, diagram aktifitas memiliki status awal.
	Titik akhir	Status akhir yang dilakukan sistem, sebuah diagram aktifitas yang memiliki sebuah status akhir.
	State	State atau status adalah keadaan sistem pada waktu tertentu. <i>State</i> dapat berubah.
	Activity	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
	Percabangan	Asosiasi percabangan dimana jika ada pilihan aktivitas dari yang terjadi.
	Fork	Kegiatan yang dilakukan secara parallel.