Penerapan *Kriptografi AES Class* Untuk Pengamanan *URL WEBSITE* Dari Serangan *SQL INJECTION*

Ridwan Andriyanto, Khairijal, Devit Satria
Prodi Teknik Informatika, Sekolah Tinggi Teknologi Dumai
Jl. Utama Karya Bukit Batrem II. Kota Dumai, Indonesia
Email : ridwanandriy@gmail.com

ABSTRAK

Seiring penggunaan *website* yang semakin luas dapat menimbulkan berbagai macam tindak kejahatan seperti pencurian, manipulasi data atau informasi penting dari suatu *website* oleh orang yang tidak bertanggung jawab. Dalam pemrograman *web* terdapat dua metode untuk mengirimkan data dari *client* ke *server*, *parameter POST method* dan *parameter GET method*. *GET method request* menempatkan data yang dikirimkan pada *URL web* yang dituju. hal ini menjadi salah satu kelemahan dari *GET method* karena nilai *variable* yang dikirim menggunakan *GET method* dapat dilihat pada bagian *URL* sehingga rentan terhadap serangan *SQL injection*. Salah satu metode kriptografi yang dapat digunakan adalah *Advanced Encryption Standard* (AES). AES adalah pengganti algoritma DES *(Data Encryption Standard)*. Hasil penelitian menunjukkan bahwa Algoritma AES dapat meng*enkripsi* dan men*dekripsi* data *URL* sebuah *website* dengan panjang kunci yang bervariasi, yaitu 128 *bit*, 192 *bit*, dan 256 *bit*. sehingga dapat menyamarkan informasi yang terdapat pada *URL*, *Enkripsi URL* ini menghasilkan keluaran berupa *URL* yang tidak menampilkan *variabel* asli melainkan *chipertext* hasil *enkripsi*.

Kata–kunci : Keamanan, Kriptografi AES, SQL *Injection*, URL

ABSTRACT

As the use of the website is increasingly widespread, it can lead to various kinds of crimes such as theft, manipulation of important data or information from a website by irresponsible people. In web programming, there are two methods for sending data from the client to the server, the POST method parameter and the GET method parameter. The GET request method places the data sent at the intended web URL. This is one of the weaknesses of the GET method because the variable values sent using the GET method can be seen in the URL section, making it vulnerable to SQL injection attacks. One of the cryptographic methods that can be used is the Advanced Encryption Standard (AES). AES is a replacement for the DES (Data Encryption Standard) algorithm. The results showed that the AES Algorithm can encrypt and decrypt the URL data of a website with various key lengths, namely 128 bits, 192 bits, and 256 bits. so that it can disguise the information contained in the URL, this URL encryption produces an output in the form of a URL that does not display the original variable but instead the encrypted ciphertext.

Keywords: *Security, AES Cryptography, SQL Injection, URL*

Pendahuluan

Seiring penggunaan *website* yang semakin luas, dapat menimbulkan berbagai macam tindak kejahatan seperti pencurian, manipulasi data atau informasi penting dari suatu *website* oleh orang yang tidak bertanggung jawab. Dalam pemrograman *web* terdapat dua metode untuk mengirimkan data dari *client* ke *server*. Kedua metode tersebut adalah *parameter POST method* dan *parameter GET method*.

POST method request dimana pengiriman data dilakukan dengan memasukkan data pada sebuah permintaan, sementara *GET method request* yang menempatkan data yang dikirimkan pada *URL web* yang dituju. Hal ini menjadi salah satu kelemahan dari *GET method* karena nilai *variable* yang dikirim menggunakan *GET method* dapat dilihat pada bagian *URL* sehingga rentan terhadap serangan *SQL injection*. Pada tahun 2016 situs PT Srikandi Inti Lestari telah diretas oleh orang yang tidak bertanggung jawab, Dari hasil analisa yang dilakukan terdapat celah keamanan pada parameter *GET method* yang digunakan dalam proses pengambilan data dari *database website.*

Oleh karena itu perlu dilakukan *mekanisme* pengamanan *enkripsi* pada *URL* yang lebih mendalam terkait penggunaan ilmu kriptografi tersebut, salah satu metode kriptografi yang dapat digunakan adalah *Advanced Encryption Standard* (AES). Dari uraian diatas maka dalam pembuatan Tugas Akhir ini penulis mengambil judul **"Penerapan Kriptografi AES *Class* Untuk Pengamanan *URL Website* Dari Serangan *SQL injection*"** (Studi Kasus : *website* PT Srikandi Inti Lestari Dumai).

Landasan Teori

1. Keamanan

Masalah keamanan merupakan salah satu aspek terpenting dari sebuah *website*. Keamanan jaringan adalah kumpulan peranti yang dirancang untuk melindungi data ketika transmisi terhadap pengaksessan, pengubahan dan penghalangan oleh pihak yang tidak berwenang. (Sadikin, 2012).

2. Kriptografi

Kriptografi pada awalnya dijabarkan sebagai ilmu yang mempelajari bagaimana menyembunyikan pesan. Namun pada pengertian modern kriptografi adalah ilmu yang bersandarkan pada teknik matematika untuk berurusan dengan keamanan informasi seperti kerahasiaan, keutuhan, data dan otentikasi entitas. Jadi pengertian kriptografi modern adlah tidak saja berurusan hanya dengan penyembunyian pesan namun lebih pada sekumpulan teknik yang menyediakan keamanan informasi. (Sadikin, 2012)

3. *Advanced Encryption Standard*

Dalam bidang ilmu kriptografi, *Data Encryption Standard* (DES) adalah sebuah algoritma *enkripsi* sandi blok kunci simetrik dengan ukuran blok 64-*bit* dan ukuran kunci 56-*bit*. DES untuk saat ini sudah dianggap tidak aman lagi. Penyebab

utamanya adalah ukuran kuncinya yang sangat pendek (56-*bit*) sehingga rentan terhadap serangan *brute force*. Sejak beberapa tahun yang lalu DES telah digantikan oleh *Advanced Encryption Standard* (AES). (Sadikin, 2012).

Algoritma kriptografi bernama *Rijndael* yang didesain oleh oleh Vincent Rijmen dan John Daemen asal Belgia keluar sebagai pemenang kontes algoritma kriptografi yang diadakan oleh NIST (*National Institutes of Standards and Technology*) milik pemerintah Amerika Serikat pada 26 November 2001. Algoritma *Rijndael* inilah yang kemudian dikenal dengan *Advanced Encryption Standard* (AES). Setelah mengalami beberapa proses standardisasi oleh NIST, *Rijndael* kemudian diadopsi menjadi standard algoritma kriptografi secara resmi pada 22 Mei 2002. Pada 2006, AES merupakan salah satu algoritma terpopuler yang digunakan dalam kriptografi kunci simetrik. (Sadikin, 2012)

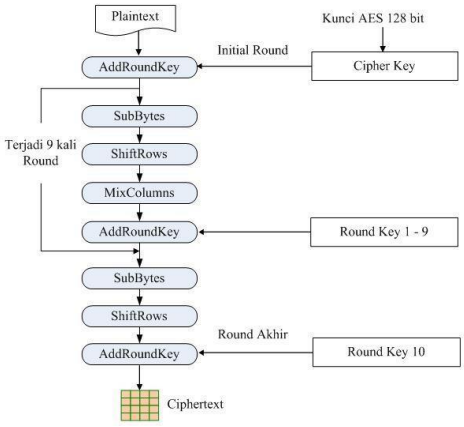
AES merupakan sistem penyandian blok yang bersifat *non-Feistel* karena AES menggunakan komponen yang selalu memiliki *invers* dengan panjang blok 128-*bit*. Kunci AES dapat memiliki panjang kunci *bit* 128, 192, dan 256 *bit*. Penyandian AES menggunakan proses yang berulang yang disebut dengan ronde.

Jumlah ronde yang digunakan oleh AES tergantung dengan panjang kunci yang digunakan. Setiap ronde membutuhkan kunci ronde dan masukan dari ronde berikutnya. Kunci ronde dibangkitkan berdasarkan kunci yang diberikan. Relasi antara jumlah ronde dan panjang kunci diberikan pada tabel 1.

Tabel 1. Panjang kunci Aes

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

Proses enkripsi algoritma AES-128 terdiri dari 4 jenis transformasi *bytes*, yaitu *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey*. Pada Awal proses enkripsi, *state* akan mengalami transformasi *byte AddRoundKey*. Setelah itu, *state* akan mengalami *transformasi SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* secara berulang sebanyak Nr (nilai *round*). Proses ini disebut sebagai *round function*. Pada round terakhir, proses yang dilakukan berbeda dari sebelumnya dimana *state* tidak mengalami transformasi *MixColumns*. Proses enkripsi AES dapat dilihat pada gambar 1 berikut ini.



Gambar 1. Proses Enkripsi AES-128

Secara garis besar proses enkripsi AES-128 dengan kunci 128 bit adalah sebagai berikut:a. *AddRoundKey*: melakukan XOR antara *state* awal (plainteks) dengan *cipher key*. Pada Tahap ini disebut juga *initial round.*

b. *Round* : Putaran sebanyak Nr – 1 kali. Proses yang dilakukan pada setiap putaran adalah:
   1) *SubBytes*: substitusi *byte* dengan menggunakan tabel substitusi (S-box).
   2) *ShiftRows*: pergeseran baris-baris *array state* secara *wrapping*.

c. *MixColumns*: mengacak data pada masing-masing kolom *array state* dengan persamaan sebagai berikut:

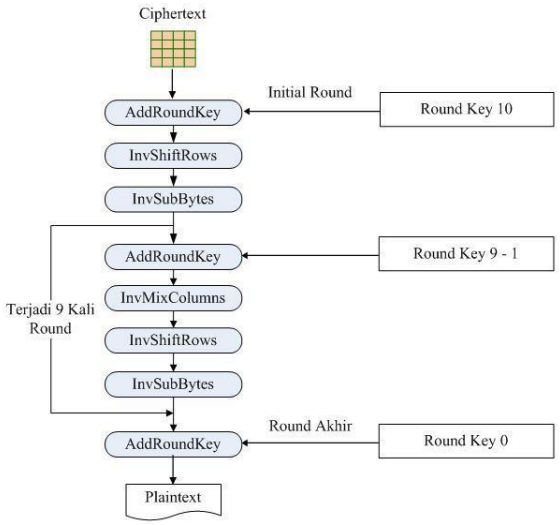
$$A(x) = \{03\}x^2 + \{01\}x^2 + \{01\}x^2 + \{02\} \quad (1)$$

d. *AddRoundKey*: melakukan XOR antara *state* sekarang *round key*.

e. *Final Round*: proses untuk putaran terakhir antara lain:
   1) *SubBytes*
   2) *ShiftRows*
   3) *AddRoundKey*

f. Pada proses terakhir akan menghasilkan karakter atau teks yang berbentuk *chipertext*.

Untuk proses Dekripsi pada AES-128 diperlukan transformasi *cipher* dengan cara dibalik sehingga menghasilkan *inverse cipher* dengan tahapan yaitu: *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*. Algoritma dekripsi dapat dilihat pada gambar 2 berikut ini:



Gambar 2. Proses Dekripsi AES-128

Secara garis besar proses dekripsi AES-128 dengan kunci 128 bit adalah sebagai berikut :

a. *InvShiftRows*: melakukan pergeseran *bit* ke kanan pada setiap blok baris.
b. *InSubBytes*: Setiap elemen pada *state* dipetakan dengan tabel *Inverse* S-Box.

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1x	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2x	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3x	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4x	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5x	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6x	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7x	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8x	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9x	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
ax	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
bx	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
cx	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
dx	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
ex	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
fx	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Gambar 3. *Table Inverse S-Box*

c. *InvMixColumns*: Setiap kolom dalam *state* dikalikan dengan matriks AES.
d. *AddRoundKey*: Mengombinasikan *state array* dan *round key* dengan hubungan XOR.
e. Pada proses terakhir menghasilkan karakter teks asli (*plaintext*).

4. Alat Bantu Perancangan Sistem

4.1. *Software*

Keberadaan perangkat lunak (*Software*) adalah selalu menyertai perangkat keras (*Hardware*). Tetapi tidak semua perangkat lunak muncul untuk dibahas. Hal ini tergantung pada perkembangan teknologi perangkat lunak itu sendiri. Secara fungsinya, perangkat lunak dapat dibagi menjadi 3, yaitu sistem *Software, programing language* dan aplikasi *Software.* A.S, (2011).

4.1.1. PHP

PHP (*Page Hypertext Prepossecor)* adalah sebuah pemrograman yang diDesain agar dapat disisipkan dengan mudah ke halaman *HTML*. *PHP* memberikan solusi sangat murah digunakan (karena gratis digunakan) dan dapat berjalan di berbagai jenis *platform*. A.S, (2011)

4.1.2. MySQL

XAMPP adalah perangkat lunak bebas yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai *server* yang berdiri sendiri, yang terdiri atas program *Apache HTTP server*, *MY*SQL *database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman *PHP* dan *Perl*. A.S, (2011)

4.1.3. XAMPP

XAMPP adalah perangkat lunak bebas yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai *server* yang berdiri sendiri, yang terdiri atas program *Apache* HTTP *server*, MySQL *database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. (Andi, 2008)

4.2. Penggunaan *Hardware*

Spesifikasi Laptop sebagai perangkat keras (*hardware*) yang digunakan dalam mengimplementasikan sistem yang dibuat ini yaitu :

1. *Processor* : AMD E-350 1.6 GHz
2. *RAM* : 4.00 GB DDR3
3. *Hardisk* : 500 GB
4. *Monitor* : LCD 12 *inch*

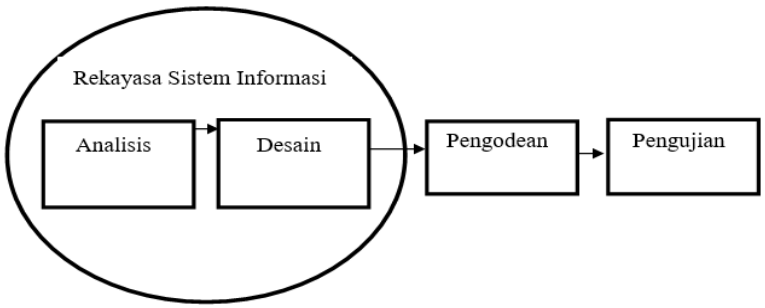
4.3. Alat Bantu Perancangan Sistem

1. Use Case Diagram
2. Activity Diagram
3. Class Diagram
4. Sequence Diagram
5. Flowchart

Metode Penelitian

Metode Pengembangan Sistem

SDLC atau *Software Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya. SDLC sangat dibutuhkan ketika membuat suatu program, karena sering ditemui para programmer yang langsung melakukan pengodean perangkat lunak tanpa menggunakan prosedur atau tahapan pengembangan perangkat lunak yang akhirnya menemui kendala di tengah jalan. A.S, (2011). Pada kasus ini model SDLC yang digunakan ialah model *waterfall.* Berikut gambar model *waterfall :*



Gambar 4. Ilustrasi Model *Waterfall*

Hasil dan Pembahasan

Analisa Proses *Enkripsi* AES 128

AES merupakan algoritma *block cipher* yang menggunakan sistem permutasi dan subtitusi (*P-BOX* dan *S-BOX*) dan memiliki 3 jenis yaitu 128, 192, dan 256 bit. Berdasarkan ukuran blok yang tetap, AES bekerja pada *matriks* berukuran 4x4 di mana tiap-tiap sel *matriks* terdiri atas 1 *byte* (8 bit).

Tabel 2. *Rijndael S-BOX*

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1x	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2x	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3x	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4x	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5x	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6x	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7x	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8x	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9x	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
ax	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
bx	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
cx	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
dx	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
ex	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
fx	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Dalam penelitian ini menggunakan AES 128-bit yang setara dengan 16 karakter. Berikut merupakan contoh penerapan AES 128 bit pada *variabel URL*.

1. Definisikan *plaintext* dan *key*.
   Plaintext : PT. SIL DUMAI 02 (16 karakter)
   Key : abcdefghij012345
2. Lakukan *konversi* *plaintext* dan *key* diatas ke dalam nilai *hexadecimal*.
   Konversi *plaintext* ke *hexadecimal*.

P	T	.		S	I	L		D	U	M	A	I		0	2
50	54	2E	20	53	49	4C	20	44	55	4D	41	49	20	30	32

Konversi *key* ke *hexadecimal*.

a	b	c	d	e	f	g	h	i	j	0	1	2	3	4	5
61	62	63	64	65	66	67	68	69	6A	30	31	32	33	34	35

Plaintext dalam HEX 128 bit: 50 54 2E 20 53 49 4C 20 44 55 4D 41 49 20 30 32
Key dalam HEX 128 bit: 61 62 63 64 65 66 67 68 69 6A 30 31 32 33 34 35

3. Proses *ekspansi* kunci

• Key dalam HEX: 61 62 63 64 65 66 67 68 69 6A 30 31 32 33 34 35
• Ubah menjadi *matriks* 4x4:
- w[0] = (61,62,63,64),
- w[1] = (65,66,67,68),
- w[2] = (69,6A,30,31),

- w[3] = (32,33,34,35)

• Ambil *matriks* terakhir dan geser *byte* ke kiri melingkar. w[3] = (33,34,35,32)
• Subtitusi *byte* dengan *S-BOX*. w[3] = (C3,18,96,23)
• XOR w[3] dengan round constant (01,00,00,00). g(w[3]) = (C2,18,96,23)
• XOR g(w[3]) dengan *matriks* awal. w[4] = w[0] ⊕ g(w[3]) =(A3,7A,F5,47)
• w[5] = w[4] ⊕ w[1] = (C6,1C,92,2F)
• w[6] = w[5] ⊕ w[2] = (AF,76,A2,1E)
• w[7] = w[6] ⊕ w[3] = (9D,45,96,2B)
• Gabung w[4],w[5],w[6],w[7], jadi roundkey pertama adalah: A37AF547 C61C922FAF76A21E9D45962B

Semua kunci yang telah melalui proses ekspansi kunci disajikan pada tabel.

Tabel 3. *Enkripsi key* setiap putaran

Putaran	Hasil (*hexadecimal)*
0	6162636465666768696a303132333435
1	a37af547c61c922faf76a21e9d45962b
2	3165d305a36586abca3f57fde009028f
3	7da52396ca5f63b44dc342b6b83600dc
4	d39d870a60eaa14519a2d0f780e3a3ca
5	6ff264302e7361523066612072696368
6	7cc6a5fab699c64efb5a84f8436c8424
7	1012a0e4b377264f794871b29941733d
8	3b0db6e7ab1a1f4800d2349210263ef7
9	3165d305a36586abca3f57fde009028f
10	99892170b7fa4022879c2102f5f5426a

Proses *Enkripsi* AES 128 bit

• *Add Round Key*, Putaran 0 9

Plaintext dalam HEX 128 bit: 50 54 2E 20 53 49 4C 20 44 55 4D 41 49 20 30 a32
Ambil *plaintext* lalu XOR dengan kunci putaran 0.

50	53	44	49		61	65	69	32
54	49	45	20		62	66	6A	33
2E	4C	4D	30	⊕	63	67	30	34
20	20	41	32		64	68	31	35

31	36	2D	7B
36	2F	3F	13
4D	2B	7D	04
44	48	70	07

Putaran 1, *Substitution Bytes*
Subtitusikan *matriks* terkini hasil XOR yaitu dengan *S-BOX* AES

31	36	2D	7B		C7	05	D8	21
36	2F	3F	13		05	15	75	7D
4D	2B	7D	04	→*S-BOX*	E3	F1	FF	F2
44	48	70	07		1B	52	51	C5

Hasil *S-BOX* AES putaran 1 adalah C705E31B0515F152D875FF51217DF2C5
Selanjutnya geser baris 1,2,3 ke kiri untuk proses *shift rows*

C7	05	D8	21	
05	15	75	7D	→*Shift rows*
E3	F1	FF	F2	
1B	52	51	C5	

			C7	05	D8	21
		05	15	75	7D	
	E3	F1	FF	F2		
1B	52	51	C5			

→ hasil *Shift rows*

C7	05	D8	21
15	75	7D	05
FF	F2	E3	F1
C5	1B	52	51

Putaran 1, *MixColumns*

Transformasi MixColumns mengalikan setiap kolom dari *array state* dengan polinom α (x) mod ($x^4 + 1$). Setiap kolom diperlakukan sebagai polinom 4- suku pada GF(2^8). α(x) yang ditetapkan adalah : $\alpha(x) = \{03\}x^3 + \{01\}x^2 + \{01\} + \{02\}$ s'(x) = α(x) ⊕ s(x). Operasi perkalian yang berlaku adalah Perkalian dengan 1 artinya tidak berubah, Perkalian dengan 2 artinya menggeser byte ke kiri dan Perkalian dengan 3 artinya menggeser byte ke kiri kemudian melakukan XOR dengan nilai awal sebelum digeser, setelah digeser harus dilakukan XOR dengan 0x1 1B apabila nilai yang digeser lebih besar daripada 0xFF.

$S`_0$	=	02	03	01	01	X	C7
$S`_1$		01	02	03	01		15
$S`_2$		01	01	02	03		FF
$S`_3$		03	01	01	02		C5

$S`_0$= ({02} . C7) ⊕ ({03} . 15) ⊕ FF ⊕ C5
= [00000010] · [11000111] ⊕ [00000011] · [00010101] ⊕ [11111111] ⊕ [11000101]

{02.C7}

{C7}.{0.2} = 11000111 digeser ke kiri 1 kali menjadi 110001110, karena 110001110 lebih besar dari 0xFF maka 110001110 di XOR dengan 0x11B atau 100011011 sehingga menjadi
= 10001110 XOR 00011011
= 10010101
{03.15}
15 = 00010101
03 = 11
= 10 XOR 01

Jadi
= {03}.{15} = {10 XOR 01}.{00010101}
= {00010101.10} XOR {00010101.01}

= {00010101.10} XOR {00010101} digeser ke kiri 1 kali menjadi 000101010 kemudian 000101010 di XOR dengan bilangan awal
= 00101010 XOR 00011011 XOR 00010101
= 00110001 XOR 00010101
= 0010 0100

28	BA	29	70
53	FE	23	2C
96	2B	37	A3
6E	91	E7	1E

{FF}= 11111111
{C5} = 11000101
Sehingga S`$_0$ = ({02} . C7) ⊕ ({03} . 15) ⊕ FF ⊕ C5
= 10010101 ⊕ 00100100 ⊕ 11111111 ⊕ 11000101
= 10110001 ⊕ 00111010
= 1000 1011
= 8B

Hasil *MixColumns* kolom seluruhnya :

C7	05	D8	21	x	02	03	01	01
15	75	7D	05		01	02	03	01
FF	F2	E3	F1		01	01	02	03
C5	1B	52	51		03	01	01	02

8B	7C	86	ED
29	E2	55	69
63	B9	95	35
29	BE	F9	35

Setelah dilakukan transformasi *MixColums,* terdapat transformasi *AddRoundKey*. Dengan langkah yang sama seperti yang dijelaskan sebelumnya, hanya saja pada proses XORnya dengan sub kunci yang bersesuaian untuk setiap iterasinya.

AddRoundKey putaran 1

XOR matrik saat ini dengan kunci putaran 1

8B	7C	86	ED	⊕	A3	C6	AF	9D
29	E2	55	69		7A	1C	76	45
63	B9	95	35		F5	92	A2	96
29	BE	F9	35		47	2F	1E	2B

Semua hasil putaran dalam *hexadecimal* yang telah melalui proses *enkripsi* disajikan dalam tabel

Tabel 5. *AddRoundKey* 10 Putaran

Putaran	Hasil (*Hexadecimal*)
0	31364d44362f2b482d3f7d707b130407
1	2853966ebafe2b91292337e7702ca31e
2	5c9bc80f40a231a60e 40055a765d305
3	ad39d87d7a9afa a523967eaa1493e0
4	84f8736c8429c64519a2d0f780e3a3c
5	b431a60eb0a603asa3657fde009028f
6	acab44791012a0e47cc6a5fab69fb5a4
7	7c6a5fab69 e 9c64519a2d0f780e3a3c
8	f186abca3f5b6afb2d9013bab1a1f48
9	0edb6e7ca5f63b 616263646566676
10	82cbecb5fbd1326f72199cd1424f6106

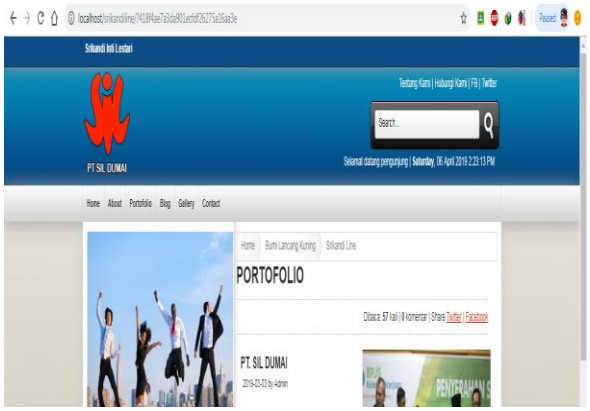
Putaran ke-10 adalah putaran akhir pada proses *enkripsi*. Pada putaran ini tidak ada proses *MixColumns*, hanya melalui proses subtitusi *S-BOX*, *shift row* dan *AddRoundKey*. Jadi *chiphertext* yang ditampilkan ke *URL* adalah hasil putaran ke-10 yaitu:
82cbecb5fbd1326f72199cd1424f6106

Implementasi Aplikasi

Gambar 5. merupakan tampilan halaman utama *website* PT Srikandi Inti Lestari Dumai. Seperti pada penjelasan sebelumnya, penelitian ini

Gambar 5. Tampilan *website*

Setelah halaman atau *page* dipilih maka *URL* akan mengirim *request* ke sistem untuk mengambil data hasil pengolahan informasi yang tersimpan didalam basis data yang ada, kemudian sistem akan melakukan proses *return* hasil *request URL* untuk dikembalikan sesuai data yang diminta tersebut. *URL* yang dikembalikan telah melalui proses *enkripsi* yang diproses sebelumnya dengan algoritma AES sehingga *URL* yang ditampilkan sudah berupa *URL* yang *terenkripsi* (tersandikan).



Gambar 6. Tampilan url telah *terenkripsi*

Pengujian Pada Sistem

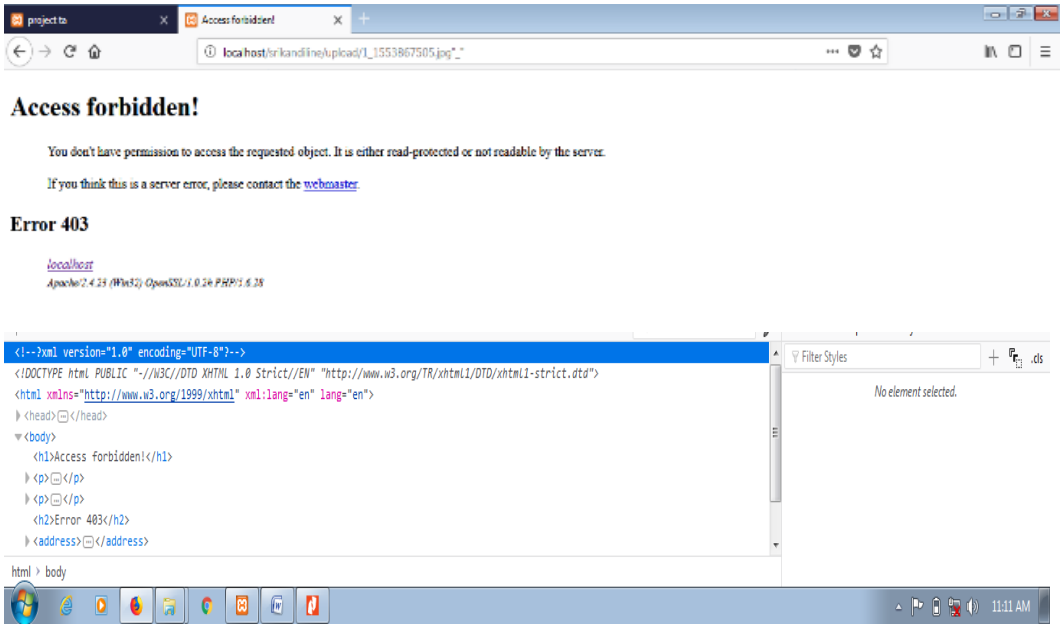
Pengujian pada penelitian ini adalah pengujian menggunakan teknik *Clasical SQL Injection* yang dilakukan terhadap *URL website* hasil *enkripsi* dan penggunaan *firebug pada browser* untuk mengetahui perbandingan waktu akses dan celah *bug* yang terdapat pada *website* sebelum *url web* di*enkripsi* dan *url web* yang telah ter*enkripsi*.

Clasical SQL Injection disebut juga *Standard SQL Injection Testing* adalah cara yang paling sering digunakan secara manual dan biasanya menggunakan metode *UNION* untuk menggabungkan dua *query* dengan tujuan mendapatkan data dari tabel tertentu dari *database*. Pada jenis serangan ini seorang *attacker* akan memanfaatkan pesan error yang ditampilkan oleh *web*.

URL website yang telah *terenkripsi* akan dilakukan serangan berupa beberapa *inject* karakter berbahaya pada *parameter GET* dari *URL* tersebut. Beberapa contoh karakter berbahaya teknik *SQL Injection* adalah special character (!,",$,%,&,',(,),*,+,,,-,.,/,:,;,<,=,>,?,@,[,\,^,_,~,{,|,},`,') Serangan tersebut bisa dilakukan diluar proses baik menggunakan *browser* atau aplikasi lain yang mendukung. Pengujian dilakukan pada proses *ciphertext* yang telah diserang.

Tujuan dari pengujian pada *aplikasi website* yang rentan adalah bagaimana *shellcode* dan *script* berbahaya yang telah dipilih untuk menampilkan informasi penting dari suatu aplikasi *web*. Berikut skenario pengujian yang dilakukan pada penelitian ini adalah sebagai berikut:

1. Mempersiapkan sebuah aplikasi *web* yang tidak menerapkan enkripsi algoritma AES untuk mengamankan alamat *URL*.
2. Mempersiapkan dan memilih *shellcode* atau *script* berbahaya untuk melakukan pengujian penyerangan pada aplikasi *web* yang rentan dengan pengujian kerentanan terhadap *SQL injection*.
3. Kemudian melakukan penyerangan pada *URL encode* yaitu memasukkan *script* berbahaya dan mengakses *file* secara langsung dari alamat *URL* aplikasi *web* untuk melihat hasil pengujian apakah aplikasi *web yang* rentan dapat diakses oleh *attacker*.



Gambar 7. Tampilan Halaman *web* setelah dilakukan *SQL injection*

4. Melakukan pengujian dengan memasukan *script* berbahaya pada aplikasi *web*, dan sampai dimana *script* tersebut dapat merusak aplikasi *web* dan apa saja yang ditampilkan oleh aplikasi *web* jika dilakukan pengujian dengan beberapa *script* berbeda.

Simpulan

Berdasarkan data penelitian yang telah dilakukan, maka dapat diambil kesimpulan bahwa *url website* yang terenkripsi memiliki keamanan sistem *website* yang lebih baik karena menghasilkan keluaran berupa *URL* yang tidak menampilkan *variable* asli melainkan *chipertext* hasil *enkripsi*. Dan berdasarkan hasil pengujian yang dilakukan terdapat selisih waktu *load sistem* dalam menampilkan data sedikit lebih lama dibandingkan dengan *url* tanpa enkripsi.

Dari penelitian dan aplikasi yang telah penulis buat ada beberapa saran yang dapat penulis sarankan terhadap aplikasi ini yaitu :

1. Penerapan Algoritma AES diharapkan juga dapat diterapkan pada proses *parameter POST method*.
2. Macam serangan yang dilakukan untuk pengujian bisa ditambah untuk mendapatkan hasil yang lebih *detail* dan *kompleks*.
3. Dengan pengujian yang dilakukan pada penelitian ini dapat menjadi masukan bahwa perlu adanya penelitian untuk pengembangan terutama dalam hal saat *load sistem* ke *database* diharapkan agar dapat lebih *dioptimalkan* lagi.

Daftar Pustaka

Ariyus, Doni, 2008, "Pengantar Ilmu Kriptografi Teori Analisis dan Implementasi", Yogyakarta : Andi

Gunadhi, Nugraha, (2016). Penerapan Kriptografi Base64 Untuk Keamanan *URL* (Uniform Resource Locator) *Website* Dari Serangan *SQL Injection*. *Jurnal Algoritma Sekolah Tinggi Teknologi Garut*, 491–498.

Manurung, Elni, Enita, 2015, "Pencegahan SQL *Injection* pada *WEB*", Yogyakarta

Negara, L. S., & Kustian, N. (2014). *Sistem Informasi Pengamanan Basis Data*. *7*(2), 188–199.

Primartha, Rifkie, (2011), Penerapan *Enkripsi* dan *Dekripsi* File Menggunakan *Data Encryption Standard* (DES). *ISNN: 2355-4614 / Universitas Sriwijaya*, *3*(2), 371–387.

Rifki, Sadikin, 2012, "Kriptografi Untuk Keamanan Jaringan", Yogyakarta : Andi

Rosa, A. S., & Shalahuddin, M. 2011. Modul Pembelajaran Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek). Bandung: Modula.

Yuniati, V., Indriyanta, G., & Rachmat C., A. (2009). *Enkripsi* Dan *Dekripsi* Dengan Algoritma Aes 256 Untuk Semua Jenis File. *Jurnal Informatika*, *5*(1). *https://doi.org/10.21460/inf.2009.51.69*diakses pada tanggal 18 September 2018.