Seminar Nasional Teknologi Informasi Dan Komunikasi
(SEMNASTIK) X
Palembang-Indonesia, 19 Oktober 2018

PENGARUH IMPLEMENTASI MATRIKS INVOLUTARY TERHADAP HASIL UJI SAC ALGORITMA AES

Novita Angraini

[1] Badan Siber dan Sandi Negara
Jl. Harsono RM no 70 Ragunan Pasarminggu Jakarta Selatan 12550
Email : novita.angraini@bssn.go.id

Abstrak. *Advanced Encryption Standard* (AES) merupakan suatu standar algoritma dan banyak digunakan pada berbagai aplikasi kriptografi. Sifat difusi pada algoritma AES dapat dipenuhi salah satunya dengan operasi perkalian matriks, dimana matriks yang digunakan adalah matriks *Maximal Distance Separable* (MDS). Pada penelitian kali ini diimplementasikan matriks *involutary* yang memenuhi sifat MDS pada algoritma AES. Setelah itu, dianalisis pengaruh dari matriks *involutary* tersebut dengan uji *Strict Avalanche Criterion* (SAC) dan dibandingkan dengan hasil uji SAC algoritma AES asli. Hasilnya algoritma AES dengan matriks *involutary* dan AES asli memenuhi uji SAC pada *round* yang sama, yaitu *round* kedua.

Kata Kunci: Matriks *involutary*, Algoritma AES, *Strict Avalanche Criterion* (SAC).

1. PENDAHULUAN

Ancaman terhadap teknologi komunikasi atau sistem informasi serta pola serangan terhadap sistem keamanan yang dibangun, secara umum berkembang lebih cepat jika dibandingkan oleh metode keamanan untuk mengantisipasi ancaman tersebut. Oleh karena itu, teknik untuk menjaga kerahasiaan data sangat dibutuhkan seiring berkembangnya teknik informasi dan komunikasi. Hal ini dapat diatasi dengan menggunakan teknik enkripsi yang merupakan bagian dari kriptografi. Salah satu teknik enkripsi yang sering diimplementasikan adalah dengan menggunakan algoritma enkripsi standar, yaitu *Advanced Encryption Standard* (AES).

Algoritma *Block Cipher* Rijndael yang diajukan oleh Joan Daemen dan Vincent Rijmen terpilih sebagai *Advanced Encryption Standard* (AES) pada tahun 2001. Algoritma ini adalah algoritma simetrik standar berbasis *Block Cipher* yang mengenkripsi 128-bit blok *input* menjadi 128-bit blok *output*. Algoritma AES menggunakan panjang kunci yang beragam, yaitu 128, 192, dan 256 bit. Berdasarkan panjang kuncinya, AES dikelompokkan menjadi AES-128, AES-192, dan AES-256 (FIPS 197, 2001).

Untuk memperoleh *cipher* pada algoritma AES, pada blok *input* dilakukan operasi *AddRoundKey*(), kemudian dilakukan transformasi berdasarkan fungsi *round* sebanyak 10, 12 atau 14 kali (tergantung panjang kunci yang digunakan). Adapun transformasi dalam fungsi *round* adalah substitusi *byte* menggunakan *S-box* (*S-box*), *shifting rows* dari *state array* (*ShiftRow*), *mixing data* dalam setiap kolom dari *state array* (*MixColumn*) dan Penambahan *round key* ke *state* (*AddRoundKey*) (FIPS 197, 2001).

Operasi *MixColumns* pada enkripsi AES menggunakan matriks MDS sebagai berikut :

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Dapat dijabarkan sebagai berikut

$b_0 = 02 \circ a_0 \oplus 03 \circ a_1 \oplus 01 \circ a_2 \oplus 01 \circ a_3$
$b_1 = 01 \circ a_0 \oplus 02 \circ a_1 \oplus 03 \circ a_2 \oplus 01 \circ a_3$
$b_2 = 01 \circ a_0 \oplus 01 \circ a_1 \oplus 02 \circ a_2 \oplus 03 \circ a_3$
$b_0 = 03 \circ a_0 \oplus 01 \circ a_1 \oplus 01 \circ a_2 \oplus 01 \circ a_3$

Operasi perkalian didefinisikan atas $GF(2^8) = x^8 + x^4 + x^3 + x + 1$. Pada proses dekripsi fungsi *InvMixColumns* menggunakan matriks *inverse* dari matriks MDS yang digunakan. Matriks *inverse* tersebut yaitu sebagai berikut :

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Pada umumnya algoritma *Block Cipher* harus memenuhi konsep difusi dan konfusi. Konfusi adalah penggunaan transformasi penyandian dengan tujuan mempersulit mencari hubungan statistik antara *ciphertext* dan *plaintext* dengan cara menyembunyikan ciri-ciri statistik *plaintext* (Stallings, 2011). Sifat konfusi pada algoritma AES dapat dipenuhi dengan fungsi S-*box*. Difusi adalah menyebarkan statistik dari *plaintext* ke dalam struktur statistik yang melibatkan kombinasi yang panjang dari bit-bit dalam *plaintext* (Stallings, 2011). Sifat difusi pada algoritma AES dapat dipenuhi salah satunya dengan operasi perkalian matriks, dimana matriks yang digunakan adalah matriks *Maximal Distance Separable* (MDS).

Matriks MDS yang digunakan algoritma AES merupakan matriks yang sederhana yang dapat diimplementasikan pada *hardware* atau *software*. Namun matriks invers yang digunakan untuk dekripsi pada algoritma AES berbeda dengan matriks yang digunakan saat enkripsi. Hal ini menyebabkan waktu untuk proses enkripsi dengan dekripsi berbeda sehingga dapat dimanfaatkan penyerang untuk melakukan *side channel attack*, salah satunya *timing attack*. Pada penelitian ini, suatu matriks MDS bersifat *involutory* diimplementasikan pada algoritma AES. Matriks *involutary* adalah matriks yang memiliki invers yang sama dengan matriks itu sendiri. Diharapkan dengan penggunaan matriks *involutary*, waktu untuk proses enkripsi dengan dekripsi tidak jauh berbeda sehingga meminimalkan terjadinya *side channel attack*. Selain itu, dengan penggunaan matriks *involutary*, implementasi pemrograman menjadi lebih efisien karena tidak diperlukan operasi untuk invers matriksnya. Namun perlu dilakukan uji *Strict Avalanche Criterion* (SAC) pada algoritma AES dengan matriks *involutary* untuk mengetahui dan menganalisis tingkat konfusi dan difusi secara keseluruhan algoritma.

Sebuah fungsi $f: \{0,1\}^n \rightarrow \{0,1\}^n$ memenuhi SAC jika untuk semua i dan $j \in (1,2, \ldots, n)$, perubahan bit *input* ke-i akan mengubah bit *output* ke-j dengan probabilitas tepat ½ (Kavut, Yucel, 2000).

Dari penjelasan tersebut dapat diformulasikan menjadi persamaan (1):

$\frac{1}{2^n} W\left(a_j^{e_i}\right) = \frac{1}{2}$ ……(1)

Dari persamaan (1) dapat dimodifikasi untuk menentukan parameter SAC, $k_{SAC}(i,j)$, menjadi persamaan (2):

$k_{SAC}(i,j) = \frac{1}{2^n} W\left(a_j^{e_i}\right) = \frac{1}{2}$ …… (2)

$k_{SAC}(i,j)$ berada pada rentang [0,1] dan merupakan probabilitas perubahan bit *output* ke-*j* ketika bit *input* ke-*i* diubah. Jika probabilitas $k_{SAC}(i,j)$ tidak sama dengan ½ untuk setiap pasangan (*i*, *j*), maka tidak memenuhi SAC. Untuk nilai *n* yang besar akan sulit untuk mendapatkan nilai yang tepat memenuhi SAC. Oleh karena itu pada pengujian SAC penelitian ini, algoritma yang dikatakan memenuhi uji SAC adalah algoritma yang memiliki nilai uji SAC sama atau mendekati nilai uji SAC algoritma AES asli.

2 METODOLOGI PENELITIAN

Pada penelitian ini menggunakan metodologi kajian pustaka dan eksperimen. Kajian Pustaka dilakukan dengan mempelajari dari buku, buku elektronik dan sumber lainnya yang terkait dengan penelitian. Selanjutnya dilakukan eksperimen dengan implementasi ke bahasa pemrograman.

2.1 Tahapan Penelitian

Tahapan penelitian dalam penelitian ini adalah sebagai berikut:
1) Kajian Pustaka tentang Matriks *involutary*, matriks MDS, algoritma AES, dan pengujian *Strict Avalanche Criterion* (SAC).
2) Menerapkan matriks *involutary* ke dalam algoritma AES dan mengimplementasikannya kedalam bahasa pemrograman C.
3) Melakukan pengujian SAC terhadap algoritma AES yang menggunakan matriks *involutary*.
4) Menganalisis hasil uji SAC algoritma AES dengan matriks *involutary*.
5) Penarikan simpulan mengenai pengaruh implementasi matriks *involutary* pada algoritma AES.

2.2 Populasi dan Sampel

Algoritma AES yang digunakan adalah Algoritma AES 128 yang memiliki input *plaintext* dan kunci sebesar 128 bit. Oleh karena itu, populasi *plaintext* dan kunci adalah $\mathbf{2^{128}}$. Setiap bagian dari populasi memiliki peluang yang sama untuk dapat menjadi sampel. Pengambilan sampel dilakukan terhadap variabel bebas yaitu kunci dan plainteks pada algoritma AES. Jumlah sampel yang digunakan dalam pengujian SAC algoritma AES adalah 20.000 dari seluruh populasi input kunci dan *plaintext* yang berjumlah $\mathbf{2^{128}}$.

2.3 Pengumpulan dan Analisis Data

Pengujian SAC pada masing-masing algoritma AES dilakukan dalam dua tahap. Tahap pertama adalah menghasilkan sampel yang digunakan sebanyak 20000 sebagai variabel bebas yang telah ditetapkan. Saat plainteks diperlakukan sebagai variabel bebas maka kunci sebagai variabel kontrol dibuat konstan dengan nilai nol. Hal ini dilakukan untuk mengetahui sifat difusi dari algoritma AES. Demikian pula, ketika kunci diperlakukan sebagai variabel bebas maka plainteks sebagai variabel kontrol dibuat konstan dengan nilai nol. Hal ini dilakukan untuk mengetahui sifat konfusi dari masing-masing algoritma AES. Penggunaan nilai nol konstan pada variabel kontrol untuk menghilangkan pengaruh variabel kontrol. Output dari proses ini adalah nilai variabel bebas (cipherteks). Tahap kedua adalah pengujian sampel yang telah dihasilkan dengan menggunakan uji SAC. Algoritma AES yang digunakan adalah algoritma AES-128 dengan jumlah *round* sebanyak 10 *round*. Algoritma yang dikatakan memenuhi uji SAC adalah algoritma yang memiliki nilai uji SAC sama atau mendekati nilai uji SAC algoritma AES asli.

3 HASIL DAN PEMBAHASAN

Pada bagian ini dijelaskan mengenai matriks *involutory* yang diimplementasikan pada algoritma AES, pembuktian matriks involutory memenuhi matriks MDS, dan perbandingan hasil uji algoritma AES asli dengan algoritma AES dengan matriks *involutory A*.

3.1 Matriks MDS yang Involutory

Sebelum membahas mengenai matriks MDS yang *involutary*, perlu dijelaskan beberapa teori dasar yang terkait dengan matriks MDS yang *involutary*. Sebuah matriks $A \in R_n$ disebut *involutory* jika dan hanya jika $A^2 = I$, dimana I adalah matriks identitas berukuran $n \times n$. R adalah suatu ring komutatif berhingga dengan identitas, sedangkan R_n melambangkan matriks berukuran $n \times n$ atas R.

Untuk memahami mengenai matriks MDS, dijelaskan beberapa istilah seperti kode linier, hamming weight, minimum distance dan MDS Code. Suatu *codeword* dengan panjang n dari pesan dengan panjang k disebut **Kode Linier** $\mathcal{C}$ jika dan hanya jika 2^k codeword membentuk subruang terhadap ruang vektor n tuple atas GF_2 (atau disebut juga F_2^n). Suatu kode linier $\mathcal{C}$ berdimensi k memiliki k buah *codeword* yang saling bebas linear misal $g_1, g_2, \dots, g_k$ maka setiap vector $\boldsymbol{v}$ anggota di $\mathcal{C}$ dapat dinyatakan sebagai kombinasi linier dari $g_1, g_2, \dots, g_k$.

$$\boldsymbol{v} = \alpha_1 g_1 + \alpha_2 g_2 + \cdots + \alpha_k g_k$$

dimana $\alpha_i \in F_2$ untuk $0 < i \leq k$.

Setiap vector tersebut dapat disusun menjadi matriks berukuran $k \times n$ yaitu sebagai berikut :

$$\boldsymbol{G} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_k \end{bmatrix}.$$

Jika $\boldsymbol{u} = (u_1, u_2, \dots, u_k)$ merupakan pesan yang akan diencode, maka *codeword* dari pesan tersebut adalah :

$$\boldsymbol{v} = \boldsymbol{u} \cdot \boldsymbol{G}$$

Matriks $\boldsymbol{G}$ merupakan generator untuk *linear code* $\mathcal{C}$.

Hamming distance $d(\boldsymbol{v}, \boldsymbol{w})$ antara dua *word* $\boldsymbol{v}$ dan $\boldsymbol{w}$ dengan panjang sama adalah banyaknya perbedaan simbol antara $\boldsymbol{v}$ dan $\boldsymbol{w}$. *Weight* dari sebuah *word* adalah banyaknya simbol *non-zero* pada *word* tersebut. *Hamming distance* dari dua *codeword* merupakan *weight* dari *difference*-nya. ***Minimum Distance*** dari suatu *linear code* $\mathcal{C}$, dilambangkan dengan d, didefinisikan sebagai :

$$d = \min\{d(\boldsymbol{v}, \boldsymbol{w}) |\, \boldsymbol{v}, \boldsymbol{w} \in \mathcal{C}, \boldsymbol{v} \neq \boldsymbol{w}\}$$

Minimum distance dari $\mathcal{C}$ adalah maksimal sebesar $n - k + 1$. Suatu *linear code* $\mathcal{C}$ disebut sebagai **MDS code** jika $d = n - k + 1$. Selanjutnya dibahas matriks lain yang berasosiasi dengan $\mathcal{C}$ yaitu matriks $\mathcal{C}^{\perp}$. Suatu ruang vector n tuple atas field F_q berdimensi n. Dengan ini berarti terdapat n buah vector yang saling bebas linier pada F_q^n yang disebut sebagai basis. Jika $\mathcal{C}$ memiliki k buah vektor yang saling bebas linier direpresentasikan ke dalam matriks G, maka terdapat $n - k$ buah vector yang saling bebas linier dan *orthogonal* terhadap vector-vektor di $\mathcal{C}$. Misal $\boldsymbol{H}$ adalah matriks reprensentasi dari $n - k$ buah vektor tersebut

$$\boldsymbol{H} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_{n-k} \end{bmatrix}.$$

Himpunan rentangan yang dibangun oleh vektor-vektor $h_1, h_2, \ldots, h_{n-k}$ membentuk sebuah subruang baru yang disebut ***dual code*** dari $\mathcal{C}$ dan dilambangkan dengan $\mathcal{C}^{\perp}$.

$$\mathcal{C}^{\perp} = \{y \in F_2^n \mid x \cdot y = 0 \text{ untuk setiap } \mathcal{C}\}$$

dimana "$\cdot$ " adalah hasil kali dalam yang didefinisikan sebagai :

$$x \cdot y = \sum_{i=1}^{n} x_i y_i \,, \forall x, y \in F_2^n$$

Sehingga suatu vektor $v \in \mathcal{C}$ jika dan hanya jika $\boldsymbol{H}v^T$=0 . Oleh karena itu, matriks $\boldsymbol{H}$ disebut juga sebagai matriks ***parity check***.

Dengan operasi baris elementer, matriks $\boldsymbol{G}$ dapat dibentuk menjadi :

$$\boldsymbol{G} = [A|I_k]$$

sedangkan matriks $\boldsymbol{H}$ akan berbentuk

$$\boldsymbol{H} = [I_{n-r}|A^T]$$

Jika *linear code* $\mathcal{C}$ merupakan MDS code maka matriks A pada persamaan $\boldsymbol{G} = [A|I_k]$ yang disebut sebagai matriks MDS.

Dalam penelitian ini, matriks *involutory* A yang diimplementasikan pada algoritma AES yaitu sebagai berikut :

$$A = \begin{bmatrix} 01 & 03 & 04 & 07 \\ 03 & 01 & 07 & 04 \\ 04 & 07 & 01 & 03 \\ 07 & 04 & 03 & 01 \end{bmatrix}$$

dimana $A * A = I$. Untuk membuktikan apakah matriks A memenuhi karakteristik matriks MDS, maka dihitung *minimum distance* dari seluruh *codeword* yang dihasilkan oleh matriks A tersebut. Untuk menghitung *minimum distance* adalah dengan menghitung *hamming weight* dari setiap *codeword* yang dihasilkan.

Tabel 1 *Hamming Weight* Seluruh *Codeword* yang Dihasilkan Matriks MDS $\boldsymbol{A}$

		Output Tak Nol				
		0	1	2	3	4
Input Tak Nol	0	0	0	0	0	0
	1	0	0	0	0	1.020
	2	0	0	0	6.120	384.030
	3	0	0	6.120	1.024.080	65.295.300
	4	0	1.020	384.030	65.295.300	4.162.570.275

Pada Tabel 1 menunjukkan frekuensi *hamming weight* dari seluruh *codeword* yang dihasilkan oleh matriks A. Berdasarkan Tabel 1, panjang *minimum distance* dari masing-masing *codeword* yang dihasilkan dari matriks A adalah $d = 5$. Suatu matriks dikatakan MDS jika dan hanya jika $d = n - k + 1$, dimana dalam penelitian ini $k = 4$ dan $n = 8$ sehingga diperoleh $d = 8 - 4 + 1 = 5$. Oleh karena itu, terbukti bahwa matriks A adalah matriks MDS.

3.2 Perbandingan Hasil Uji SAC Algoritma AES dan Algoritma AES dengan Matriks *Involutary*

Berdasarkan hasil uji SAC yang telah dilakukan menggunakan variabel bebas plainteks, dapat dilihat pada Tabel 2 bahwa algoritma AES asli dan algoritma AES dengan matriks *involutary* dapat memenuhi uji SAC. Oleh karena itu pada pengujian SAC menggunakan variabel bebas plainteks, algoritma AES dengan matriks *involutary* tetap mempunyai sifat difusi yang baik. Algoritma AES dengan matriks *involutary* dapat memenuhi uji SAC sejak *round* kedua. Hal ini menunjukkan bahwa pemenuhan uji SAC algoritma AES dengan matriks *involutary* sama dengan algoritma AES asli. Hasil pengujian SAC menggunakan variabel bebas plainteks dapat dilihat pada Tabel 2.

Berdasarkan hasil uji SAC yang telah dilakukan menggunakan variabel bebas kunci, dapat dilihat pada Tabel 3 bahwa algoritma AES asli dan algoritma AES dengan matriks *involutary* dapat memenuhi uji SAC. Oleh karena itu pada pengujian SAC menggunakan variabel bebas kunci, algoritma AES dengan matriks *involutary* tetap mempunyai sifat konfusi yang baik. Algoritma AES dengan matriks *involutary* dapat memenuhi uji SAC sejak *round* kedua. Hal ini menunjukkan bahwa pemenuhan uji SAC algoritma AES dengan matriks *involutary* sama dengan algoritma AES asli. Hasil pengujian SAC menggunakan variabel bebas kunci dapat dilihat pada Tabel 3.

Tabel 2 Hasil Pengujian SAC Menggunakan Variabel Bebas Plainteks

Round	Hasil Uji SAC dari Setiap Algoritma (%)			
	AES asli		AES dengan matriks *involutary*	
	Min	Maks	Min	Maks
1	0	57,075	0	57,075
2	48,61	51,665	48,53	51,66
3	48,575	51,4	48,44	51,46
4	48,53	51,37	48,49	51,37
5	48,805	51,355	48,695	51,465
6	48,625	51,295	48,7	51,39
7	48,725	51,28	48,515	51,375
8	48,6	51,31	48,7	51,475
9	48,245	51,445	48,545	51,275
10	48,67	51,31	48,49	51,5

Tabel 3 Hasil Pengujian SAC Menggunakan Variabel Bebas Kunci

Round	Hasil Uji SAC dari Setiap Algoritma (%)	
	AES asli	AES dengan matriks *involutary*

	Min	Maks	Min	Maks
1	0	100	0	100
2	48,635	51,56	48,29	51,61
3	48,71	51,39	48,665	51,3
4	48,58	51,33	48,525	51,49
5	48,755	51,34	48,66	51,41
6	48,685	51,305	48,695	51,505
7	48,595	51,395	48,545	51,415
8	48,65	51,395	48,33	51,34
9	48,74	51,33	48,745	51,33
10	48,645	51,52	48,565	51,32

4 KESIMPULAN

Berdasarkan hasil penelitian ini, dapat disimpulkan hal-hal sebagai berikut:

1) Matriks A yang diimplementasikan pada algoritma AES merupakan matriks *involutary* dan terbukti memenuhi sifat matriks MDS.
2) Pengujian SAC algoritma AES dengan matriks *involutary* yang telah dilakukan menggunakan variabel bebas kunci dan plainteks, semua algoritma AES dengan matriks *involutary* yang diuji dapat memenuhi uji SAC sejak *round* kedua.
3) Algoritma AES dengan matriks *involutary* tetap memiliki sifat difusi dan konfusi yang baik berdasarkan hasil uji SAC algoritma tersebut.

Referensi

Stallings, William. (2011). Cryptography and Network Security, Fifth edition, pearson education

J.V. Brawley, R.O. Gamble. Involutory matrices over finite commutative rings. Linear Algebra and its Applications, 21.2, 175-188.

Lin, S., Costello, D. J. (2004). Error control coding (Vol. 2), Englewood Cliffs: Prentice Hall

NIST: Federal Information Processing Standards Publication (FIPS) 197. (2001). Springfield. National Institute of Standards and Technology (NIST)

S. Kavut and M.D. Yucel. (2000). On Some Cryptographic Properties of Rijndael, Middle East Technical University