E-ISSN :2623-1700
P-ISSN : 1693-3672

Perbandingan Performa Algoritma AES dan Twofish Menggunakan Metode *Strict Avalanche Criterion* pada Nomor Induk Kependudukan Indonesia

Benedict Ell Nino
Universitas Singaperbangsa Karawang
benedict.nino18072@student.unsika.ac.id

ABSTRAK

Nomor Induk Kependudukan merupakan salah satu data penting yang bersifat rahasia. Salah satu cara agar menjaga agar data NIK yang tersimpan tidak mudah diketahui oleh penyerang, adalah dengan menerapkan salah satu pengamanan data digital yakni kriptografi. Penelitian ini menggunakan algoritma AES dan Twofish sebagai algoritma pengujian sebagai salah satu teknik kriptografi untuk mengamankan data NIK, yang diharapkan dapat mengamankan data NIK dengan aman dan cepat. Penelitian ini menggunakan kriteria *Strict Avalanche Criterion* yaitu kriteria dimana apabila ada perubahan satu bit dalam data masukan berupa plainteks atau kunci, maka akan mengubah data keluaran berupa cipherteks sebanyak 50% dari panjang bit keluaran. Dari hasil penelitian ini menyimpulkan jika algoritma AES dapat digunakan untuk mengenkripsi data NIK dibanding Twofish karena memiliki nilai *margin of error* yang lebih sedikit dibanding Twofish untuk memenuhi kriteria SAC. Selain itu, algoritma AES memiliki kecepatan enkripsi yang lebih cepat dibanding Twofish.

Kata kunci: Kriptografi, AES, Twofish, Avalanche Effect, Strict Avalanche Criterion, dan Algoritma

ABSTRACT

The Population Identification Number is one of the important data that is confidential. One way to keep the stored NIK data from being easily known by attackers, is to apply one of the digital data security, namely cryptography. This research uses AES and Twofish algorithms as testing algorithms as one of the cryptographic techniques to secure NIK data, which is expected to secure NIK data safely and quickly. This research uses the Strict Avalanche Criterion which is a criterion where if there is a change of one bit in the input data in the form of plaintext or key, it will change the output data in the form of ciphertext by 50% of the output bit length. The results of this study conclude that the AES algorithm can be used to encrypt NIK data compared to Twofish because it has a margin of error value that is less than Twofish to meet the SAC criteria. In addition, the AES algorithm has a faster encryption speed than Twofish.

Keywords: Cryptography, AES, Twofish, Avalanche Effect, Strict Avalanche Criterion, and Algorithm

E-ISSN :2623-1700
P-ISSN : 1693-3672

PENDAHULUAN

Kemajuan teknologi di bidang informasi turut memajukan media komunikasi sebagai sarana penyampaian informasi, sehingga memudahkan akses terhadap media komunikasi oleh pengguna. Namun kemudahan akses ini dapat memberikan dampak terhadap keamanan dari informasi atau pesan tersebut, dimana informasi atau pesan tersebut sangat rentan untuk diketahui, diambil, dimanipulasi, dan disalahgunakan oleh pihak-pihak yang tak bertanggung jawab.

Keamanan tentang data dan informasi diatur dalam UU Nomor 11 Tahun 2008 tentang ITE pasal 26 ayat 1 yang berbunyi

"*penggunaan setiap informasi melalui media elektronik yang menyangkut data pribadi seseorang harus dilakukan atas persetujuan orang yang bersangkutan*"

sehingga hal ini merupakan sebuah tantangan bagi penyelenggara sistem elektronik dalam mengamankan data yang ada pada sistem.

Dalam rangka mewujudkan keamanan data, penyelenggara sistem elektronik diharapkan mampu memaksimalkan segala komponen pada bagian *Back End* untuk mewujudkan hal tersebut. Salah satu komponen yang dapat mempengaruhi keamanan data pada bagian *Back End* adalah kriptografi.

Kriptografi dapat diklasifikasikan menurut jenis kunci dan cara operasi. Berdasarkan jenis kunci, kriptografi dibagi menjadi dua yaitu kunci simetris dan kunci asimetris. Kunci simetris adalah kunci tunggal yang digunakan dalam proses enkripsi dan dekripsi, sedangkan kunci asimetris adalah dua kunci dimana satu kunci yang bersifat publik digunakan untuk proses enkripsi dan kunci privat digunakan sebagai proses dekripsi.

Selain jenis kunci, kriptografi juga dapat diklasifikasikan berdasarkan cara operasinya yaitu *stream cipher* dan *block cipher*. *Stream cipher* beroperasi pada blok-blok karakter pada proses enkripsi dan dekripsi, sedangkan *block cipher* beroperasi pada blok-blok bit yang ada pada karakter.

Contoh dari kriptografi berjenis kunci simetris yang beroperasi pada *blockcipher* adalah *Advanced Encryption Standard* (yang selanjutnya disebut AES), dan Twofish. AES adalah penerus algoritma DES yang lama dengan varian 128 bit, 192 bit, dan 256 bit, yang dipublikasikan oleh Rijndael. Sedangkan Twofish adalah penerus algoritma Blowfish yang lama dengan varian yang mirip dengan AES, dan dipublikasikan oleh Bruce Schneier.

Namun sepanjang tahun 2020 hingga 2021, terdapat 10 kasus kebocoran data di Indonesia dimana ½ kasus kebocoran data tersebut mengalami kebocoran pada data Nomor Induk Kependudukan.[1][2]

Jika kondisi ini dibiarkan dan tidak mendapat perhatian yang serius akan menurunkan kepercayaan masyarakat (selaku pemilik data) terhadap kredibilitas penyelenggara system elektronik. Salah satu langkah yang biasa dilakukan dalam menyelesaikan permasalahan tersebut adalah dengan mengaplikasikan kriptografi pada sebuah sistem.[3]

E-ISSN :2623-1700
P-ISSN : 1693-3672

METODE

Penelitian ini menggunakan metode komparatif dimana penelitian berfokus pada perhatian kepada kelompok subyek penelitian, kemudian dilanjutkan dengan memperhatikan variabel yang diteliti yang terdapat dalam kelompok yang akan dikomparasikan. (Juwa, P. 2015). Metodologi ini memiliki empat tahapan utama, yakni melakukan penelaahan kepustakaan, merancang pendekatan atau pengujian, melakukan validasi, dan mengumpulkan serta menganalisis data untuk membuat kesimpulan.

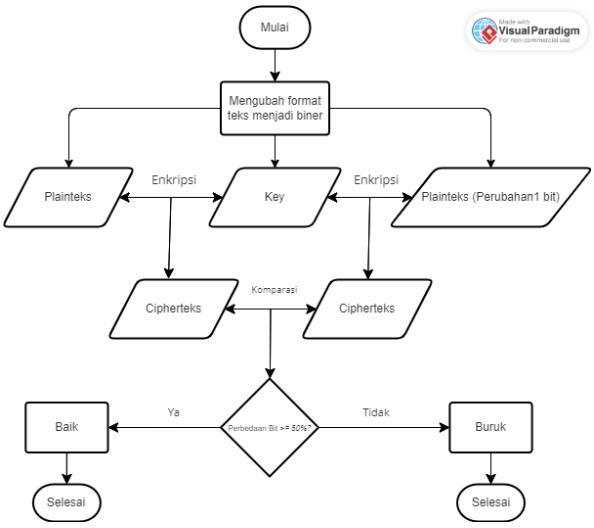
Pengumpulan data dilakukan untuk menemukan beberapa informasi-informasi terkait algoritma (AES dan Twofish) dan kriteria SAC yang dijadikan sebagai objek dan metode dalam penelitian ini. Selain itu, data tentang kebocoran data KTP dari penyelenggara sistem elektronik didapat dari sumber berita daring. Untuk Data NIK sebagai bahan uji coba didapat dari Kp. Pulojahe (Jakarta Timur) dan Kp. Ciparay (Bandung). Perangkat lunak yang digunakan untuk mengenkripsi dan mencari nilai *Avalanche* Effect dari setiap algoritma adalah CrypTool yang bersifat *open source*.

Pengujian data dilakukan menggunakan variabel kunci dengan nilai tetap sepanjang 128 bit, yaitu 0x4B54504B54504B54504B54504B54504B, dan menggunakan mode ECB. Sedangkan variabel plainteks merupakan variabel dinamis yang berubah sebanyak sampel yang ditentukan dalam pengujian SAC. Satu cipherteks hasil plainteks awal digunakan sebagai pembanding terhadap sampel cipherteks lainnya. Bit berbeda didapat dengan mencari perbedaan susunan bit 0 dan 1 diantara kedua cipherteks. Setelah perbedaan bit didapat, langkah selanjutnya adalah melakukan penghitungan nilai *Avalanche Effect* dengan membagi total nilai bit yang berbeda dengan jumlah panjang bit yang terdapat pada algoritma.

Proses validasi dilakukan dengan menghitung nilai rata-rata *Avalanche Effect* dengan nilai ideal SAC (50%) untuk mendapatkan nilai error yang dihasilkan dari algoritma AES dan Twofish.

Algoritma yang memiliki nilai akhir terkecil, dapat disimpulkan sebagai algoritma yang memiliki nilai difusi baik sesuai standar SAC. Selain itu, algoritma ini dapat dijadikan rekomendasi untuk digunakan dalam mengenkripsi data Nomor Induk Kependudukan.

2.1 Analisa Avalanche Effect



Gambar 2.1 Flowchart Analisa Avalanche Effect

Avalanche Effect adalah suatu properti dari algoritma yang terdapat pada kriptografi, yang umumnya terdapat pada *block cipher* dan fungsi hash. Properti ini diperkenalkan oleh Shannon, dan terminologinya digunakan pertama kali oleh Horst Feistel.

Properti ini memiliki konsep dimana jika sebuah masukkan pada enkripsi kriptografi mengalami pergantian bit (contohnya, pembalikan satu bit), maka keluaran dari enkripsi tersebut akan menghasilkan perubahan bit secara signifikan (idealnya adalah 50% dari panjang bit).[4] Dalam kasus penggunaan block cipher berkualitas tinggi, perubahan kecil pada kunci atau plainteks dapat membuat perubahan drastis pada cipherteks yang dihasilkan.

Jika sebuah kriptografi *block cipher* atau fungsi hash tidak memiliki nilai derajat *Avalanche Effect* yang ideal, maka kriptografi atau fungsi hash tersebut memiliki nilai acak (yang merupakan kunci dari efisiensi teknik enkripsi)[5] yang buruk. Hal ini dapat mempermudah kriptanalis untuk membuat prediksi mengenai masukan, hanya dengan

menggunakan keluaran. Konsep ini memiliki persamaan sebagai berikut[6];

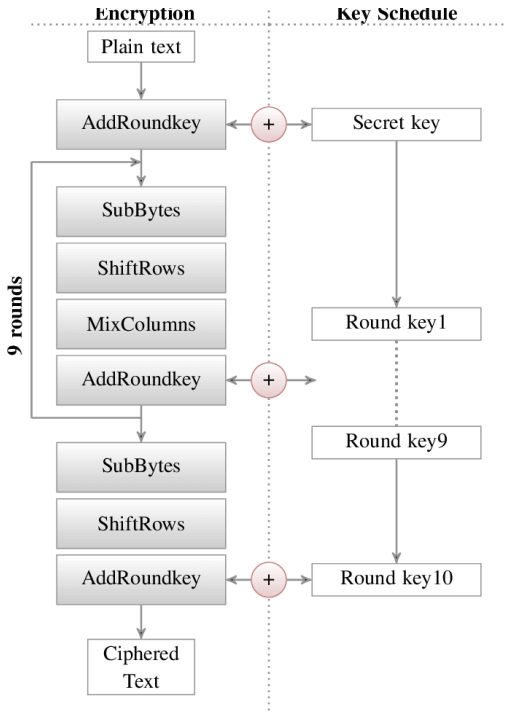
$$AE = \frac{Jumlah\ bit\ berbeda}{Panjang\ bit}\ x\ 100\%$$

Sebelum melakukan analisa, semua masukan harus diubah kedalam bilangan biner. Hal ini dikarenakan nilai *Avalanche Effect* didapat dari jumlah perbedaan bit pada masukan.

Dalam melakukan analisa nilai *Avalanche Effect* dengan kunci terbatas, diperlukan dua buah plainteks dan satu kunci sebagai masukan. Masukan ini digunakan untuk melakukan komparasi perbedaan bit apabila ada perubahan salah satu bit di salah satu plainteks.

Setelah hasil enkripsi berupa cipherteks didapat, selanjutnya adalah melakukan perbandingan bit dari kedua cipherteks. Jika hasil perbedaan bit lebih dari sama dengan 50% dari panjang bit, maka algoritma tersebut memiliki nilai *Avalanche Effect* yang baik. Sebaliknya, jika hasil perbedaan bit kurang dari 50% dari panjang bit, maka algoritma tersebut memiliki nilai *Avalanche Effect* yang buruk.

2.2 Analisa Avalanche Effect AES



Gambar 2.2.1 Flowchart Enkripsi AES

Proses enkripsi dimulai dengan melakukan *Key Expansion* sebanyak 10 iterasi. Proses ini melibatkan proses *Sub Bytes*, *Shift Rows* vertikal, operasi XOR dengan matriks sebelumnya, dan operasi XOR dengan Kotak RCON.

TABLE II
THE CONTENT OF THE RCON[ROUND]

Round	1	2	3	4	5	6	7	8	9	10
Rcon[]	01	02	04	08	10	20	40	80	1b	36
	00	00	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00	00	00

Gambar 2.2.2 Kotak RCON Algoritma AES

Kemudian plainteks dan kunci diindeks kedalam matriks berukuran 4x4 untuk operasi XOR. Setelah operasi XOR, operasi selanjutnya adalah proses *Sub Bytes* dengan menggunakan properti kotak S pada AES.

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
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 2.2.2 Kotak S Algoritma AES

Setelah proses substitusi, proses berikutnya adalah *Shift Rows*. Pada proses ini, baris pada matriks akan mengalami pergeseran ke kiri sesuai posisi baris. Setelah proses pergeseran, selanjutnya matriks akan dikali dengan matriks *MixColumn*.

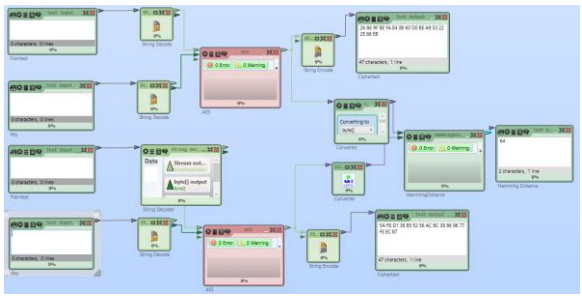
$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02\ 03\ 01\ 01 \\ 01\ 02\ 03\ 01 \\ 01\ 01\ 02\ 03 \\ 03\ 01\ 01\ 02 \end{bmatrix} \times \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Gambar 2.2.3 Operasi *MixColumn* AES

Setelah proses perkalian matriks, hasil perkalian akan melalui operasi XOR dengan kunci hasil pembangkitan sesuai dengan iterasi, untuk menciptakan matriks untuk iterasi selanjutnya.

E-ISSN :2623-1700
P-ISSN : 1693-3672

Proses diatas diulangi sebanyak 9 iterasi dan pada iterasi terakhir, tahapan yang dilakukan hanyalah operasi *Sub Bytes*, *Shift Rows*, dan operasi XOR dengan ronde kunci.

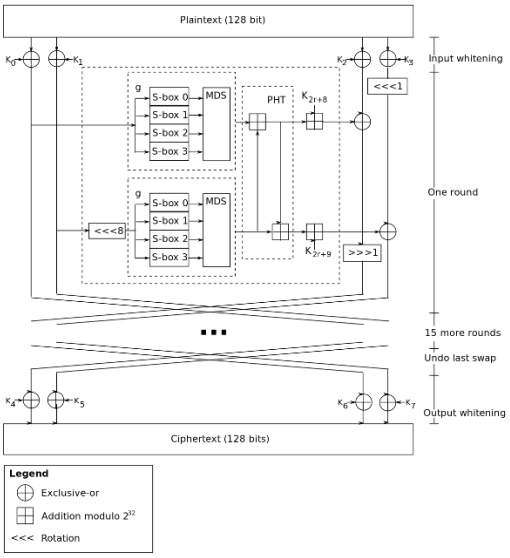


Gambar 2.2.4 Proses Analisa *Avalanche Effect* AES Menggunakan CrypTool

Proses analisa *Avalanche Effect* dimulai dengan menggunakan proses enkripsi diatas dengan dua buah plainteks yang memiliki perbedaan bit sebanyak 1 bit. Masing-masing plainteks dienkripsi dengan kunci yang sama untuk menghasilkan cipherteks yang berbeda.

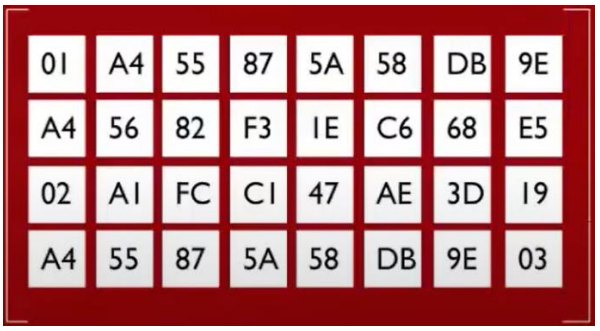
Kedua cipherteks hasil enkripsi akan melalui operasi XOR untuk mendapatkan jumlah perbedaan bit. Jika hasil perbedaan bit >= 50% panjang cipherteks (dalam bit), maka algoritma tersebut memiliki performa yang bagus dan nilai difusi yang baik sesuai standar SAC.

2.3 Analisa *Avalanche Effect* Twofish



Gambar 2.3.1 Flowchart Enkripsi Twofish

Dalam proses penjadwalan kunci, terdapat dua buah set subkunci (K dan S), dimana set K digunakan pada putaran iterasi dan S digunakan pada kotak Reed Solomon.



Gambar 2.3.2 Kotak Reed Solomon

Total subkunci yang digunakan adalah 40 dengan beberapa ketentuan seperti;

1. 4 subkunci digunakan dalam *Input Whitening*
2. 4 subkunci digunakan dalam *Output Whitening*
3. 32 subkunci yang tersisa digunakan dalam setiap ronde pada proses enkripsi (Fungsi F)

Set subkunci S memiliki 2 bagian subkunci S0 dan S1, dimana keduanya berasal dari kunci awal (M) yang dikalikan dengan matriks Reed Solomon. Dua set subkunci ini nantinya akan digunakan dalam proses kotak S algoritma Twofish. Sedangkan subkunci K memiliki 4 bagian subkunci, dimana subkunci dibagi menjadi subkunci ganjil dan genap untuk digunakan pada proses sesudah *Pseudo Hadamard Transformation*.

Pada proses enkripsi, masukan melakukan proses *Input Whitening*. Dalam proses ini, plainteks dan key akan melakukan operasi XOR dan menghasilkan 4 bagian, yaitu R0, R1, R2, dan R3. Bagian R0 dan R1 selanjutnya akan memasuki fungsi G, dimana bagian R1 akan mengalami *Left Shift Bit* sebanyak 8 bit.

Dalam fungsi G, masing masing bagian R dipecah menjadi 4 bagian X berukuran 8 bit untuk dapat diproses dalam kotak S yang bergantung pada kunci. Setiap kotak S memiliki masukan dan keluaran sebesar 8 bit. Sedangkan dalam kotak S, keempat bagian X melakukan operasi permutasi q0 dan q1, dimana bagian ganjil beroperasi dengan permutasi q0 dan bagian genap beroperasi dengan permutasi q1.

q0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
t0	8	1	7	D	6	F	3	2	0	B	5	9	E	C	A	4
t1	E	C	B	8	1	2	3	5	F	4	A	6	7	0	9	D
t2	B	A	5	E	6	D	9	0	C	8	F	3	2	4	7	1
t3	D	7	F	4	1	2	6	E	9	B	3	0	8	5	C	A

q1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
t0	2	8	B	D	F	7	6	E	3	1	9	4	0	A	C	5
t1	1	E	2	B	4	C	3	7	6	D	A	5	F	9	0	8
t2	4	C	7	5	1	6	9	A	0	E	D	8	2	B	3	F
t3	B	9	5	1	C	3	D	E	6	4	7	F	2	0	8	A

Gambar 2.3.3 Kotak Permutasi q1 dan q2

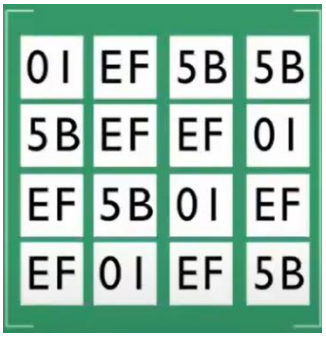
Dalam proses permutasi, permutasi q0 dan q1 memiliki struktur internal yang sama, dan hanya berbeda pada isi kotak S yaitu t0 hingga t3. Permutasi ini diproses dengan membagi masukan menjadi dua bagian (a dan b) dengan panjang 4 bit. Bagian a melakukan operasi XOR dengan bagian b dan menghasilkan t0, sedangkan bagian b dilakukan operasi *Right Shift Bit* sebanyak 4 bit dan melakukan operasi XOR dengan bagian a serta bagian a yang telah melewati operasi modulo 8ai mod 16. Langkah ini dilakukan sekali lagi untuk mendapatkan t2 dan t3 sebagai keluaran permutasi, dimana ukuran keluaran ini memiliki panjang 8 bit.

Setiap operasi XOR pada proses ini, bagian a dan b akan mengalami substitusi kotak permutasi. Kotak permutasi ini bekerja sesuai dengan jenis permutasi (q1 atau q2) dan keluaran operasi XOR (t0 hingga t3).

Saat proses permutasi berakhir, langkah selanjutnya adalah melakukan operasi XOR dengan kunci S yang dihasilkan dari operasi penjadwalan kunci. Setelah operasi XOR dilakukan, selanjutnya 4 bagian ini kembali melakukan operasi permutasi q0 dan q1. Namun pada operasi permutasi kedua, pembagian operasi permutasi mengalami perbedaan dimana kedua bagian pertama mengalami permutasi q0 dan kedua bagian terakhir mengalami permutasi q1.

Setelah proses permutasi kedua, keluaran operasi permutasi mengalami operasi XOR kembali dengan kunci S. Langkah terakhir, keempat bagian mengalami operasi permutasi terakhir yang berkebalikan dengan operasi permutasi pertama, dimana bagian ganjil beroperasi dengan permutasi q1 dan bagian genap beroperasi dengan permutasi q0.

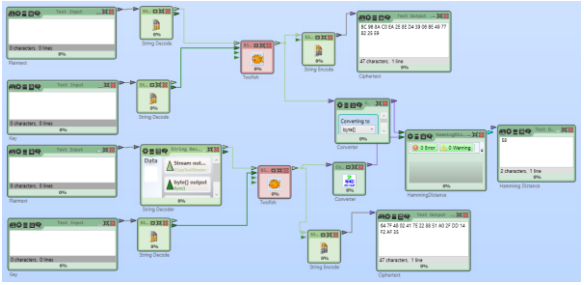
Setelah proses ini, keempat bagian ini melakukan operasi perkalian dengan matriks MDS dan menghasilkan keluaran T0 dan T1 sebesar 32 bit.



Gambar 2.3.4 Matriks *Maximum Distance Separable*

Selanjutnya keluaran pada operasi matriks MDS akan melakukan operasi PHT dimana operasi ini membutuhkan 3 buah masukan, yaitu T0, T1, dan r (sesuai ronde enkripsi). Pada operasi PHT, keluaran akan saling melakukan operasi XOR dan bagian T0 akan melakukan operasi XOR dengan kunci yang diperluas K_{2r+8} dan T1 akan melakukan operasi XOR dengan kunci yang diperluas K_{2r+9} (dimana r merupakan ronde dari enkripsi). Keluaran dari operasi PHT ini adalah F0 dan F1 dengan masing-masing berukuran 32 bit.

F0 dan F1 kemudian akan melakukan operasi XOR dengan bagian R2 dan R3 yang telah mengalami *Left Shift* sebanyak 1 bit. Hasil XOR dari F0 dan R2 mengalami *Right Shift* sebanyak 1 bit. Proses ini akan menghasilkan bagian C1 dan C2. Selanjutnya, R0, R1, C1, dan C2 mengalami *Shift* sebanyak 2 bit, sehingga bagian C1 dan C2 digunakan sebagai R0 dan R1 pada langkah selanjutnya, dan R0 dan R1 digunakan pada operasi XOR pasca operasi PHT.



Gambar 2.3.5 Proses Analisa *Avalanche Effect* Twofish Menggunakan CrypTool

Sama seperti AES, proses analisa *Avalanche Effect* dimulai dengan menggunakan proses enkripsi diatas dengan dua buah plainteks yang memiliki perbedaan bit sebanyak 1 bit. Masing-masing plainteks dienkripsi dengan kunci yang sama untuk menghasilkan cipherteks yang berbeda.

Kedua cipherteks hasil enkripsi akan melalui operasi XOR untuk mendapatkan jumlah perbedaan bit. Jika hasil perbedaan bit >= 50% panjang cipherteks (dalam bit), maka algoritma tersebut memiliki performa yang bagus dan nilai difusi yang baik sesuai standar SAC.

2.4 Kriteria *Strict Avalanche Criterion*

Kriteria SAC adalah bentuk formalisasi dari konsep *Avalanche Effect*. Kriteria ini terpenuhi apabila perubahan satu bit pada masukan, akan mempengaruhi hasil keluaran sebanyak 50% sebagai nilai ideal. Kriteria ini dibangun pada konsep *avalanche* dan *completeness* yang terdapat pada kriptografi. Kriteria ini dapat digeneralisasi dalam banyak definisi yang berujung pada dua kemungkinan definsi.

Pertama, definisi asli SAC yang digagas oleh Webster dan Taraves, dimana properti ini didapat ketika banyak masukan bit yang diubah secara berkala (misalnya i_1, i_2, ...i_m dimana m merupakan panjang bit dari algoritma kriptografi).

Kedua adalah definisi SAC yang digagas oleh Forre, dimana properti ini didapat dari memperbaiki bit masukan tertentu dengan nilai konstan dan harus berlaku pada semua subsistem yang dihasilkan.

Untuk membedakan kedua definisi ini, ada dua terminologi yang dapat digunakan;

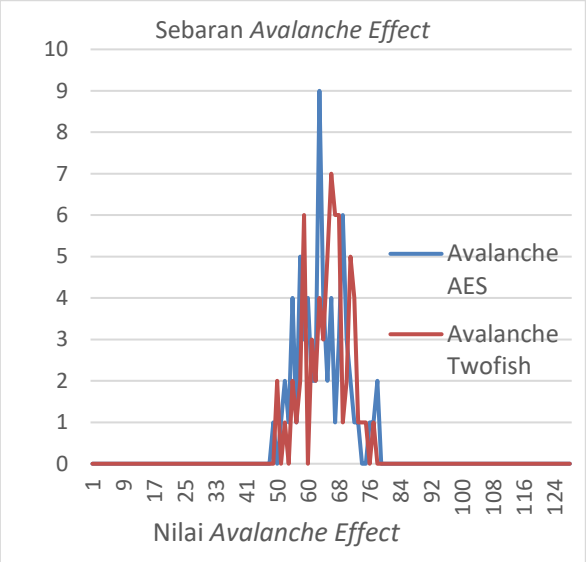
1. SAC1 merupakan definisi pertama SAC dimana untuk vektor masukan sebanyak n-bit dengan syarat $1 \leq i \leq n$.
2. SAC2 merupakan definisi Forre dimana untuk vektor masukan sebanyak n-bit dengan syarat $1 \leq i \leq n-2$.

Kriteria ini dapat tercapai apabila perubahan satu bit pada masukkan mengakibatkan perubahan yang signifikan pada cipherteks sebesar setengah dari panjang cipherteks. Algoritma yang memiliki nilai AE mendekati 50% maka berhasil memenuhi kriteria ini.[7][8] Pada penelitian ini, penulis menggunakan definisi SAC menurut Webster dan Taraves.

E-ISSN :2623-1700
P-ISSN : 1693-3672

HASIL DAN PEMBAHASAN

Hasil dari pengujian enkripsi NIK dari kedua algoritma memberikan hasil sebaran nilai *Avalanche Effect* sebagai berikut;



Gambar 3.1 Grafik Persebaran Nilai *Avalanche Effect*

Berdasarkan grafik diatas, dapat dilihat bahwa nilai *Avalanche Effect* dari algoritma AES terpusat di nilai 63 dengan jumlah 9 dari 66 data, sedangkan pada algoritma Twofish nilai *Avalanche Effect* terpusat di nilai 66 dan 59 dengan masing masing data berjumlah 7 dari 66 data dan 6 dari 66 data. Hal ini menunjukkan bahwa algoritma AES memiliki properti difusi yang baik sehingga dapat menghasilkan nilai *Avalanche Effect* yang ideal menurut kriteria *Strict Avalanche Criterion*, dibanding dengan algoritma Twofish.

Selain itu, dari hasil penelitian ini didapatkan rata-rata nilai *Avalanche Effect* dari algoritma AES adalah 63.72, sedangkan Twofish mendapat rata-rata nilai *Avalanche Effect* sebesar 65.03. Sehingga nilai *margin of error* algoritma AES untuk mencapai nilai ideal *Avalanche Effect* sesuai kriteria SAC adalah 0.27, sedangkan Twofish memiliki nilai *margin of error* sebesar 1.03.

Ada tiga faktor yang dapat mempengaruhi nilai *Avalanche Effect* pada suatu algoritma, yakni;

1. Jumlah Operasi Algoritma

Jumlah operasi yang terdapat pada algoritma dapat mempengaruhi hasil nilai *Avalanche Effect* yang didapat dari algoritma tersebut. Misalnya pada algoritma AES-128 dengan kunci yang bebas, jika tanpa operasi *MixColumn* akan menghasilkan nilai *Avalanche Effect* sebesar 51,27% dari total panjang bit. Sedangkan pada algoritma AES-128 dengan plainteks bebas, jika tanpa operasi *Shift Rows* akan menghasilkan nilai *Avalanche Effect* sebesar 51,28% dari total panjang bit.[9]

Sehingga algoritma AES-128 dapat memenuhi kriteria SAC walaupun hanya menggunakan beberapa operasi saja dalam proses enkripsi.

2. Jumlah Iterasi Algoritma

Jumlah iterasi yang dijalankan pada algoritma dapat mempengaruhi hasil nilai Avalanche Effect yang didapat dari algoritma tersebut. Misalnya pada AES yang beroperasi pada panjang bit sebesar 128 bit dengan pesan bebas, pada iterasi ke-5 sudah menghasilkan nilai Avalanche Effect yang hampir memenuhi kriteria SAC sebesar 47.031%.

Sedangkan pada AES yang beroperasi pada panjang bit sebesar 192 bit, pada iterasi ke-2 sudah menghasilkan nilai Avalanche Effect yang sama seperti AES-128.[10]

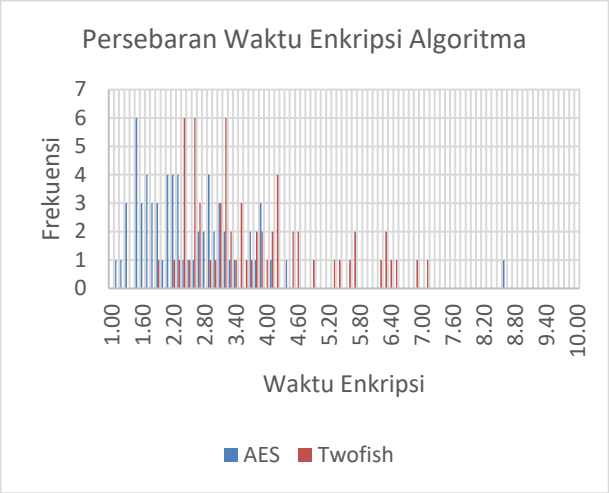
3. Penerapan Konsep Matriks *Involuntary* pada Kotak S

Matriks *involuntary* adalah matriks yang memiliki invers yang sama dengan matriks itu sendiri. Penerapan konsep ini adalah agar meminimalkan terjadinya *timing attack* yang merupakan *side channel attack*, dimana serangan terjadi apabila waktu enkripsi dan dekripsi memiliki perbedaan waktu yang signifikan.

Misalnya pada AES-128 plainteks bebas, penerapan matriks *involuntary* dapat menyebabkan perubahan nilai *Avalanche Effect* pada saat pengujian SAC sebesar 0.088% hingga 0.121% pada setiap iterasi. Sedangkan pada AES-128 kunci bebas, perubahan nilainya sebesar 0.0845% hingga 0.1215%.[11]

E-ISSN :2623-1700
P-ISSN : 1693-3672

Untuk performa enkripsi data NIK, kedua algoritma menghasilkan hasil sebaran nilai waktu enkripsi sebagai berikut;

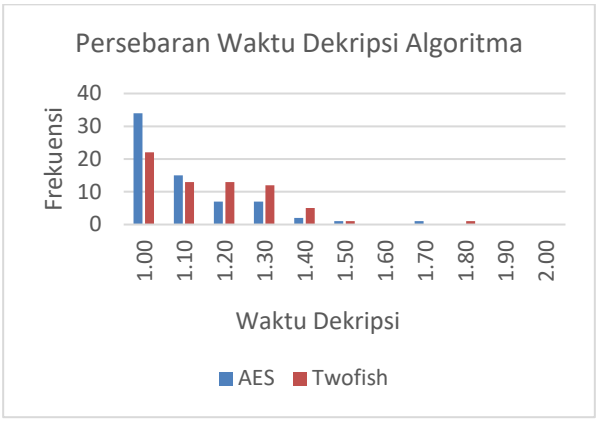


Gambar 3.2 Grafik Persebaran Kecepatan Enkripsi Algoritma

Dapat dilihat pada grafik diatas, waktu enkripsi algoritma AES berkisar antara 1.1 ms hingga 4.4 ms, dan hanya menyisakan satu data dimana waktu enkripsinya adalah 8.6 ms.

Sedangkan pada Twofish, waktu enkripsinya berkisar antara 1.9 ms hingga 7.10 ms. Sehingga dapat disimpulkan bahwa kecepatan enkripsi teks algoritma AES lebih cepat dibandingkan enkripsi teks algoritma Twofish.

Untuk performa dekripsi data NIK, kedua algoritma ini tidak memiliki banyak perbedaan dibanding performa enkripsi data NIK.



Berdasarkan grafik diatas, waktu enkripsi algoritma AES berkisar antara 1 ms hingga 1.7 ms, sedangkan Twofish berkisar antara 1 ms hingga 1.8 ms.

Ada dua faktor utama yang dapat mempengaruhi performa kecepatan enkripsi dan dekripsi dari suatu algoritma, yaitu;

1. **Ukuran data masukan/keluaran**

   Pada penelitian ini, ukuran data yang digunakan dalam operasi enkripsi dan dekripsi adalah sebesar 16 byte (128 bit), sehingga waktu enkripsi dan dekripsi relatif rendah.

   Sedangkan pada data yang berukuran lebih besar, memiliki waktu enkripsi dan dekripsi yang lebih lama.[12][13]

2. **Mode operasi**

   Operasi diatas menggunakan mode ECB yang mana proses enkripsi dan dekripsi dapat dilakukan secara paralel. Apabila proses enkripsi dan dekripsi menggunakan mode CBC, maka proses enkripsi relatif jauh lebih lama daripada proses dekripsi dikarenakan enkripsi pada mode CBC dilakukan secara sekuensial.[14]

KESIMPULAN

Algoritma AES dalam mengenkripsi data NIK memiliki rata-rata performa yang lebih bagus dari segi kecepatan enkripsi data sebesar 1.34 ms dari algoritma Twofish. Sedangkan untuk mendekripsi data NIK, algoritma AES memiliki rata-rata performa yang tidak jauh beda dengan algoritma Twofish yaitu sebesar 0.06 ms dibandingkan algoritma Twofish. Perbedaan kecepatan pada masing-masing proses enkripsi dan proses dekripsi dalam suatu algoritma disebabkan karena adanya perbedaan ukuran data masukan yang dienkripsi, atau data keluaran yang didekripsi. Selain itu, mode operasi yang digunakan dalam proses enkripsi dan dekripsi dari suatu algoritma juga dapat mempengaruhi kecepatan dari enkripsi dan dekripsi, hal ini dikarenakan ada beberapa mode operasi yang dapat melakukan proses enkripsi atau dekripsi secara paralel, dan ada juga yang tidak.

Dalam hal kriteria SAC, kedua algoritma ini sama-sama memenuhi kriteria SAC. Dimana dalam pengujian ini, perubahan satu bit di akhir plainteks, dapat menghasilkan rata-rata nilai *Avalanche Effect* yang sesuai dengan kriteria SAC, yaitu nilai perubahan sebesar 50% dari panjang bit. Perbedaan nilai *Avalanche Effect* dari sebuah algoritma dipengaruhi oleh tiga faktor, yaitu jumlah operasi algoritma, jumlah iterasi yang terdapat dalam algoritma, dan penerapan matriks *involuntary* pada properti kotak S pada suatu algoritma.

Sehingga dapat disimpulkan bahwa algoritma AES sangat cocok untuk mengamankan data NIK karena memiliki performa kecepatan enkripsi yang lebih baik dari Twofish, walaupun performa kecepatan dekripsi yang tidak terlalu berbeda. Selain itu dalam penelitian ini, algoritma AES hanya memiliki *margin of error* sebesar 0.27 untuk memenuhi kriteria SAC, sedangkan Twofish memiliki 1.03.

DAFTAR PUSTAKA

[1] Diambil dari surat kabar berita Kompas https://tekno.kompas.com/read/2021/01/01/14260027/7-kasus-kebocoran-data-yang-terjadi-sepanjang-2020

[2] Diambil dari surat kabar berita Kompas https://www.kompas.com/tren/read/2022/01/08/163000065/sederet-kasus-kebocoran-data-penduduk-di-server-pemerintah

[3] Merliana, Ni Putu Eka (2020). *Pemanfaatan Teknologi Kriptografi dalam Mengatasi Kejahatan Cyber*. Vol. 3 No. 2. ISSN 2548

[4] Horst Feistel (1973). *Cryptography theory of secrecy system*. Bell System Technical Journal. Vol 28(4), pp. 656-715

[5] Hercigonja, Z., Gimnazija, D., & Varazdin, C. (2016). *Comparative analysis of cryptographic algorithms and advanced cryptographic*. International Journal of Digital Technology & Economy. Vol. 1 No. 2 pp. 1-8

[6] Ramanujam, Sriram., Karuppiah, Marimuthu. (2011). *Designing an algorithm with high Avalanche Effect*. IJCSNS International Journal of Computer Science and Network Security. Vol.11 No. 1

[7] Conrad, Eric (2016). *Security Engineering CISSP study guide*. Elsevier

[8] NIST (2017). *Advanced Encryption Standard Algorithm Validation List*. Retrieved from http://csrc.nist.gov

[9] Sarita D. Sanap (2021). *Performance Analysis of Encryption Technique Based on Avalanche Effect and Strict Avalanche Criterion*. 3rd International Conference on Signal Processing and Communication

[10] Novita Angraini, M. Wibisono, Nugroho Jati. (2018). *Pengaruh Komponen Algoritma AES terhadap Hasil Uji SAC dari Algoritma AES*. Seminar Nasional Teknologi Informasi dan Multimedia 2018. ISSN: 2302-3805

[11] Novita Angraini. (2018). *Pengaruh Implementasi Matriks Involuntray terhadap Hasil Uji SAC Algoritma AES*. Seminar Nasional Teknologi Informasi dan Komunikasi X Palembang-Indonesia.

[12] Tyagi, Pronika S. S. (2021). *Performance analysis of encryption and decryption algorithm*. Indonesian Journal of Electrical Engineering and Computer Science. Vol. 23 No. 2 pp. 1030-1038. ISSN: 2502-4752. DOI: 10.11591/ijeecs.v23.i2.pp1030-1038

[13] Meko, Donizilio Antonio. (2018). *Perbandingan Algoritma DES, AES, IDEA, dan Blowfish dalam Enkripsi dan Dekripsi Data*. Jurnal Teknologi Terpadu Vol. 4 No.1. ISSN: 2477-0043

[14] Diambil dari NIST Special Publication 800-38A 2001 Edition https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf