

RANCANG BANGUN PROGRAM KRIPTOGRAFI ADVANCED ENCRYPTION STANDARD

Sumi Khairani¹
Fherly Agustin²
Ananda Fahmi³

sumi@potensi-utama.ac.id, fherly@potensi-utama.ac.id, fahmi@potensi-utama.ac.id

ABSTRAKSI

Untuk menjamin keamanan dan keutuhan dari suatu data, dibutuhkan suatu proses penyandian. Enkripsi dilakukan ketika data akan dikirim. Proses ini akan mengubah suatu data asal menjadi data rahasia yang tidak dapat dibaca. Sementara itu, proses dekripsi dilakukan oleh penerima data yang dikirim tersebut. Data rahasia yang diterima akan diubah menjadi data asal. Dengan cara penyandian, data asli tidak akan terbaca oleh pihak yang tidak berkepentingan, melainkan hanya oleh penerima yang memiliki kunci dekripsi. AES digunakan sebagai standard algoritma kriptografi yang terbaru. Algoritma sebelumnya dianggap tidak mampu lagi untuk menjawab tantangan perkembangan teknologi komunikasi yang sangat cepat. AES sendiri adalah algoritma kriptografi dengan menggunakan algoritma Rijndael yang dapat mengenkripsikan block data sepanjang 128 bit dengan panjang kunci 128 bit, 192 bit dan 256 bit.

Kata kunci : AES, Algoritma Rijndael, Dekripsi, Enkripsi, Kriptografi

PENDAHULUAN

Kriptografi berbasis pada algoritma pengkodean data informasi yang mendukung kebutuhan dari dua aspek keamanan informasi, yaitu *secrecy* (perlindungan terhadap kerahasiaan data informasi) dan *authenticity* (perlindungan terhadap pemalsuan dan pengubahan informasi yang tidak diinginkan).

Keamanan dan kerahasiaan data merupakan salah satu aspek yang sangat penting pada sistem informasi saat ini. Hal ini disebabkan pesatnya perkembangan ilmu pengetahuan dan teknologi yang memungkinkan munculnya suatu teknik-teknik yang baru yang disalahgunakan oleh pihak-pihak tertentu yang mengancam keamanan dari sistem informasi tersebut. Ironisnya, teknik yang digunakan untuk mengancam keamanan data selalu setingkat lebih maju daripada teknik yang digunakan untuk mengamankan

-
1. **Dosen Program Studi Teknik Informatika, STMIK Potensi Utama**
Jl. K.L. Yos Sudarso Km. 6,5 No. 3A Medan, Telp. (061) 6640525
 2. **Dosen Program Studi Sistem Informasi, STMIK Potensi Utama**
Jl. K.L. Yos Sudarso Km. 6,5 No. 3A Medan, Telp. (061) 6640525
 3. **Mahasiswa Program Studi Teknik Informatika, STMIK Potensi Utama**
Jl. K.L. Yos Sudarso Km. 6,5 No. 3A Medan, Telp. (061) 6640525

data. Karena itu timbul suatu gagasan yang mengacu kepada permasalahan tersebut, yakni untuk membuat suatu sistem keamanan yang dapat melindungi data yang dianggap penting dengan cara menyandikan data sehingga sulit untuk dideteksi oleh pihak yang tidak berhak. Untuk keperluan tersebut, maka diperlukan teknik kriptografi dengan metode enkripsi dan dekripsi.

Salah satu faktor yang menyebabkan suatu teknik kriptografi menjadi lebih sering digunakan adalah tingkat kerahasiaan untuk mengamankan data yang tinggi disertai dengan kemudahan penggunaannya.

ANALISA

AES didesain berdasarkan *wide trail strategy* yang dicetuskan pendesain Rijndael dan Daemen, dalam disertasinya. Strategi ini mengusulkan agar cipher terdiri dari tiga komponen utama yaitu pencampuran kunci, transformasi tidak linear dan transformasi linear. Pencampuran kunci bertujuan agar keamanan algoritma tidak terletak pada dirahasiakannya algoritma, melainkan pada kerahasiaan kunci. Transformasi nonlinear bertujuan agar bila diketahui keluaran, maka tidak dapat diketahui masukannya. Hal ini dapat dilakukan dengan *S-Box*. Transformasi linear bertujuan agar sebanyak mungkin transformasi nonlinear yang aktif. Dengan memisahkan transformasi linear dengan non linear, diharapkan kita dapat mendesain transformasi non linear terbebas dari transformasi linear dan sebaliknya. Daemen menekankan betapa perlunya desain transformasi linear yang baik. Analisis sandi pada DES menunjukkan betapa buruknya transformasi linear pada DES (permutasi) sehingga tidak dapat menahan analisis sandi linear (ASL) dan analisis sandi diferensial (ASD). Pada awal dan akhir cipher, diberikan operasi *Pre-whitening* dan *post-whitening*, yang berupa XOR plaintext /ciphertext dengan subkey. Operasi ini bertujuan meningkatkan keamanan seperti halnya pada DES-X. Sedangkan pada awal dan akhir DES hanya terdapat permutasi yang tidak meningkatkan keamanan.

Kotak-S (*S-Box*)

Kotak-S pada AES didesain dengan rumusan matematika untuk menghilangkan kecurigaan akan ditanamnya backdoor pada kotak-S. Penggunaan inversi x^{-1} pada GF(28) dikarenakan ketahanan operasi ini terhadap analisis sandi linear dan diferensial. Maksimum peluang propagasi diferensialnya $DP_{maks} = 2^{-6}$ dan korelasi linear maksimumnya $LP_{maks} = 2^{-3}$. Meskipun inversi ini sudah mengamankan AES dari analisis sandi diferensial (ASD) dan linear (ASL), namun karena kesederhanaannya, maka dikuatirkan kotak-S ini mudah diserang dengan *algebra attack*, khususnya *interpolation attack*. Karena itulah ditambahkan transformasi linear yang memiliki sifat tidak mempengaruhi ketahanan terhadap ASD dan ASL, namun menghilangkan kesempatan

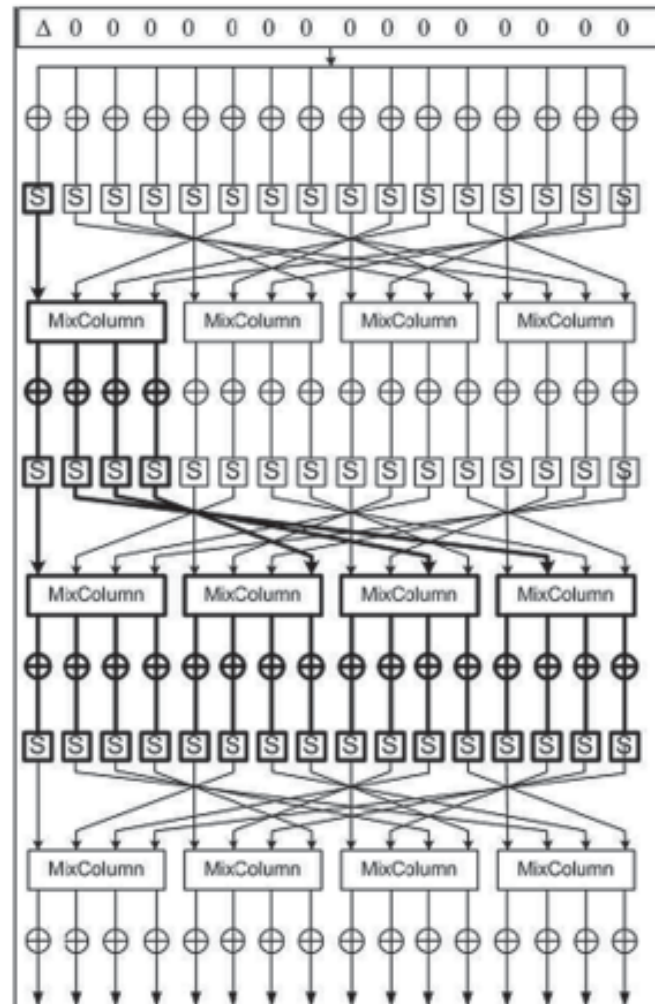
penyerang untuk mengeksploitasi kesederhanaan aljabar AES. Karena itu dikalikanlah dengan matrik L dalam GF(2). Ketidakkompatibelan operasi antara GF(28) dan GF(2) ini mempersulit serangan aljabar terhadap AES. Kemudian, untuk menghindari pemetaan 0 ke 0, maka ditambahkan konstanta c. Pemetaan 0 ke 0 dalam kotak-S pada DES membuat DES rentan terhadap serangan ASD dan ASL. Dalam ASD, pemetaan semacam ini memudahkan penyerang mendapatkan karakteristik iteratif sehingga DES dapat dipecahkan. Kotak-S juga didesain agar memungkinkan operasi berjalan secara paralel penuh. Pada setiap ronde, ke-16 kotak-S AES dapat dioperasikan bersamaan, khususnya pada perangkat keras, sehingga operasi dapat berjalan dengan sangat cepat. AES hanya memiliki satu macam kotak-S (*S-Box*), sehingga menghemat jumlah gerbang yang diperlukan pada perangkat keras. Kotak-S ini juga invertible sehingga inversinya dapat digunakan pada proses dekripsi.

Berikut ini adalah ringkasan kriteria kotak-S :

- Invertibility (untuk dekripsi)
- Minimisasi korelasi antara kombinasi linear bit-bit masukan dan kombinasi linear bit-bit keluaran (menahan ASL)
- Minimisasi nilai terbesar pada tabel XOR (menahan ASD)
- Kompleksitas ekspresi aljabar pada GF(28)
- Kesederhanaan deskripsi (mudah analisisnya)

Operasi Mixcolumns

Bersama dengan operasi *ShiftRows*, *MixColumn* merupakan transformasi linear yang bertujuan untuk menyebarkan pengaruh transformasi nonlinear ke sebanyak mungkin komponen nonlinear di ronde selanjutnya. Bila *ShiftRows* bertujuan menyebarkan pada arah baris, maka *MixColumn* bertujuan menyebarkan ke arah kolom. Dengan perpaduan dua operasi ini, diperoleh difusi yang sangat baik. Dalam *MixColumn* diperkenalkan konsep jumlah cabang (yang dicetuskan Daemen dalam disertasinya) $\hat{a}$ untuk matrik M. Bila jumlah koefisien tidak-nol dalam vektor a dinyatakan dengan $wb(a)$, maka untuk $a \circ b = \min \{ wb(a \hat{\circ} b) + wb(Ma \hat{\circ} Mb) \}$ Matrik *MixColumn* M memiliki $\hat{a} = 5$. Artinya bahwa beda tidak nol dalam satu byte akan disebarkan paling sedikit ke beda tidak nol dalam empat byte, seperti terlihat pada gambar 1, terlihat bahwa ASD dan ASL untuk 4 ronde ke atas akan melibatkan sedikitnya 21 kotak-S aktif, sehingga $DP_{maks} = (2-6)21 = 2-126$ dan $LP_{maks} = 2(21-1)(2-3)21 = 2202-63 = 2-43$.



Gambar 1. Analisis Sandi Diferensial dan Linier AES

Ini mengindikasikan bahwa jumlah pasang plaintext yang diperlukan untuk ASD minimal sebanyak 2126 dan untuk ASL sebanyak $(\frac{1}{2}-43)2 = 286$. Sedangkan jumlah plaintext yang mungkin sebanyak 2^{128} . Artinya, 4 ronde AES akan kebal terhadap ASD dan ASL, sedangkan telah diketahui bahwa jumlah minimal ronde AES adalah 10 ronde. Bila diperhitungkan pula *differential* dan *linear hull* diperkirakan AES 5 ronde akan mampu menghadapinya.

Kriteria Ekspansi Kunci

Subkey pada tiap ronde dapat diperoleh dari rumus : $W[i] = W[i-6] \oplus W[i-1]$, $W[6i] = W[6i-6] \oplus f(W[6i-1])$ di mana $f()$ merupakan fungsi penggunaan kotak-S (*S-Box*) dan penambahan konstanta ronde. Penjadwalan kunci dapat diimplementasikan tanpa eksplisit menggunakan array W . Jika jumlah yang tersedia kecil, maka kunci per ronde dapat dihitung *on-the-fly* dan hanya membutuhkan buffer sebesar 64 byte. Rekursi ekspansi kunci bersifat invertible. Ini berarti bahwa mengetahui 4 word (64 byte) berurutan dari kunci terekspansi akan dapat membangkitkan seluruh tabel subkey. Meskipun demikian, mengetahui sebagian bit subkey (kurang dari 4 word) tidak akan mengijinkan mengetahui bit-bit subkey atau *key* yang lain. Ekspansi kunci sederhana dan efisien untuk banyak prosesor. Konstanta per ronde menghilangkan sifat simetri. Substitusi pada ekspansi kunci memberikan ketidaklinearan sehingga dapat menghindari *related-key attack*

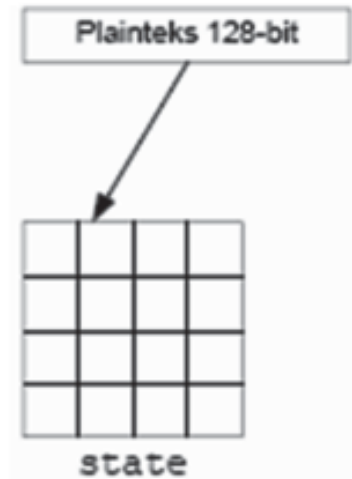
Analisis Desain AES

Pengelompokkan jenis AES ini adalah berdasarkan panjang kunci yang digunakan. Angka-angka di belakang kata AES menggambarkan panjang kunci yang digunakan pada tiap AES. Selain itu, hal yang membedakan dari masing-masing AES ini adalah banyaknya round yang dipakai. AES-128 menggunakan 10 round, AES-192 sebanyak 12 round, dan AES-256 sebanyak 14 round. AES memiliki ukuran block yang tetap sepanjang 128 bit dan ukuran kunci sepanjang 128, 192, atau 256 bit.

Algoritma Rijndael mempunyai 3 parameter yaitu

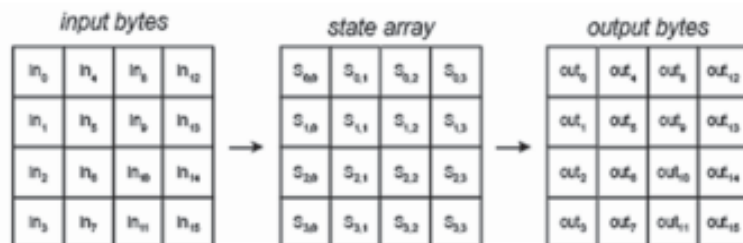
1. Plaintext : *array* yang berukuran 16-byte, yang berisi data masukan.
2. Ciphertext : *array* yang berukuran 16-byte, yang berisi hasil enkripsi.
3. Key : *array* yang berukuran 16-byte, yang berisi kunci ciphering (disebut juga *cipher key*).

Dengan 16 *byte*, maka baik blok data dan kunci yang berukuran 128-bit dapat disimpan di dalam ketiga *array* tersebut ($128 = 16 \times 8$). Selama kalkulasi plainteks menjadi ciphertexts, status sekarang dari data disimpan di dalam *array of bytes* dua dimensi, state, yang berukuran $NROWS \times NCOLS$. Untuk blok data 128-bit, ukuran state adalah 4×4 . Elemen *array* state diacu sebagai $S[r,c]$, dengan $0 \leq r < 4$ dan $0 \leq c < Nb$ (Nb adalah panjang blok dibagi 32. Pada AES-128, $Nb = 128/32 = 4$).



Gambar 2. State Awal

Pada awal enkripsi, 16-byte data masukan, $in_0, in_1, \dots, in_{15}$ disalin ke dalam *array* state (direalisasikan oleh fungsi `CopyPlaintextToState(state, plaintext)`) seperti diilustrasikan sebagai berikut:



Gambar 3. Alur State Array

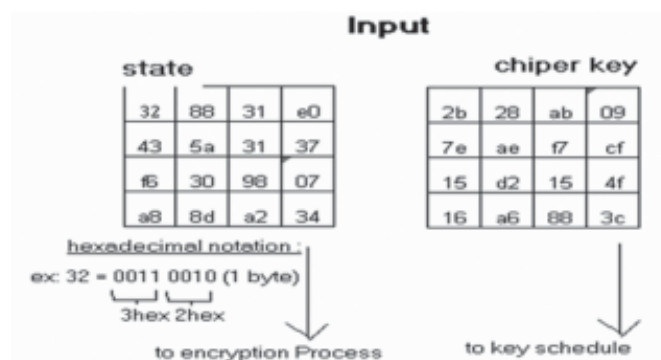
Operasi enkripsi/dekripsi dilakukan terhadap *array* S , dan keluarannya ditampung didalam *array* out . Skema penyalinan *array* masukan in ke *array* S : $S[r, c] \leftarrow in[r + 4c]$ untuk $0 \leq r < 4$ dan $0 \leq c < Nb$. Skema penyalinan *array* S ke *array* keluaran out : $out[r+4c] \leftarrow S[r, c]$ untuk $0 \leq r < 4$ dan $0 \leq c < Nb$.

Berdasarkan ukuran block yang tetap, AES bekerja pada matriks berukuran 4x4 di mana tiap-tiap sel matriks terdiri atas 1 byte (8 bit). Berikut adalah pemahaman algoritma enkripsi AES-128 menurut Rinaldi Munir (2004 : 5), berdasarkan garis besar Algoritma *Rijndael* yang beroperasi pada blok 128-bit dengan kunci 128-bit adalah sebagai berikut:

1. Siapkan array berukuran 4x4 bernama kunci.
2. Siapkan array berukuran 4x4 bernama state
3. Cetak : "Masukkan 16 bilangan heksadesimal sebagai kunci."
4. Simpan 16 nilai tersebut sebagai nilai dari masing-masing elemen array kunci.
5. Cetak : "Masukkan teks yang akan dienkripsikan."
6. Konversikan teks tersebut kedalam bentuk bit menggunakan kode ASCII.
7. Konversikan kode ASCII tersebut dalam heksadesimal.
8. Kelompokkan bit-bit teks tersebut menjadi 128 bit tiap bagiannya.
9. Ambil 128 bit pertama untuk diproses.
10. Kelompokkan bit teks tersebut menjadi 16 bagian dengan 8 bit tiap bagiannya.
11. Masukkan tiap-tiap bagian eks tersebut ke dalam tiap-tiap sel pada matriks berukuran 4x4.
12. konversikan bit kedalam heksadesimal.
13. Lakukan langkah AddRoundKey
14. Lakukan langkah Sub Bytes
15. Lakukan langkah ShiftRows
16. Lakukan langkah MixColumns
17. Lakukan langkah AddRoundKey
18. Ulangi langkah 13 -16 sebanyak 9 kali.
19. Jika langkah 17 sudah dilakukan, maka lakukan langkah SubBytes
20. Lakukan langkah ShiftRows
21. Lakukan langkah AddRoundKey.
22. Selesai.

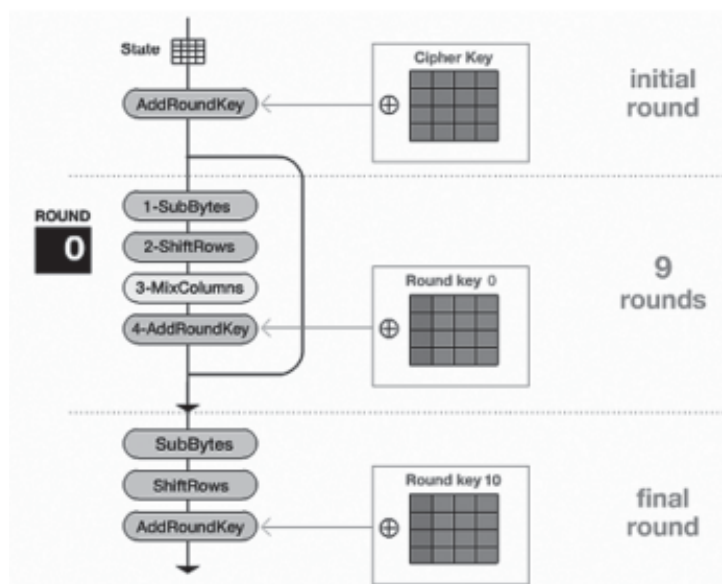
Penjelasannya :

Berikut ini adalah contoh penerapan proses enkripsi AES lihat. Pada rounds pertama *plaintext* dan *key* yang bernotasi *hexadecimal* di XOR adala sebagai berikut:



Gambar 4. Elemen state dan kunci dalam notasi HEX

Misal, teks sudah diolah sedemikian rupa sehingga matriks (array) berukuran 4 x 4 dan begitu pula kata kunci.



Gambar 5. Diagram Proses Enkripsi

Kemudian akan dilakukan 4 transformasi sebagai berikut sebanyak 9 kali :

1. SubBytes.
2. ShiftRows.
3. MixColumns.
4. AddRoundKey

Setelah itu, untuk round ke 10 dilakukan 3 transformasi sebagai berikut :

1. SubBytes.
2. ShiftRows.
3. AddRoundKey.

Berikut adalah penjelasan dari Proses SubBytes, ShiftRows, MixColumns dan AddRoundKey adalah :

1. Transformasi *SubBytes*

Transformasi *SubBytes* memetakan setiap byte dari *array state* dengan menggunakan tabel substitusi *S-box*. *AES* hanya mempunyai satu buah *S-box*. Tabel *S-box* yang digunakan adalah:

Tabel III.1 Substitusi (*S-Box*)

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

Gambar 6. Transformasi SubBytes

Cara pensubstitusian adalah sebagai berikut: untuk setiap byte pada *array state*, misalkan $S[r, c] = xy$, yang dalam hal ini xy adalah digit heksadesimal dari nilai $S[r, c]$, maka nilai substitusinya, dinyatakan dengan $S'[r, c]$, adalah elemen di dalam *S-box* yang merupakan perpotongan baris x dengan kolom y . Misalnya $S[0, 0] = 19$, maka $S'[0, 0] = d4$.

2. Transformasi *ShiftRows*

ShiftRows bertujuan menyebarkan pada arah baris Transformasi *ShiftRows* melakukan pergeseran secara *wrapping* (siklik) pada 3 baris terakhir dari *array state*. Jumlah pergeseran bergantung pada nilai baris (*r*). Baris $r = 1$ digeser sejauh 1 *byte*, baris $r = 2$ digeser sejauh 2 *byte*, dan baris $r = 3$ digeser sejauh 3 *byte*. Baris $r = 0$ tidak digeser.

1. Transformasi *ShiftRows()* melakukan pergeseran secara *wrapping* (siklik) pada 3 baris terakhir dari *array state*.
2. Jumlah pergeseran bergantung pada nilai baris (*r*). Baris $r = 1$ digeser sejauh 1 *byte*, baris $r = 2$ digeser sejauh 2 *byte*, dan baris $r = 3$ digeser sejauh 3 *byte*. Baris $r = 0$ tidak digeser. (lihat Gambar 6)

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

d4	e0	b8	1e
bf	b4	41	27
11	98	5d	52
ae	f1	e5	30

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
ae	f1	e5	30

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

Gambar 7. State awal ,rotate 1, rotate2 dan rotate 3

3. Transformasi *MixColumn*

MixColumn merupakan transformasi linear yang bertujuan untuk menyebarkan pengaruh transformasi nonlinear ke sebanyak mungkin komponen nonlinear di ronde selanjutnya, dan menyebarkan ke arah kolom.

1. Transformasi *MixColumns()* mengalikan setiap kolom dari *array state* dengan polinom $a(x) \bmod (x^4 + 1)$.
2. Setiap kolom diperlakukan sebagai polinom 4-suku pada $GF(28)$.
3. $a(x)$ yang ditetapkan adalah:

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

Transformasi ini dinyatakan sebagai perkalian matriks:

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

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

$$s'_{0,x} = (\{02\} \bullet s_{0,x}) \oplus (\{03\} \bullet s_{1,x}) \oplus s_{2,x} \oplus s_{3,x}$$

$$s'_{1,x} = s_{0,x} \oplus (\{02\} \bullet s_{1,x}) \oplus (\{03\} \bullet s_{2,x}) \oplus s_{3,x}$$

$$s'_{2,x} = s_{0,x} \oplus s_{1,x} \oplus (\{02\} \bullet s_{1,x}) \oplus (\{03\} \bullet s_{3,x})$$

$$s'_{3,x} = (\{03\} \bullet s_{0,x}) \oplus s_{0,x} \oplus s_{1,x} \oplus (\{02\} \bullet s_{3,x})$$

e0	b8	1e
b4	41	27
52	11	98
ae	f1	e5

d4
bf
5d
30

 $=$

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

 $=$

04
66
81
e5

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

Gambar 8. Proses mix columns

4. Transformasi AddRoundKey

Pencampuran kunci (*AddRoundKey*) bertujuan agar keamanan algoritma tidak terletak pada dirahasiakannya algoritma, melainkan pada kerahasiaan kunci. Transformasi ini melakukan operasi XOR terhadap sebuah *round key* dengan *array state*, dan hasilnya disimpan di *array state*. Contoh AddRoundKey adalah :

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

 $\oplus$

a0	88	23	2a
fa	54	a3	6c
fe	2c	39	76
17	b1	39	05

 $\rightarrow$

a4	68	6b	02
9c	9f	5b	6a
7f	35	ea	50
f2	2b	43	49

Gambar 9. Proses add round keys

Kolom ke-1 mariks state ang telah melalui proses SubBytes dan ShiftRows di – XOR kan dengan kolom ke-1 RoundKey Begitu pula dengan kolom ke-2, ke-3 dan ke-4, kemudian akan dilanjutkan sampai RoundKey 10. Berikut adalah hasil dari AddRoundKey Ke-1 :

a4	68	6b	02
9c	9f	5b	6a
7f	35	ea	50
f2	2b	43	49

Gambar 10. Hasil AddRoundKey ke-1

Berikut ini adalah tampilan *rounds* selanjutnya setelah penjelasan dari round pertama, dimulai dari round ke dua sampai kesepuluh untuk satu buah mariks state, teks 128 bit. Round terakhir tidak melibatkan pengubahan dengan cara mix columns.

	Round 2	Round 3	Round 4	Round 5	Round 6
After SubBytes	49 45 72 77 dc db 39 02 89 f7 1a 3b	ac ef 13 45 73 c1 b5 23 cf 11 d6 5a	52 85 e3 fe 59 a4 11 cf 2f 5e c8 6a	e1 e8 25 97 4f 2b c8 6c 42 2b 96 aa	a1 78 10 40 c3 4f e8 d5 e8 29 3d 93
After ShiftRows	49 45 72 77 db 39 02 4e 87 53 d7 56 3b 89 21 1a	ac ef 13 45 c1 b5 23 73 d6 5a cf 11 b8 7b df b5	52 85 e3 fe a4 11 cf 50 c8 6a 2f 5e 94 28 d7 97	e1 e8 25 97 b5 c8 6c 4f 96 aa d2 2b 7c 9b ba 53	a1 78 10 40 4f e8 d5 93 3d 03 e8 29 2e 2c d2 23
After MixColumns	58 1b db 1b d4 4b e7 8b 0a 5a 0a 30 f1 ac a8 e5	75 29 53 bb ec 00 ec 25 09 63 ef d0 93 33 7c dc	0f 60 6f 5e 29 31 e0 b3 da 38 10 13 e9 bf 4b 91	25 bd b6 4c d1 11 5a 4e a9 d1 33 e0 ad 68 8e b0	4b 2c 33 37 88 4a 9d d2 8d 89 e4 18 6d 88 e8 d0
Round Key	f2 7a 59 73 c2 96 35 50 95 b9 80 c4 f2 43 7a 7f	3d 47 1e 6d 80 16 23 7a 47 2e 7e 88 74 5e 44 3b	ef e8 b6 db 44 52 71 8b e5 5b 25 ad 41 7f 3b 90	d4 7c ea 11 d1 83 f2 f9 c4 9d b8 15 f8 87 bc bc	6d 11 db ea 88 0b f9 90 a3 3e 66 93 7a 66 41 6d
After AddRoundKey	aa 61 82 68 8f d0 d2 32 5f e3 4a 46 03 ef d2 9a	48 67 6d d6 60 1d e3 5f 4e 9d b1 58 ee 0d 38 e7	e0 c8 d9 85 92 63 b1 b8 7f 63 35 be e8 c0 50 01	f1 c1 7c 5d 00 92 e8 b8 4f 4e 8b d5 55 ef 32 0c	26 5d e8 fd 5e 41 64 d2 2e b7 72 8b 17 7d e9 25

	Round 7	Round 8	Round 9	Round 10
After SubBytes	27 27 9b 54 ad 93 42 b5 31 a9 40 3a f0 f2 d3 3d	ba d4 0a da 83 3d e1 64 2c 06 da f2 c8 e0 d4 6e	87 e2 42 97 4a c3 48 e7 8e d8 55 e8 6e 80 d8 95	e9 0b 3d ef 09 31 22 2e 89 07 7d 2c b5 72 5f 94
After ShiftRows	27 27 9b 54 83 43 b5 ab 40 3d 31 e9 32 f0 f2 d3	ba d4 0a da 3b e1 e6 92 d4 f2 2c 64 7c 0e c0 6d	87 e2 42 97 6a 4c 90 ec 46 e7 4a c3 ee 80 d8 95	e9 0b 3d ef 21 32 2e 09 7d 2c 89 07 b5 72 5f 94
After MixColumns	14 46 27 34 15 10 46 2a b5 15 56 d8 bf ee 47 43	00 b1 54 fa 51 c2 7d 1b 2f 60 6d 99 d1 f2 cd ea	47 40 e3 40 37 d4 70 97 94 ea 3a 42 ed e5 e5 be	48 c9 e1 b6 14 ee 2f 83 28 25 8a 8c a8 89 08 a8
Round Key	4e 5f 84 4e 54 5f ee e5 ef e9 4f da 4e 23 b2 42	ee b5 31 72 d2 88 2b 6d 73 be f0 28 21 d2 80 22	ac 19 28 57 77 2a d1 50 64 d8 28 88 23 21 41 6a	d8 c9 e1 b6 14 ee 2f 83 28 25 8a 8c a8 89 08 a8
After AddRoundKey	3a 19 a3 7a 41 49 e0 8c 42 de 19 04 51 1f 65 0e	aa 04 65 85 e3 45 5d 96 5e 33 98 b0 20 3d cd e5	eb 98 8d 10 40 26 a1 c3 22 58 13 43 1e 64 e7 d3	39 02 d0 19 25 d0 11 6a 84 08 65 0b 1d 2b 07 33

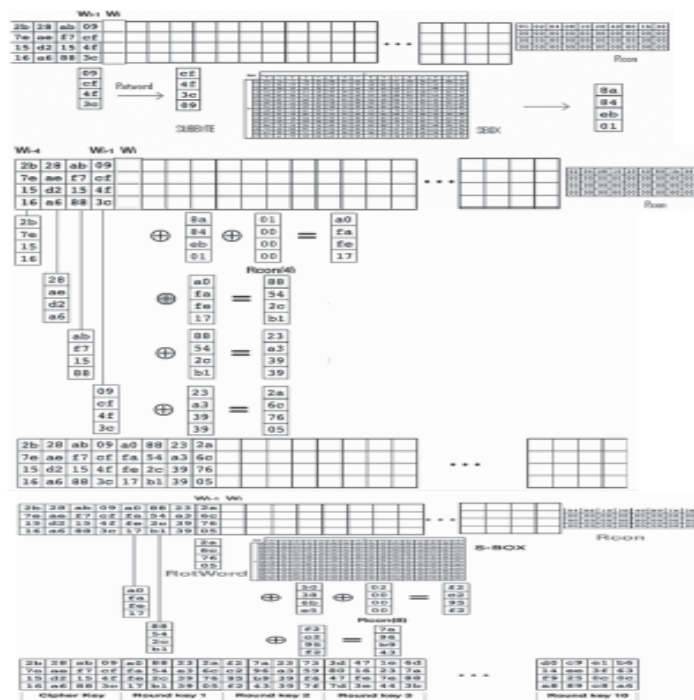
Gambar 11. Proses round kedua sampai kesepuluh

5. Ekspansi Kunci

Algoritma AES melaksanakan *cipher key* dan membuat suatu ekspansi kunci untuk menghasilkan suatu *key schedule*. Ekspansi kunci yang diperlukan AES Nb(Nr+1), word, sehingga bias digunakan AES 128 bit maka, $4(10+1) = 40 \text{ word} = 40 \times 32 \text{ bit} = 1408 \text{ bit}$ sub key. Ekspansi dari 128 menjadi 1408 bit sub key. Proses ini disebut dengan key schedule. Subkey ini diperlukan karena setiap round merupakan suatu inisial dari Nb word untuk Nr = 0 dan 2 Nb untuk Nr = 1,3 untuk Nr = 2,....., yang berisi array linier empat byte word (Wi), $0 \leq i < Nb(Nr+1)$. Contoh :

- RotWord () mengambil input empat-byte word [a0, a1,a2, a3]dan membentuk cyclic permutasi seperti [a1,a2,a,3,a0].
- Sub word () mengambil input empat-byte word dan menggunakan S-Box sehingga didapat empat byte procedure output word.
- Rcon () menghasilkan round yang tetap dari word array dan berisi nilai yang diberikan oleh [xid1,{00},{00},{00}] dengan xid1 dari i ke 1.

Pada AES memiliki sistem penjadwalan kunci sehingga pada setiap perputaran atau rounds kunci selalu berubah- ubah, menurut Rinaldi Munir (2004 : 16) dalam www.ilmukomputer.org seperti pada gambar 12. Penjadwalan kunci sebagai berikut :



Gambar 12. Penjadwalan kunci

Dalam algoritma AES ukuran teks yang dapat dienkripsikan senilai 8192 karakter. Karena dalam AES 128 memiliki panjang kunci (*Nr word*) = 4. dengan catatan 1 *word* = 32 bit atau 4 *word* = 128 bit, dan maksimal block adalah 512 block. Maka 512 block x 128 bit = 65536 bit. Sehingga dapat dituliskan 8192 karakter = 8192 byte = 65536 bit, maka dalam batasan masalah algoritma program untuk ukuran teks maka hanya dapat memproses atau merahasiakan data sebesar 8192 karakter. Berikut adalah penggalan program untuk membatasi ukuran teks adalah sebagai berikut :

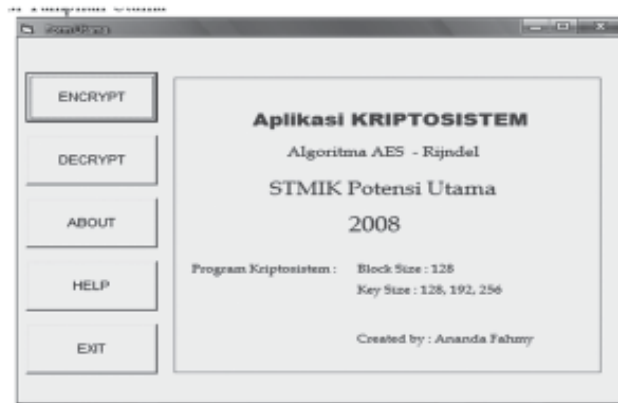
```
Private Sub DisplayString(TheTextBox As TextBox, ByVal TheString
As String)
    If Len(TheString) < 65536 Then
        TheTextBox.Text = TheString
    Else
        MsgBox "Can not assign a String larger than 64k " &
vbCrLf & _
            "to the Text property of a TextBox
control." & vbCrLf & _
            "If you need to support Strings longer
than 64k," & vbCrLf & _
            "you can use a RichTextBox control
instead.", vbInformation
        MsgBox "File Size Error - ciphertext file not a
multiple of block size"
```

Hal yang membedakan dari masing-masing AES ini adalah banyaknya round yang dipakai. AES-128 menggunakan 10 round, AES-192 sebanyak 12 round, dan AES-256 sebanyak 14 round. AES memiliki ukuran block yang tetap sepanjang 128 bit dan ukuran kunci sepanjang 128, 192, atau 256 bit.

IMPLEMENTASI

Aplikasi yang telah selesai dirancang selanjutnya akan diteruskan kepada tahap pengimplementasian. Pada tahap ini difokuskan kepada penerapan aplikasi yang telah diciptakan dengan bahasa pemrograman yang sesuai, sehingga pada akhirnya diperoleh hasil yang sesuai dengan yang diinginkan. Adapun implementasi dari hasil perancangan program adalah sebagai berikut:

1. Modul Tampilan Utama



Gambar 13. Tampilan Utama Program

2. Modul Enkripsi

Modul enkripsi yang berfungsi sebagai pengacak data atau informasi sehingga menjadi bentuk yang tidak dapat terbaca oleh orang lain dan menjadi suatu data yang rahasia. Fungsi modul enkripsi pada program ini berjalan sesuai dengan spesifikasi rancangan. Tampilan modul enkripsi dapat dilihat pada gambar dibawah ini.

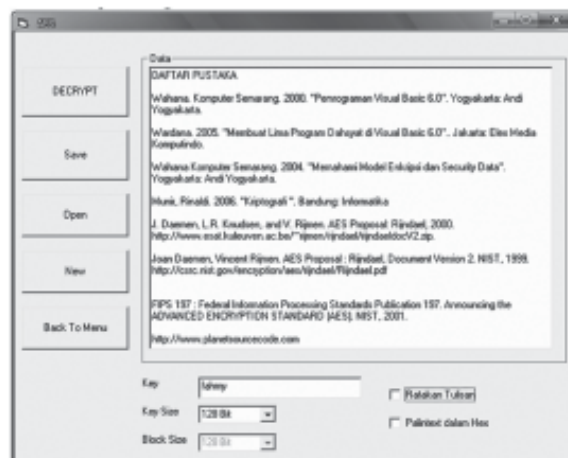


Gambar 14. Modul Enkripsi

2. Modul Dekripsi

Modul Dekripsi yang berfungsi untuk menerjemahkan pesan yang telah diacak sehingga dapat dibaca oleh si penerima pesan atau oleh pihak – pihak yang berhak menerima

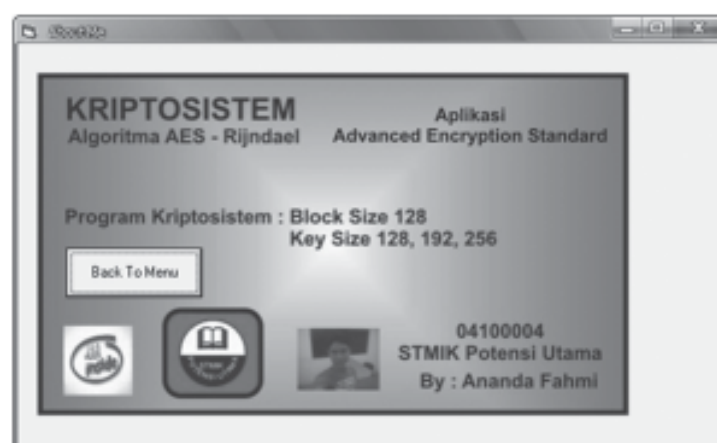
data tersebut. Fungsi modul dekripsi pada program ini berjalan sesuai dengan spesifikasi rancangan. Tampilan modul dekripsi sebagai berikut :



Gambar 15. Modul Dekripsi

4. Modul About

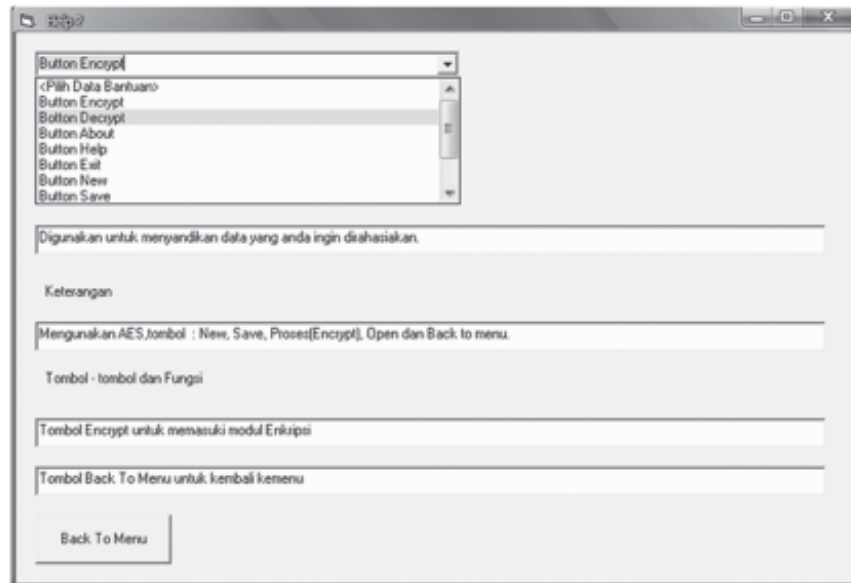
Modul ini menampilkan penjelasan singkat mengenai program aplikasi dalam pembuatan program aplikasi ini. Fungsi modul About pada program aplikasi ini berjalan sesuai dengan spesifikasi rancangan. Modul ini berjalan dengan baik, dapat dilihat pada gambar dibawah ini :



Gambar 16. Modul About

5. Modul Help.

Modul Help dapat digunakan sebagai panduan untuk menjalankan program aplikasi ini. Fungsi modul ini berjalan sesuai dengan spesifikasi rancangan. Modul ini berjalan dengan baik, dapat dilihat pada gambar 17.



Gambar 17. Modul Help

KESIMPULAN

Dari hasil perancangan dan pembuatan program aplikasi kriptosistem menggunakan algoritma Advanced Encryption Standard (AES) ini, dapat diambil kesimpulan sebagai berikut :

1. Program aplikasi kriptosistem ini menjaga kerahasiaan data atau informasi yang ada dalam sebuah komputer dan akan membatasi orang yang tidak berhak atas data yang dimiliki oleh si-pengirim untuk dibaca karena data sudah dienkripsi.
2. Kriptografi *Advanced Encryption Standard* (AES) adalah seni dan ilmu menyembunyikan data dengan menggunakan algoritma AES yang memiliki panjang kunci 128 bit, 192 bit dan 256 bit.
3. Program aplikasi kriptosistem ini dapat mengenkripsikan dan mendekripsikan data yang berupa text yang tersedia pada keyboard, bukan suara maupun gambar.

DAFTAR PUSTAKA

1. Ariyus Dony, 2006, *Kriptografi Keamanan Data Dan Komunikasi*, Graha Ilmu, Yogyakarta.
2. Fahmi Ananda, 2008, *Perancangan Aplikasi Kriptography Advanced Encryption Standard*, Penelitian Tugas Akhir Mahasiswa Strata 1 Program Studi Teknik Informatika, STMIK Potensi Utama, Medan
3. J. Daemen, L.R. Knudsen, and V. Rijmen. AES Proposal: Rijndael, 2000, Diakses pada tanggal 15 Januari 2008 dari <http://www.esat.kuleuven.ac.be/rijmen/rijndael/rijndaeldocV2.zip>.
4. Joan Daemen, Vincent Rijmen. *AES Proposal : Rijndael, Document Version 2*. NIST, 1999, Diakses pada tanggal 15 Januari 2008 dari <http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf>
5. Munir Rinaldi, 2006. **"Kriptografi"**. Bandung, Informatika
6. Wahana. Komputer Semarang. 2000. **"Pemrograman Visual Basic 6.0"**, Penerbit Andi Yogyakarta.
7. Wardana. 2005. **"Membuat Lima Program Dahsyat di Visual Basic 6.0"**, Elex Media Komputindo, Jakarta.