Analisis Perbandingan Antara Algoritma Rijndael Dan Algoritma Twofish Dalam Penyandian Teks

Izzat Shulhan
STMIK Budi Darma Medan, Medan - Indonesia
E-Mail : zathunt@gmail.com

ABSTRAK

Kemudahan pengaksesan media komunikasi dengan memanfaatkan kemajuan teknologi informasi tentunya akan memberikan dampak bagi keamanan informasi atau pesan yang menggunakan media komunikasi tersebut. Penyandian terhadap file diperlukan untuk meminimalisasi file yang akan dikirim atau yang disimpan agar tidak diketahui, dimanipulasi atau diambil oleh pihak yang tidak bertanggungjawab, maka dibutuhkan metode yang dapat menjaga kerahasiaan suatu informasi yang salah satunya adalah kriptografi. Algoritma rijndael dan twofish sama-sama memiliki panjang kunci yang dapat digunakan, yaitu 128, 192 dan 256 bit. Algoritma rijndael memiliki putaran sebanyak 10 kali dengan panjang kunci 128 bit dan melakukan beberapa transformasi dalam proses enkripsi dan dekripsinya. Algoritma twofish menggunakan sejenis jaringan feistel, fungsi f, MDS matriks dan pseudo hadamard transform dengan putaran sebanyak 16 kali dengan panjang kunci 128 bit dalam proses enkripsi dan dekripsinya.

Kata Kunci: Kriptografi, algoritma, rijndael, twofish.

ABSTRACT

The ease of accessing communication media by utilizing advances in information technology will certainly have an impact on the security of information or messages that use these communication media. Encryption of files is needed to minimize files that will be sent or stored so that they are not known, manipulated or taken by irresponsible parties, a method is needed to maintain the confidentiality of information, one of which is cryptography. The rijndael and twofish algorithms both have key lengths that can be used, namely 128, 192 and 256 bits. The rijndael algorithm has a rotation of 10 times with a 128-bit key length and performs several transformations in the encryption and decryption process. The Twofish algorithm uses a type of feistel network, function f, MDS matrix and pseudo hadamard transform with a rotation of 16 times with a key length of 128 bits in the process of encryption and decryption.

Keywords: Cryptography, algorithm, rijndael, twofish.

1. PENDAHULUAN

Kemudahan pengaksesan media komunikasi dengan memanfaatkan kemajuan teknologi informasi tentunya akan memberikan dampak bagi keamanan informasi atau pesan yang menggunakan media komunikasi tersebut. Penyandian terhadap *file* diperlukan untuk meminimalisasi *file* yang akan dikirim atau yang disimpan agar tidak diketahui, dimanipulasi atau diambil oleh pihak yang tidak bertanggungjawab, maka dibutuhkan metode yang dapat menjaga kerahasiaan suatu informasi yang salah satunya adalah kriptografi.

Algoritma *rijndael* dan algoritma *twofish* adalah salah satu algoritma kriptografi modern dengan kunci simetri *cipher* blok. Algoritma *rijndael* menggunakan substitusi, permutasi dan sejumlah putaran yang dikenakan pada tiap blok yang akan dienkripsi atau dekripsi. Setiap putarannya, *rijndael* menggunakan kunci yang berbeda. Algoritma *rijndael* dapat mendukung panjang kunci 128, 192 dan 256 *bit*. Algoritma *twofish* beroperasi pada satu blok *plaintext*. *Block plaintext* pada *twofish* terdiri dari 128 *bit*. Algoritma *twofish* menggunakan struktur sejenis *feistel* dalam 16 putaran dengan tambahan teknik *whitening* terhadap *input* dan *output* serta fungsi-fungsi yang digunakan untuk membangkitkan kunci. Algoritma *twofish* mendukung kunci 128, 192, dan 256 *bit*.

Berdasarkan latar belakang di atas, maka rumusan masalah yang dibahas dalam penelitian ini diuraikan sebagai berikut :

1. Bagaimana proses enkripsi dan dekripsi *file* dengan algoritma *rijndael* dan algoritma *twofish*?
2. Bagaimana *performance* algoritma *rijndael* dan algoritma *twofish* dalam proses komputasi, kecepatan enkripsi, ukuran *file* setelah enkripsi atau dekripsi dan ketahanan algoritma dari serangan?

Adapun yang menjadi batasan masalah dalam penulisan penelitian ini adalah sebagai Data yang dienkripsi adalah data yang berbentuk *file* dengan bentuk ekstensi *.txt* dan bukan berbentuk karakter yang diinputkan secara manual, Algoritma kriptografi yang digunakan adalah algoritma *rijndael* dan algoritma *twofish*, khususnya dalam proses enkripsi dan dekripsi *file*, Tidak membahas tentang proses enkripsi dan dekripsi dalam komunikasi *client server* atau *transmitter* (pengirim) dan *receiver* (penerima) dan Panjang kunci yang digunakan adalah 128 *bit*.

2. LANDASAN TEORI

2.1 Analisis

Analisis adalah kata yang berasal dari bahasa latin *ana* yang artinya menjadi, kembali, kepada (*ana* mempunyai banyak arti), dan *lysis* yang artinya pecah. Jadi, analisa artinya memecah kembali, atau menguraikan ke dalam bagian-bagian kecil (Gunawan Wiradi, 2009). Berdasarkan pendapat ahli tersebut, maka dapat disimpulkan bahwa analisa adalah suatu aktivitas awal dari suatu kegiatan yang diuraikan dan dikelompokkan menjadi beberapa bagian untuk mendapatkan pengertian yang tepat secara keseluruhan.

2.2 Perbandingan

Perbandingan adalah pernyataan matematika secara sederhana yang membandingkan dua besaran atau lebih dan besaran-besaran tersebut harus memiliki satuan yang sama (Muflihah, 2006). Perbandingan adalah dua buah bilangan yang dibandingkan satu sama lainnya atau membandingkan dua nilai atau lebih dari suatu besaran yang sejenis dan dinyatakan dengan cara yang sederhana (Siti Nuraisyah, 2014).

2.3 Kriptografi

Kriptografi berasal dari bahasa Yunani, *crypto* dan *graphia*. *Crypto* berarti *secret* (rahasia) dan *graphia* berarti *writing* (tulisan). Menurut terminologinya, kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain (Dony Ariyus, 2008). Kriptografi Modern dipicu oleh perkembangan peralatan komputer digital. Dengan computer digital, *cipher* yang lebih kompleks menjadi sangat mungkin untuk dapat dihasilkan. Tidak seperti kriptografi klasik yang mengenripsi karakter per karakter, kriptografi modern beroperasi pada *string biner*.

2.4 Algoritma Rijndael

Algoritma *rijndael* disebut juga dengan *Advanced encryption standard* (AES). Algoritma *rijndael* menggunakan substitusi dan permutasi, dan sejumlah putaran (*cipher* berulang), setiap putaran meggunakan kunci internal yang berbeda (kunci setiap putaran disebut *round key*). Panjang kunci yang digunakan adalah 128 *bit* sampai 256 *bit* dengan langkah 32 *bit*, *rijndael* beroperasi dalam orientasi *byte* (untuk memangkuskan implementasi algoritma ke dalam *software* dan *hardware*) (Rinaldi Munir, 2006).

Algoritma *rijndael* mempunyai 3 parameter (Rinaldi Munir, 2006) :

1. *Plaintext* : *array* yang berukuran 16-*byte*, yang berisi data masukan.
2. *Ciphertext* : *array* yang berukuran 16-*byte*, yang berisi hasil enkripsi.

Key : *array* yang berukuran 16-*byte*, yang berisi kunci *ciphering* (disebut juga *cipher key*).

Langkah-langkah enkripsi untuk algoritma *rijndael* (Rinaldi Munir, 2006), adalah sebagai berikut :

1. *Addroundkey*, melakukan *XOR* antara *state* awal (plainteks) dengan *cipher key*, tahap ini disebut juga *initial round*.

2. Putaran sebanyak Nr-1, proses yang dilakukan pada tiap putaran adalah *subbytes, shiftrows, mixcolumns,* dan *addroundkey*.
3. *Final round,* proses untuk putaran terakhir melakukan *subbytes, shiftrows, addroundkey*.

2.5 Algoritma Twofish

Algoritma *twofish* memiliki beberapa blok pembangun (Dony Ariyus, 2008) yaitu :

1. Jaringan *Feistel*
   Jaringan *Feistel* merupakan model komputasi berulang yang digunakan oleh banyak *cipher block*. Fungsi dari model jaringan *Feistel* adalah untuk memetakan suatu fungsi enkripsi yang sederhana menjadi fungsi enkripsi yang rumit dan kuat. Jaringan *Feistel* memiliki sifat *reversible,* dalam konteks ini berarti dapat menggunakan fungsi enkripsi dan dekripsi yang sama tanpa perlu mengubah algoritma yang digunakan. Cukup dengan mengganti urutan kunci yang digunakan pada tiap iterasi yang dilakukan.
2. *S-box*
   S-box merupakan matriks substitusi yang digunakan untuk memetakan *bit-bit*. *S-box* dapat bervariasi bentuk dan ukuran *input outputnya*. *Twofish* menggunakan empat *S-box* yang berbeda, dengan ukuran 8x8 *bit* permutasi.
3. Matriks *Maximum Distance Separable*
   Matriks *Maximum Distance Separable (MDS)* adalah matriks transformasi dari sebuah kode linear. Perubahan *pseudo-hadamard* merupakan transformasi dua arah yang menghasilkan difusi.
4. *Whitening,* merupakan teknik untuk meningkatkan keamanan dari *cipher* blok yang menggunakan iterasi, tujuannya adalah agar *input* dan *output* dari fungsi F tidak diketahui. *Whitening* dilakukan dengan cara mengubah data dengan meng-XOR data dengan sebagian dari kunci sebelum iterasi pertama dan setelah iterasi terakhir dari enkripsi

3. PEMBAHASAN

3.1 Analisis Sistem

File yang ada pada komputer bisa jadi merupakan data yang sangat penting dan rahasia yang tidak dimungkinkan orang lain mengaksesnya. Dengan mengimplementasikan kriptografi, sebuah *file* menjadi tidak dimengerti oleh pihak yang tidak berwenang karena suatu proses perubahan terhadap *file* itu sendiri dengan sandi atau kode tertentu. Sebuah kunci digunakan dalam algoritma untuk menyandikan semua data menjadi *cryptogram*. Sehingga, sebuah data hanya akan diakses oleh orang yang mengetahui kunci saja dan kemungkinan dicuri atau dirusak telah diminimalisasi.

3.1.1. Proses Enkripsi Algoritma Rijndael

File yang dienkripsi dengan algoritma *rijndael* ditransformasi secara berulang kali selama beberapa putaran. Kunci yang digunakan adalah 128 *bit,* maka jumlah putaran adalah 10 kali, jika *file* teks tidak berjumlah 128 *bit,* maka harus ditambahkan 00 hingga berjumlah 128 *bit* agar *file* bisa dienkripsi.

Tahapan-tahapan enkripsi *file* algoritma *rijndael* adalah sebagai berikut:

1. *File* diinputkan
2. Isi *file* diblok sepanjang 128 *bit* untuk proses pengenkripsian dengan kunci sepanjang 128 *bit*.
3. Kunci diekspansi dengan membagi kunci menjadi 4 *word*, menjadi W_0, W_1, W_2, W_3, kemudian mencari kunci untuk setiap putaran dengan rumus:
   $W_4 = W_0 \oplus Subword(Rootword(W_3)) \oplus R_{con}$
4. Plainteks dibagi menjadi 4 *word* menjadi P_0, P_1, P_2, P_3.
5. Melakukan transformasi *addroundkey*, yaitu melakukan operasi Xor antara plainteks dengan kunci.
6. Melakukan transformasi *subbytes*, yaitu melakukan pergantian plainteks yang sudah di-Xor dengan kunci menjadi isi yang berada di dalam tabel *s-box rijndael* berdasarkan nilai plainteks, nilai 4 *bit* yang pertama menjadi nilai baris dan 4 *bit* yang kedua menjadi nilai kolom.

7. Melakukan transformasi *shiftrows*, yaitu melakukan pergeseran terhadap baris plainteks, dengan ketentuan.
8. Melakukan transformasi *mixcolumns*, yaitu melakukan percampuran antara baris dan kolom dengan melakukan perkalian dengan matriks yang sudah ditentukan.
9. Proses transformasi dilakukan sebanyak 9 putaran, kemudian untuk putaran terakhir atau *final round* proses transformasi *mixcolumns* tidak digunakan, maka cipherteks didapatkan.

3.1.2 Proses Dekripsi Algoritma Rijndael

Proses dekripsi pada algoritma *rijndael* merupakan kebalikan dari proses enkripsi. Proses dekripsi *rijndael* kunci yang digunakan sama dengan kunci untuk mengenkripsi plainteks. Tahapan dalam proses dekripsi adalah kebalikan dari proses enkripsi yang dimulai dari proses *final round*, kemudian melakukan 10 putaran transformasi.

Langkah-langkah dalam mendekripsi cipherteks adalah sebagai berikut:

1. *File* diinputkan
2. Isi *file* diblok sepanjang 128 *bit* untuk proses pengdekripsian dengan kunci sepanjang 128 *bit*.
3. Mengekspansi kunci
   Kunci dibagi menjadi beberapa blok, jika panjang kunci 128 *bit*, maka panjang kunci (Nk) dibagi 32.
   128/32 = 4, urutan pengisian baris dan kolom adalah berdasarkan kolom, maka
   W_0, W_2, W_1, W_3
   R_{con} yang digunakan adalah:
   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
   Sebelum mencari *RoundKey* yang pertama, dicari terlebih dahulu nilai W3 dengan rumus:
   W_3 = SubWord (RootWord(W_3))
   Kemudian mencari *RoundKey* yang pertama dengan rumus:
   $W_4 = W0 \oplus W3 \oplus Rcon1$
   W_3 = RootWord 524d414d = 4d414d52
   = SubWord 4d414d52 = e383e300
   W_4 = 53544d49 $\oplus$ e383e300 $\oplus$ 01000000
   = b1d7ae49
   Untuk mendapatkan *RoundKey* kolom II sampai kolom IV tidak perlu di XOR kan dengan Rcon, Rcon hanya digunakan untuk mendapatkan *byte*
   kolom I tiap putaran yang di XOR kan hanya hasil dari kolom sebelumnya,
   maka:
   $W_5 = W_4 \oplus W_1$
   = b1d7ae49 $\oplus$ 4b5f4255
   = fa88ec1c
4. Melakukan putaran pertama transformasi *inverse of addroundkey*, meng-Xor-kan cipherteks dengan kunci putaran kesepuluh.

Cipherteks		*roundkey* X		Hasil
69 6a d8 2f		13 e3 f3 4d		7a 89 2b 62
c4 65 cd b4	$\oplus$	11 94 07 2b	=	d5 f1 ca 9f

5. Melakukan putaran pertama transformasi inverse *shiftrow*

7a 89 2b 3d		7a 89 2b 3d
d5 ef ca 9f	=	ef ca 9f d5

6. Melakukan putaran pertama transformasi inverse *subbyte*

Hasil *Shiftrow*		*S-Box*
7a 89 2b 3d		bd f2 0b ab
ef ca 9f d5	=	6e b5 a1 10

7. Untuk putaran kedua sampai sepuluh dilakukan transformasi *inverse of addroundkey*, *inverse mixcolumn*, *inverse shiftrow* dan *inverse subbyte*. Hasil seluruh putaran dapat dilihat pada tabel berikut:

Tabel 1 Hasil putaran dua sampai sepuluh dekripsi rijndael

Putaran kedua				Putaran ketiga				Putaran keempat			
fd	05	35	f1	d1	79	b4	d6	c6	f7	cc	84
83	e5	47	fe	87	c4	55	6f	2f	5e	79	f9
ba	d0	96	37	6c	30	94	f4	e1	ed	39	cf
d2	d7	4e	f1	0f	0a	ad	1f	09	c3	5d	5d
Putaran kelima				Putaran keenam				Putaran ketujuh			
c8	9b	25	b0	24	69	6e	88	fa	25	40	57
16	7a	02	26	72	66	d2	42	63	b3	66	24
77	c9	79	19	40	b3	75	5b	6a	39	8a	4d
bc	3b	92	96	23	fa	32	63	28	09	31	17
Putaran kedelapan				Putaran kesembilan				Putaran kesepuluh			
49	55	da	1f	1a	1f	11	1c	00	40	80	c0
15	e5	ca	0a	0e	00	05	12	10	50	90	d0
59	d7	94	63	17	11	0c	0d	20	60	a0	e0
8f	a0	fa	ff	08	1d	00	09	30	70	b0	f0

8. Langkah yang terakhir adalah *keyaddition*:

Hasil putaran 10		Kunci		Plainteks
1a 1f 11 1c		53 4b 44 52		49 54 55 4e
0e 00 05 12	⊕	54 5f 49 4d	=	5a 5f 4c 5f

3.1.3 Proses Enkripsi Algoritma Twofish

twofish menggunakan sebuah struktur jaringan *feistel* 16 putaran dengan tambahan *whitening* pada masukan dan keluaran. Satu-satunya unsur *non-feistel* adalah 1 *bit* rotasi. Perputaran dapat diperoleh ke dalam fungsi f untuk membuat suatu struktur *feistel* murni, tapi memerlukan suatu tambahan perputaran kata-kata yang tepat sebelum langkah keluaran *whitening*.

Tahapan dalam proses enkripsi *file* dengan algoritma *twofish* adalah sebagai berikut:

1. *File* diinputkan, isi *file* diblok sepanjang 128 *bit*.
2. blok plainteks 128 *bit* dibagi menjadi 4 *word*, 2 *word* pertama dijadikan bagian kiri dan 2 *word* kedua dijadikan bagian kanan.
3. Melakukan ekspansi kunci dengan menggunakan fungsi h.
4. Melakukan putaran sebanyak 16 kali, dengan tiap putaran melewati fungsi f.
5. Proses didalam fungsi f adalah fungsi g, kemudian MDS matriks dan *pseudo hadamard transform* (PHT).

3.1.4 Proses Dekripsi Algoritma Twofish

Proses dekripsi algoritma *twofish* adalah sama dengan proses enkripsi, tetapi hanya arahnya saja berlawanan. Proses yang dilalui secara berurutan yaitu : *output whitening*, *swap* blok terakhir, 16 iterasi dekripsi dan *input whitening*. Inputnya adalah cipherteks dan kunci untuk memperoleh plainteks sama saja dengan kunci enkripsi.

3.1.5 Perbandingan Algoritma Rijndael dan Algoritma Twofish

Perbandingan antara algoritma *rijndael* dan *twofish* dapat dilihat dari tabel berikut:

Tabel 2 Perbandingan algoritma rijndael dan twofish

Objek	Algoritma kriptografi yang lebih unggul	
	Rijndael	*Twofish*
Kecepatan	✓	-
Ukuran Memori	✓	-
Ketahanan terhadap serangan	-	✓

1. Kecepatan proses enkripsi dan dekripsi

Perbandingan yang dilakukan adalah seberapa banyak waktu yang dibutuhkan dalam melakukan proses enkripsi dan dekripsi, perbandingan kecepatan dapat
dihitung melalui operasi yang dilakukan pada tiap algoritma yang mengenkripsi data sebesar 128 *bit*.

Waktu proses enkripsi dan dekripsi pada algoritma *rijndael* lebih rendah dibandingkan dengan waktu proses pada algoritma *twofish* dikarenakan struktur *cipher* dan penjadwalan kunci pada algoritma *rijndael* lebih sederhana dibandingkan *twofish*. Semakin sederhana struktur *cipher* dan penjadwalan pada algoritma kriptografi maka waktu proses enkripsi dan dekripsi yang dibutuhkan akan semakin kecil. Pada struktur cipher *rijndael* ditekankan kesederhanaan tranformasi dengan menggunakan matrik dengan panjang 16 *byte array* dimana pada matrik tersebut terjadi beberapa proses transformasi pergeseran nilai kolom dan baris. Pada algoritma *twofish* digunakan jaringan *Feistel* sebagai pembentukan struktur dan proses penjadwalan kunci. Pada jaringan *Feistel* tersebut diperlukan waktu untuk proses *round*, semakin banyak proses *round* pada jaringan *Feistel* maka semakin besar waktu yang dibutuhkan untuk melakukan proses enkripsi dan dekripsi. Algoritma *twofish* memiliki *round* dalam jaringan *Feistel* sebanyak 16 putaran dengan di dalamnya melalui proses fungsi g, fungsi h, fungsi f yang merupakan suatu perhitungan yang rumit.
Rijndael :
Jumlah putaran = 10
Transformasi = 4
Diasumsikan kecepatan computer 0,6 proses/detik.
Pada putaran 10 transformasi mixcolumns ditiadakan.
Kecepatan: 10 * 4 – 1/ 0,6 = 65 detik = 0,065 milidetik
Twofish:
Putaran = 16
Operasi yang dilakukan : 7
Kecepatan : 16 * 7 / 0,6 =186,7 detik =0,187 milidetik

Tabel 3 Perbandingan kecepatan algoritma *rijndael* dan *twofish*

Algoritma	Kecepatan (ms)
Rijndael	0,065
Twofish	0,187

2. Ukuran data setelah enkripsi
   Ukuran data dapat dilihat dari besar data yang dihasilkan, *rijndael* memiliki hasil enkripsi yang lebih kecil dari *twofish*, hal ini dipengaruhi dari kesederhanaan algoritma dalam mengenkripsi data.
   Perbandingan ukuran data setelah dienkripsi dengan aplikasi *Netbeans* menggunakan *file* a.txt dapat dilihat pada tabel berikut:

Tabel 4 Perbandingan ukuran data setelah dienkripsi

Algoritma	Ukuran sebelum enkripsi	Ukuran setelah enkripsi
Rijndael	1 kb	1 kb
Twofish	1 kb	1,089 kb

3. Analisa *Brute Force Attack* pada sistem ini adalah dengan melakukan perhitungan matematis dari masing-masing kunci untuk mendapatkan nilai durasi proses memecahkan kemungkinan kunci. Pada Algoritma *rijndael*, kemungkinan kunci yang dapat dihasilkan dengan menggunakan panjang 128 *bit* adalah sejumlah 3,4 x 10^{38} kunci. Asumsi kecepatan komputasi adalah 10^6 *key/sec* maka:

$$\frac{3{,}4\times10^{38}}{10^{6}}=\frac{3{,}4\times10^{32}\,\text{sec}}{3{,}1536\times10^{7}}=1{,}08\times10^{25}$$

menghasilkan durasi sejumlah 1,08 x 10^{25} tahun untuk memecahkan seluruh kemungkinan kunci. Pada algoritma *twofish* , Kemungkinan kunci yang dapat dihasilkan dengan menggunakan panjang 128 *bit* adalah sejumlah 1,2 x 10^{77} kunci. Asumsi kecepatan komputasi adalah 10^{6} *key/sec* maka :

$$\frac{1{,}2\times10^{77}}{10^{6}}=\frac{1{,}2\times10^{71}\,\text{sec}}{3{,}1536\times10^{7}}=3{,}8\times10^{63}$$

menghasilkan durasi sejumlah 3,8 x 10^{63} tahun untuk memecahkan seluruh kemungkinan kunci. Dari hasil analisa melalui serangan *Brute Force Attack*, algoritma *twofish* memiliki ketahanan yang lebih lama dibandingkan algoritma *rijndael*.

Tabel 5 Perbandingan ketahanan algoritma

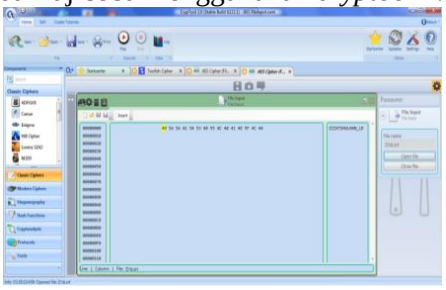
Algoritma	Waktu pemecahan algoritma
Rijndael	$1{,}08\times10^{25}$ tahun atau 10800000000000000000000 abad
Twofish	$3{,}8\times10^{63}$ atau 380000000000000000000000000000000000000000000000000000000000000000000000 0 Abad

4. Kesimpulan

Algoritma *rijndael* lebih baik dalam hal kecepatan dan penggunaan memori, sedangkan algoritma *twofish* memiliki ketahanan yang lebih baik, dikarenakan butuh waktu yang lebih lama untuk memecahkan algoritma tersebut. Berdasarkan hasil analisa yang telah dilakukan, maka algoritma *rijndael* lebih baik untuk digunakan karena penggunaan memori yang lebih sedikit memungkinkan pengguna untuk menggunakannya dalam komputer yang memiliki kapasitas rendah dan ketahanan algoritma *rijndael* juga membutuhkan waktu yang cukup lama untuk memecahkannya.

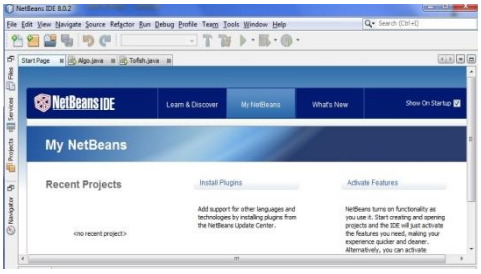
4. IMPLEMENTASI

Berikut adalah tampilan dari ujicoba menggunakan *cryptool 2.0*:



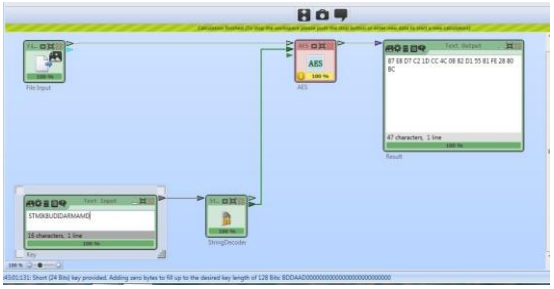
Gambar 1 Tampilan memasukkan *file*

Berdasarkan gambar 1, *file* dimasukkan ke dalam aplikasi *cryptool* dan akan ditampilkan isi *file* tersebut.



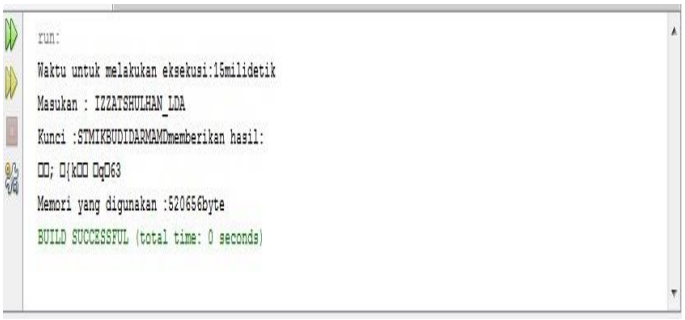
Gambar 2 Tampilan awal Netbeans 8.2.0

Berdasarkan gambar 2, merupakan tampilan awal aplikasi Netbeans 8.2.0 yang digunakan untuk melihat waktu enkripsi.



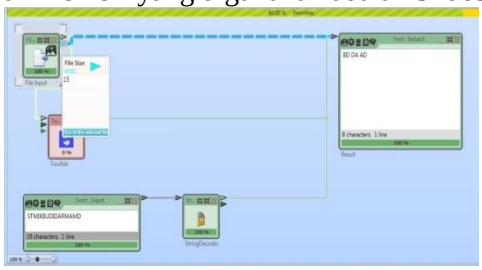
Gambar 3 Tampilan proses enkripsi *rijndael*

Berdasarkan gambar 3 terlihat bagaimana proses enkripsi dengan menggunakan algoritma *rijndael*. Untuk mengetahui berapa waktu dan memori yang diperlukan oleh algoritma *rijndael* dalam melakukan enkripsi dapat dilihat pada gambar 4 berikut ini:



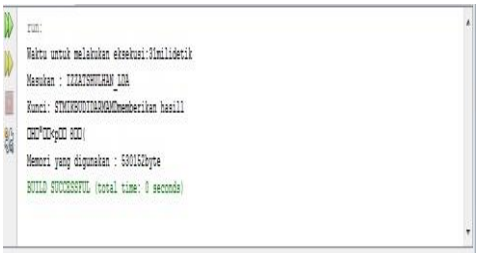
Gambar 4 Waktu dan memori proses enkripsi *rijndael*

Berdasarkan gambar 4, terlihat waktu yang diperlukan algoritma *rijndael* dalam proses enkripsi adalah 15 milidetik dan memori yang digunakan adalah 520658 *byte*.



Gambar 5 Tampilan proses enkripsi *twofish*

Berdasarkan gambar 5 terlihat bagaimana proses enkripsi dengan menggunakan algoritma *twofish*. Memerlukan waktu 0,031 detik dan memori yang digunakan adalah 530152 *byte* seperti terlihat pada gambar 6 berikut ini:



Gambar 6 Waktu dan memori proses enkripsi *twofish*

5. KESIMPULAN

Berdasarkan pembahasan dan evaluasi sebelumnya, maka dapat diambil kesimpulan-kesimpulan. Adapun kesimpulan-kesimpulan tersebut adalah sebagai berikut:

1. Algoritma *rijndael* dan *twofiish* merupakan algoritma modern yang memiliki proses enkripsi dan dekripsi berbasis *block cipher*, Algoritma *rijndael* memiliki 10 putaran dalam melakukan proses enkripsi dan dekripsi serta beberapa transformasi. Sedangkan algoritma *twofish* berupa struktur sejenis feistel yang memiliki beberapa fungsi dalam proses enkripsi dan dekripsinya.
2. Ditinjau dari kecepatan dalam proses enkripsi dan dekripsi dan ukuran *file* yang dihasilkan setelah enkripsi algoritma *rijndael* lebih unggul, sedangkan jika ditinjau dari segi ketahanan algoritma terhadap serangan, algoritma *twofish* memiliki keunggulan dikarenakan butuh proses yang lebih lama untuk memecahkannya. Dilihat dari keunggulan algoritma pada tiap kriteria, algoritma *rijndael* lebih unggul dari algoritma *twofish* karena dapat digunakan dalam komputer yang memiliki kapasitas memori yang rendah.

DAFTAR PUSTAKA

[1]. Anwar. (2008). Konsep Jitu Matematika SMP Untuk Kelas 1,2 Dan 3. Jakarta: PT Wahyu Media.

[2]. Arifin, Johar. (2007). Cara Cerdas Menilai Kinerja Perusahaan (Aspek Finansial & Non Finansial) Berbasis Komputer. Jakarta: PT.Elex Media Komputindo.

[3]. Ariyus, Dony. (2008). Pengantar Ilmu Kriptografi Teori Analisis dan Implementasi. Yogyakarta: Penerbit Andi.

[4]. Chan, S. A. (2014). Penerapan Kriptografi Rijndael dalam Mengamankan File Menggunakan Interface USB Flashdisk (Memory Eksternal). Medan: Pelita Informatika Budi Darma, 6, 2301-9425.

[5]. Enterprise, Jubilee. (2010) Rahasia Manajemen File. Jakarta: PT.Elex Media Komputindo.

[6]. Hidayat, Syarif. (2010). Parallel Programming. Yogyakarta: Penerbit Andi.

[7]. Muflihah. (2006). Kumpulan Lengkap Rumus Matematika SMP. Jakarta: Puspa Swara.

[8]. Munir, Rinaldi. (2011). Algoritma dan Pemrograman. Bandung: Informatika Bandung.

[9]. Munir, Rinaldi. (2006). Kriptografi. Bandung: Informatika Bandung.

[10]. Nuraisyah, Siti. (2014). Sekali Baca Langsung Inget Rumus Detail Matematika Untuk SMP/MTS. Jakarta: Kunci Aksara.

[11]. Rachmat C, Antonius. (2010). Algoritma dan Pemrograman dengan Bahasa C-Konsep, Teori & Implementasi. Yogyakarta: Penerbit Andi.

[12]. Sadikin, Rifki. (2010). Kriptografi untuk Keamanan Jaringan dan Implementasinya dalam Bahasa Java. Yogyakarta: Penerbit Andi.

[13]. Suarga. (2006). Algoritma dan Pemrograman. Yogyakarta: Penerbit Andi.

[14]. Wiradi, Gunawan. (2009). Metodologi Studi Agraria. Bogor: Sajogyo Institute.