APLIKASI PENGAMANAN DOKUMEN DENGAN MENGGUNAKAN TEKNIK KRIPTOGRAFI ALGORITMA AES-RINJDAEL

Ari

Teknik Informatika STMIK ATMA LUHUR PANGKALPINANG

Jl.Jend. Sudirman Selindung Lama Pangkalpinang Kepulauan Babel

email: aridoank85@gmail.com

Abstrak

Seiring dengan perkembangan zaman, kebutuhan manusia yang semakin meningkat termasuk kebutuhan akan informasi. Oleh sebab itu, pengiriman dan penyimpanan data melalui media elektronik memerlukan suatu proses yang mampu menjamin keamanan dan keutuhan dari data tersebut. 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 kembali menjadi data asal. Dengan cara penyandian tadi, data asli tidak akan terbaca oleh pihak yang tidak berkepentingan, melainkan hanya oleh penerima yang memiliki kunci dekripsi. Didorong oleh kegunaan yang penting tadi, teknik (algoritma) penyandian telah berkembang sejak zaman dahulu kala. Mulai dari era sebelum masehi, hingga sekarang algoritma penyandian ini selalu berkembang. Pertimbangan bahwa sebuah standard algoritma yang baru sangatlah diperlukan untuk tetap menjaga kerahasiaan suatu data. Dalam hal ini, kunci yang lebih panjang juga merupakan keharusan.

Saat ini, AES digunakan sebagai standar 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 mengenkripsi dan mendekripsi blok data sepanjang 128 bit dengan panjang kunci 128 bit, 192 bit, atau 256 bit.

1. Pendahuluan

Berbagai organisasi, perusahaan, atau pun pihak – pihak lain telah memanfaatkan teknologi komputer untuk menyimpan dan mengelola data organisasi atau perusahaannya. Saat ini, keamanan terhadap data yang tersimpan di dalam komputer sudah menjadi persyaratan mutlak. Pengamanan terhadap jaringan komputer yang terhubung dengan dokumen-dokumen sudah tidak lagi menjamin keamanan data karena kebocoran data dapat disebabkan oleh “orang dalam” atau pihak – pihak yang langsung berhubungan dengan dokumen-dokumen seperti administrator . Hal ini menyebabkan pengguna dokumen harus menemukan cara untuk mengamankan data tanpa campur tangan administrator.

Kriptografi dapat digunakan untuk mengamankan data. Oleh karena itu, pengguna dokumen membutuhkan bantuan untuk memenuhi kebutuhan keamanan akan data yang disimpannya. Penerapan kriptografi pada Tugas Akhir ini akan

difokuskan bagaimana kriptografi dapat mengamankan data dengan tiga kunci yang panjang kuncinya berbeda-beda, mulai dari 128, 192 dan 256 byte. Algoritma kriptografi yang akan digunakan ialah kriptografi AES (Anvanced Encryption Standard) dengan algoritma Rinjdael. Teknik kriptografi AES ini dipilih karena keamanannya lebih terjamin di bandingkan dengan algoritma-algoritma yang lain.

Berdasarkan atas informasi di atas, penulis membuat sebuah implementasi dengan menerapkan metode sistem enkripsi dengan menggunakan algoritma Rinjdael untuk membuat aplikasi kriptografi untuk keamanan dokumen-dokumen yang memerlukan pengamanan dari pihak-pihak yang tidak berkepentingan.

Enkripsi dilakukan pada saat pengiriman dengan cara mengubah data asli menjadi data rahasia, sedangkan dekripsi dilakukan pada saat penerimaan dengan cara mengubah data rahasia menjadi data asli. Jadi data yang dikirimkan selama proses pengiriman adalah data rahasia, sehingga data asli tidak dapat diketahui oleh pihak yang tidak berkepentingan. Data asli hanya dapat diketahui oleh penerima dengan menggunakan kunci rahasia.

Disini enkripsi dapat diartikan sebagai kode atau *cipher*. Sebuah *system* pengkodean menggunakan suatu tabel atau kamus yang telah didefinisikan untuk kata dari informasi atau yang merupakan bagian dari pesan, data, atau informasi yang di kirim. Sebuah *cipher* menggunakan suatu algoritma yang dapat meng - kodekan semua aliran data (*stream*) bit dari suatu pesan asli (*plaintext*) menjadi *cryptogram* yang tidak di mengerti. Karena *system cipher* merupakan suatu sistem yang telah siap untuk di outomasi, maka teknik ini digunakan dalam sistem keamanan jaringan komputer.

National Institute of Standard and Technology (NIST) untuk pertama kalinya mengumumkan suatu algoritma standar penyandian data yang telah dijadikan standard sejak tahun 1977 adalah *Data Encryption Standard* (*DES*). Kekuatan *DES* ini terletak pada panjang kuncinya yaitu 56-bit. Untuk menanggapi keinginan agar mengganti algoritma DES sebagai standar. Perkembangan kecepatan perangkat keras dan meluasnya penggunaan jaringan komputer terdistribusi mengakibatkan penggunaan *DES*, dalam beberapa hal, terbukti sudah tidak aman dan tidak mencukupi lagi terutama dalam hal yang pengiriman data melalui jaringan internet. Perangkat keras khusus yang bertujuan untuk menentukan kunci 56-bit *DES* hanya dalam waktu beberapa jam sudah dapat dibangun. Beberapa pertimbangan tersebut telah manandakan bahwa diperlukan sebuah standard algoritma baru dan kunci yang lebih panjang. *Triple-DES* muncul sebagai alternative solusi untuk masalah-masalah yang membutuhkan keamanan data tingkat tinggi seperti perbankan, tetapi ia terlalu lambat pada beberapa penggunaan enkripsi.

Pada tahun 1997, *the U.S. National Institue of Standards and Technology* (*NIST*) mengumumkan bahwa sudah saatnya untuk pembuatan standard algoritma penyandian baru yang kelak diberi nama *Advanced Encryption Standard* (*AES*). Algoritma *AES* ini dibuat dengan tujuan untuk menggantikan algoritma *DES & Triple-DES* yang telah lama digunakan dalam menyandikan data elektronik. Setelah melalui beberapa tahap seleksi, algoritma *Rijndael* ditetapkan sebagai algoritma kriptografi *AES* pada tahun 2000.

Algoritma *AES* merupakan algoritma kriptografi simetrik yang beroperasi dalam mode penyandi blok (*block cipher*) yang memproses blok

data 128-bit dengan panjang kunci 128-bit (*AES*- 128), 192-bit (*AES*-192), atau 256-bit (*AES*-256). Beberapa mode operasi yang dapat diterapkan pada algoritma kriptografi penyandi blok *AES* di antaranya adalah *Electronic Code Book* (*ECB*), *Cipher Block Chaining* (*CBC*), *Cipher Feedback* (*CFB*), dan *Output Feedback* (*OFB*). Implementasi *AES* dengan mode operasi *ECB*, *CBC*, *CFB*, dan *OFB* tentu saja memiliki kelebihan dan kekurangan tertentu dalam aspek tingkat keamanan data.

2. TINJAUAN PUSTAKA

2.1 Konsep Dasar Sistem

2.1.1 Pengertian Sistem

Menurut Jogianto HM (1989), Sistem adalah jaringan dari elemen-elemen yang saling berhubungan, membentuk satu kesatuan untuk melaksanakan satu tujuan pokok dari sistem tersebut. Sistem adalah totalisasi dari beberapa himpunan bagian yang saling berinteraksi satu sama lain dan bersama-sama untuk mencapai suatu tujuan atau sekelompok tujuan dalam satu lingkungan. Sedangkan bagian sistem yang biasa disebut juga dengan sub sistem yang ada merupakan suatu kumpulan dari unsur-unsur tertentu namun dalam mencapai tujuan.

Ciri utama dari setiap sistem adalah berorientasi untuk mencapai tujuan. Sistem bersifat terbuka sehingga mempunyai karakteristik ekuifinalitas, artinya bahwa tujuan akhir dari suatu sistem dapat dicapai dengan berbagai kemungkinan permulaan, maka ada berbagai cara yang baik untuk mencapai tujuan tertentu.

1.2 Algoritma AES

Input dan output dari algoritma AES terdiri dari urutan data sebesar 128 bit. Urutan data yang sudah terbentuk dalam satu kelompok 128 bit tersebut disebut juga sebagai blok data atau *plaintext* yang nantinya akan dienkripsi menjadi *ciphertext*. *Cipher key* dari AES terdiri dari *key* dengan panjang 128 bit, 192 bit, atau 256 bit.

1.2.1 Algoritma AES - *Rijndael*

Pada algoritma AES, jumlah blok input, blok output, dan *state* adalah 128 bit. Dengan besar data 128 bit, berarti *Nb* = 4 yang menunjukkan panjang data tiap baris adalah 4 byte. Dengan panjang kunci 128-bit, maka terdapat sebanyak $3{,}4 \times 1038 = 2128$ kemungkinan kunci. Jika komputer tercepat dapat mencoba 1 juta kunci setiap detik, maka akan dibutuhkan waktu $5{,}4 \times 1024$ tahun untuk mencoba seluruh kunci. Jika tercepat yang dapat mencoba 1 juta kunci setiap milidetik, maka dibutuhkan waktu $5{,}4 \times 1018$ tahun untuk mencoba seluruh kunci

Dengan blok input atau blok data sebesar 128 bit, *key* yang digunakan pada algoritma AES tidak harus mempunyai besar yang sama dengan blok input. *Cipher key* pada algoritma AES bisa menggunakan kunci dengan panjang 128 bit, 192 bit, atau 256 bit. Perbedaan panjang kunci akan mempengaruhi jumlah *round* yang akan diimplementasikan pada algoritma AES ini. Di bawah ini adalah Tabel yang memperlihatkan jumlah *round* (*Nr*) yang harus diimplementasikan pada masing-masing panjang kunci.

idak seperti *DES* yang berorientasi bit, *Rijndael* beroperasi dalam orientasi *byte*. Setiap putaran mengunakan kunci internal yang berbeda (disebut *round key*). *Enciphering* melibatkan operasi substitusi dan permutasi.

1.2.1.1 Ekspansi Kunci

Algoritma AES mengambil kunci *cipher*, K, dan melakukan rutin ekspansi kunci (*key expansion*) untuk membentuk *key schedule*.

Ekspansi kunci menghasilkan total Nb(Nr+1) *word*. Algoritma ini membutuhkan set awal *key* yang terdiri dari *Nb word*, dan setiap *round Nr* membutuhkan data kunci sebanyak *Nb word*. Hasil *key schedule* terdiri dari array 4 byte word linear yang dinotasikan dengan $[w_i]$. *SubWord* adalah fungsi yang mengambil 4 byte *word* input dan mengaplikasikan S-Box ke tiap-tiap data 4 byte untuk menghasilkan *word* output. Fungsi *RotWord* mengambil *word* $[a_0, a_1, a_2, a_3]$ sebagai input, melakukan permutasi siklik, dan mengembalikan *word* $[a_1, a_2, a_3, a_0]$. Rcon[i] terdiri dari nilai-nilai yang diberikan oleh $[x^{i-1}, \{00\}, \{00\}, \{00\}]$, dengan x^{i-1} sebagai pangkat dari x (x dinotasikan sebagai {02} dalam *field GF*(2^8). W*ord* ke *Nk* pertama pada ekspansi kunci berisi kunci *cipher*. Setiap word berikutnya, w[i], sama dengan XOR dari *word* sebelumnya, w[i-1] dan *word Nk* yang ada pada posisi sebelumnya, w[i-Nk]. Untuk *word* pada posisi yang merupakan kelipatan *Nk*, sebuah transformasi diaplikasikan pada w[i-1] sebelum XOR, lalu dilanjutkan oleh XOR dengan konstanta *round*, Rcon[i]. Transformasi ini terdiri dari pergeseran siklik dari byte data dalam suatu word *RotWord*, lalu diikuti aplikasi dari *lookup Tabel* untuk semua 4 byte data dari word *SubWord*. Lihat tabel di bawah ini:

1.2.1.2 Enkripsi

Proses enkripsi pada algoritma AES terdiri dari 4 jenis transformasi bytes, yaitu SubBytes, ShiftRows, Mixcolumns, dan AddRoundKey. Pada awal proses enkripsi, input yang telah dikopikan ke dalam *state* akan mengalami transformasi byte AddRoundKey. Setelah itu, *state* akan mengalami transformasi SubBytes, ShiftRows, MixColumns, dan AddRoundKey secara berulang-ulang sebanyak *Nr*. Proses ini dalam algoritma AES disebut sebagai *round function*. *Round* yang terakhir agak berbeda dengan *round-round* sebelumnya dimana pada *round* terakhir, *state* tidak mengalami transformasi MixColumns. Lihat gambar di bawah ini :

1) SubBytes

SubBytes merupakan transformasi byte dimana setiap elemen pada state akan dipetakan dengan menggunakan sebuah tabel substitusi (*S-Box*). Hasil yang didapat dari pemetaan dengan menggunakan Tabel *S-Box*.

2) ShiftRows

Transformasi Shiftrows pada dasarnya adalah proses pergeseran bit dimana bit paling kiri akan dipindahkan menjadi bit paling kanan (rotasi bit). Transformasi ini diterapkan pada baris 2, baris 3, dan baris 4. Baris 2 akan mengalami pergeseran bit sebanyak satu kali, sedangkan baris 3 dan baris 4 masing-masing mengalami pergeseran bit sebanyak dua kali dan tiga kali.

3) MixColumns

MixColumns mengoperasikan setiap elemen yang berada dalam satu kolom pada *state*. Elemen pada kolom dikalikan dengan suatu polinomial tetap $a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$.

Mixcolumns membutuhkan tabel penunjang dalam memperoleh hasil tabel tersebut adalah tabel E dan tabel L.

4) AddRoundKey

Pada proses AddRoundKey, sebuah *round key* ditambahkan pada *state* dengan operasi bitwise XOR. Setiap *round key* terdiri

dari *Nb word* dimana tiap *word* tersebut akan dijumlahkan dengan *word* atau kolom yang bersesuaian dari *state*.

Transformasi *cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES. Transformasi byte yang digunakan pada invers *cipher* adalah InvShiftRows, InvSubBytes, InvMixColumns, dan AddRoundKey. Algoritma dekripsi dapat dilihat pada skema berikut, pada gambar di bawah ini :

1) InvShiftRows

InvShiftRows adalah transformasi byte yang berkebalikan dengan transformasi ShiftRows. Pada transformasi InvShiftRows, dilakukan pergeseran bit ke kanan sedangkan pada ShiftRows dilakukan pergeseran bit ke kiri. Pada baris kedua, pergeseran bit dilakukan sebanyak 3 kali, sedangkan pada baris ketiga dan baris keempat, dilakukan pergeseran bit sebanyak dua kali dan satu kali.

2) InvSubBytes

InvSubBytes juga merupakan transformasi bytes yang berkebalikan dengan transformasi SubBytes. Pada InvSubBytes, tiap elemen pada *state* dipetakan dengan menggunakan Tabel *inverse S-Box*. Tabel ini berbeda dengan Tabel *S-Box* dimana hasil yang didapat dari Tabel ini adalah hasil dari dua proses yang berbeda urutannya, yaitu transformasi *affine* terlebih dahulu, baru kemudian perkalian *invers* dalam GF(2^8).
Perkalian *invers* yang dilakukan pada transformasi InvSubBytes ini sama dengan erkalian *invers* yang dilakukan pada transformasi SubBytes.

3) InvMixColumns

Pada InvMixColumns, kolom-kolom pada tiap *state* (*word*) akan dipandang sebagai polinom atas GF(2^8) dan mengalikan modulo $x^4 + 1$ dengan polinom tetap $a^{-1}(x)$ yang diperoleh dari : $a^{-1}(x) = \{0B\}x^3 + \{0D\}x^2 + \{09\}x + \{0E\}$.

4) Inverse AddRoundKey

Transformasi Inverse AddRoundKey tidak mempunyai perbedaan dengan transformasi AddRoundKey karena pada transformasi ini hanya dilakukan operasi penambahan sederhana dengan menggunakan operasi bitwise XOR.

3. RANCANGAN DAN PEMBUATAN APLIKASI

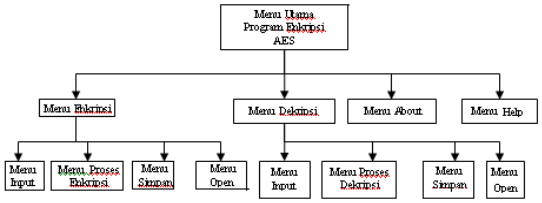
3.1 Rancangan Sistem

Perancangan program aplikasi kriptografi dengan menggunakan algoritma *Advanced Encryption Standard (AES)*. Rancangan ini digunakan untuk meningkatkan keamanan dalam proses pengiriman atau pertukaran data. Rancangan ini dilakukan dalam beberapa tahap yaitu dimulai dari pembuatan diagram hirarki, yang dilanjutkan dengan menggunakan *State Transition Diagram* (STD), namun sebelumnya telah dibuat rancangan *Flowchart* dan *System Development Life Cycle* (SDLC) dan dilanjutkan lagi dengan perancangan antar muka program. setelah rancangan sistem ini selesai dilanjutkan dengan pembuatan program aplikasi. Setelah selesai, program aplikasi tersebut diuji.

3.2 Rancangan Diagram Hirarki

Rancangan ini dibuat untuk memudahkan proses perancangan aplikasi kriptosistem dengan

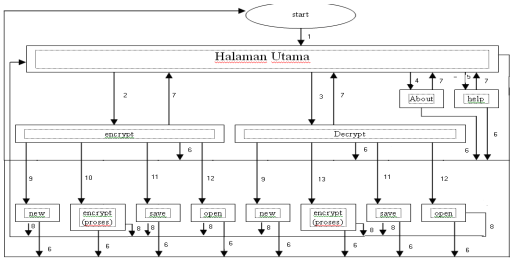
menggunakan algoritma enkripsi *Advanced Encryption Standard* (AES). Diagram hirarki ini memiliki empat sub menu, yaitu: menu Encrypt, menu Decrypt , menu About dan menu Help. Sub menu Encrypt terbagi lagi menjadi New, enkripsi(proses), Save, dan Open . Sub menu Decrypt hampir sama dengan menu encrypt antara lain menjadi New, Dekripsi(proses), Save, dan Open. Gambar diagram hirarki dapat dilihat pada Gambar 28.



Gambar 3.28 :Diagram Hirarki

3.3 Rancangan State Transition Diagram

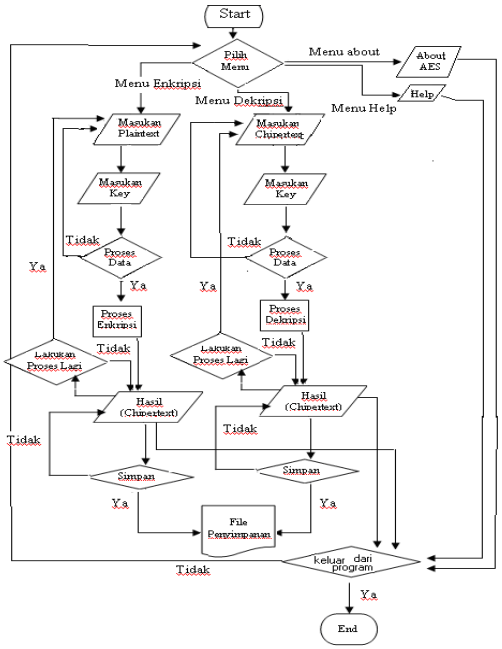
Rancangan ini digunakan untuk mengetahui apa saja yang terjadi pada sistem pada saat timbul perubahan – perubahan antara satu *state* dan *state* yang lain, apa yang menyebabkan timbulnya perubahan itu, dan apa akibat yang ditimbulkan dari perubahan itu. Rancangan *State Transition Diagram* untuk program aplikasi ini dapat dilihat pada Gambar 29.



Gambar 3.29 : Diagram State Transition Diagram (STD)

3.4 Rancangan Flowchart

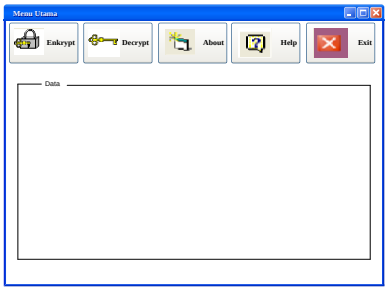
Rancangan ini digunakan untuk mendesain dan merepresentasikan program. Sebelum pembuatan program, fungsinya adalah mempermudah *programmer* dalam menentukan alur logika program yang akan dibuat. Sesudah pembuatan program fungsinya adalah untuk menjelaskan alur program kepada orang lain atau user. Rancangan ini dapat dilihat pada Gambar 30 dan 31.



Gambar 3.30 : Flow Chart Menu Aplikasi AES

3.5 Rancangan Antarmuka

Rancangan ini digunakan untuk mendukung proses pembuatan program aplikasi kriptosistem dengan menggunakan algoritma AES. Rancangan antarmuka ini terdiri dari satu layar menu utama dapat dilihat ada Gambar 3.32, yang terdiri atas beberapa modul yaitu :

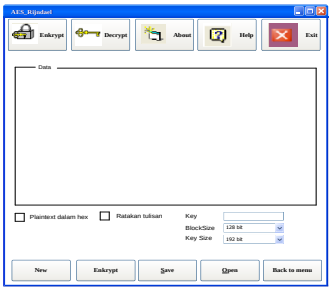


Gambar 3.32 : Rancangan Antar Muka

3.5.1 Rancangan Modul Enkripsi

Pada rancangan ini terdapat *frame message* untuk menampung pesan yang akan diubah dalam proses enkripsi. Pada modul ini terdapat pada *toolbar* - *toolbar* yang terletak pada bagian atas program aplikasi. *Toolbar* adalah daftar tombol pembantu yang dapat digunakan untuk mengaktifkan fungsi aplikasi. Tombol – tombol yang akan muncul bila anda menekan *toolbar Encrypt* adalah tombol New, tombol Proses encrypt, tombol Open, tombol Save, dan tombol Back to menu.

Jika anda menekan tombol new maka pada layar terlihat *frame message* yang kosong dan siap untuk diisi, bila *frame message* terdapat kata atau tulisan maka tombol new akan berubah menjadi clear. Tombol clear memiliki fungsi untuk membersihkan *frame message*. Pada tombol Proses encrypt memiliki fungsi untuk memulai pengacakan informasi, dan anda akan diminta untuk mengisikan kunci yang telah disepakati, setelah selesai mengisi kunci program aplikasi akan memproses pesan anda. Bila anda ingin menyimpan data atau informasi yang telah anda rahasiakan maka tekan tombol save. Setelah anda menyimpan data atau informasi anda dapat membuka file yang telah anda simpan sebelumnya dengan menekan tombol open. Lihat gambar 3.33 di bawah ini :

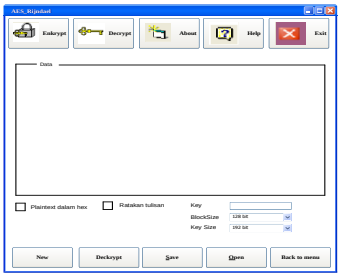


Gambar 3.33 : Rancangan Modul Enkripsi

3.5.2 Rancangan Modul Dekripsi

Pada rancangan ini terdapat *frame message*, seperti pada modul enkripsi *toolbar* ini memiliki beberapa tombol, antara lain adalah sebagai berikut : tombol New, tombol Open, tombol Save, tombol Proses Decrypt, dan tombol Back to menu.

Jika anda menekan tombol new maka pada layar terlihat *frame message* yang kosong dan siap untuk diisi, bila *frame message* terdapat kata atau tulisan maka tombol new akan berubah menjadi clear. Tombol clear memiliki fungsi untuk membersihkan *frame message*. Pada tombol Proses decrypt memiliki fungsi untuk mengembalikan pesan atau informasi yang telah mengalami pengacakan, dan anda akan diminta untuk mengisikan kunci yang telah disepakati, setelah selesai mengisi kunci program aplikasi akan memproses data anda. Jika kunci yang anda masukan salah pesan yang ditampilkan akan tidak sesuai dengan yang anda inginkan, jika kunci tersebut benar maka pesan sebenarnya dapat dimengerti dan maksud pengirim dapat tersampaikan. Bila anda ingin menyimpan data atau informasi yang telah anda rahasiakan maka tekan tombol save. Setelah anda menyimpan data atau informasi anda dapat membuka file yang telah anda simpan sebelumnya dengan menekan tombol open. Lihat gambar 3.34 di bawah ini :



Gambar 3.34 : Rancangan Modul Dekripsi

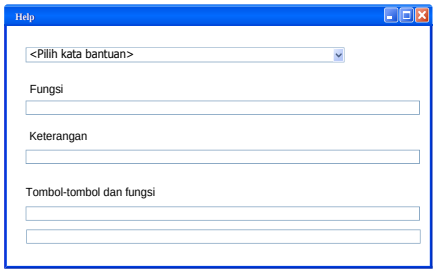
3.5.3 Rancangan Modul About

Pada modul about, pengguna dapat melihat keterangan singkat mengenai program kriptosistem dan pembuatnya. Modul ini akan ditampilkan pada saat pengguna menekan *toolbar* about pada menu utama yang terletak pada bagian atas program aplikasi. Lihat gambar 3.35 di bawah ini :

Gambar 3.35 : Rancangan Modul About

3.5.4 Rancangan Modul Help

Pada modul help, pengguna dapat melihat dan memilih keterangan mengenai cara menjalankan program dan keterangan mengenai fungsi –fungsi yang terdapat dalam program aplikasi kriptosistem tersebut. Modul ini akan ditampilkan bila pengguna menekan *toolbar* help pada menu utama yang terletak pada bagian atas program aplikasi. Lihat gambar 3.36 di bawah ini :



Gambar 3.36 : Rancangan Modul Help

4. PENGUJIAN DAN IMPLEMENTASI

4.1 Pengujian Modul Enkripsi

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

4.2 Pengujian 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 pesan tersebut. Fungsi modul dekripsi pada program ini berjalan sesuai dengan spesfikasi rancangan. Tampilan modul dekripsi dapat dilihat pada Lampiran Pengujian Modul Dekripsi.

4.3 Pengujian Modul About

Modul ini menampilkan penjelasan singkat mengenai program aplikasi da pembuat program aplikasi ini. Fungsi modul About pada program aplikasi ini berjalan sesuai dengan spesfikasi rancangan. Modul ini berjalan dengan baik, dapat dilihat pada Lampiran Pengujian Modul About.

4.4 Pengujian Modul Help

Modul Help dapat digunakan sebagai panduan untuk menjalankan program aplikasi ini. Fungsi modul ini berjalan sesuai dengan spesfikasi rancangan. Modul ini berjalan dengan baik, dapat dilihat pada Lampiran Pengujian Modul Help.

1. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil perancangan dan pembuatan program aplikasi kriptosistem menggunakan algoritma

*Advanced Encryption Standar*d (AES) ini, dapat diambil kesimpulan sebagai berikut :

a. Spesifikasi program aplikasi ini dapat dijalankan sesuai dengan spesifikasi teknis yang dirancang.
b. Program aplikasi kriptosistem ini akan membatasi orang yang tidak berhak atas informasi atau data yang dimiliki oleh si-pengirim untuk dibaca karena pesan sudah dienkripsi.
c. Program aplikasi kriptosistem ini menjaga kerahasiaan pesan atau informasi file-file yang ada dalam sebuah komputer.

5.2 Saran

Saran – saran yang berguna untuk pengembangan lebih lanjut terhadap program aplikasi ini adalah sebagai berikut :

a. Input untuk proses enkripsi tidak hanya dilakukan untuk format berbentuk text atau angka saja, tetapi bisa juga digunakan untuk mengenkripsi data yang berupa gambar (*image*), suara, video dan lain sebagainya.
b. Pengguna juga dapat memilih format penyimpanan data tidak hanya pada microsoft word (*.doc) saja, tetapi format lain juga dapat digunakan.

DAFTAR PUSTAKA

[ELI 1991] : Eli Biham, Adi Shamir, *Differential Cryptanalysis of The Full 16 – round. DES*, 1991.

[M. MATSUI 1993] : M. Matsui, *Linear Cryptanalysis Method for DES Cipher*. Abstracts of EUROCRYPT'93, 1993.

[ELI 1998] : Eli Biham, *Design Tradeoffs of The AES Candidates*. Oktober 1998.

[JOAN 1999] : Joan Daemen, Vincent Rijmen, *AES Proposal : Rijndael, Document Version 2*. NIST, 1999. http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf

[JOAN 2002] : Joan Daemen, Vincent Rijmen, *The Design of Rijndael : AES – The Advanced Encryption Standard*. Springer-Verlag, 2002.

[BRIAN 2003] : Brian Gladman, *A Specification for Rijndael, The AES Algorithm*. 2003.

[J. DAEMAN 2000] : Joan Daemen, Vincent Rijmen, *The Design of Rijndael : AES – The Advanced Encryption Standard*. Springer-Verlag, 2002.

[ABDUL 2003] : Abdul, Kerangka Dasar Sistem Informasi Manajemen. Pustaka Binaan Pressindo, Jakarta, 2003