BAB VI

ANALISIS KINERJA PERANGKAT LUNAK

6.1 Penanganan Kesalahan

Dalam tahap ini akan dijelaskan mengenai pengujian program aplikasi yang digunakan pada enkripsi dan dekripsi file maupun text. Dengan pengujian ini diharapkan tingkat kesalahan baik dalam pengenkripsian file maupun sistem itu sendiri menjadi sangat minim bahkan tidak ada.

Pengujian kinerja pada enkripsi/dekripsi file ini dilakukan untuk mengetahui kesalahan tersebut. Penanganan kesalahan pada enkripsi/dekripsi file ini dilakukan dengan memberikan dalam bentuk pesan peringatan yang berisikan informasi tentang keharusan untuk mengisikan data tertentu.

Dalam hal ini pengujian kinerja untuk mendeteksi kesalahan pada enkripsi/dekripsi file terdiri dari pengujian normal dan pengujian tidak normal.

6.2 Pengujian Normal

Pengujian normal dilakukan dengan memberikan masukan sesuai dengan data yang dibutuhkan, dimana data yang dimasukan harus benar.

6.2.1 Pengujian Normal Form Encrypt File

Pada form encrypt file masukkan data berupa key proses enkripsi, nama file yang akan dienkripsi, file untuk menyimpan hasil dari proses enkripsi, nama file

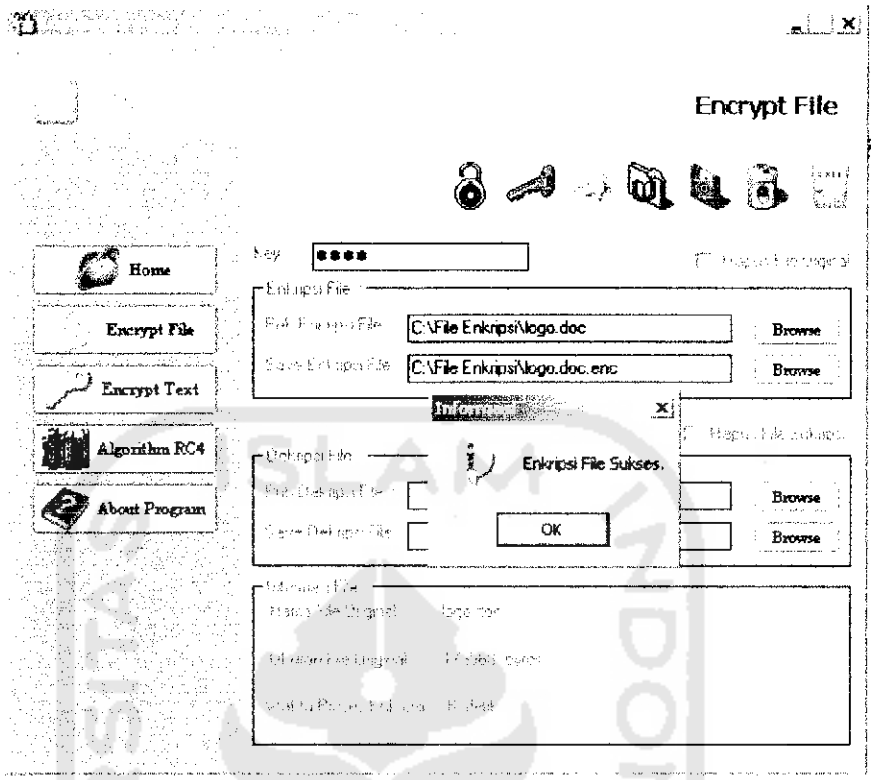
yang akan didekripsi dan file untuk menyimpan hasil dari proses dekripsi. Hasil outputnya dapat dilihat di informasi file berupa nama file yang akan dienkripsi/dekripsi, ukuran file dan waktu proses enkripsi/dekripsi file. Misal data yang akan dimasukkan untuk proses enkripsi file adalah:

- Key Proses Enkripsi : 1234
- File yang akan dienkripsi : C:\File Enkripsi\logo.doc
- Save Enkripsi File : C:\File Enkripsi\logo.doc.enc

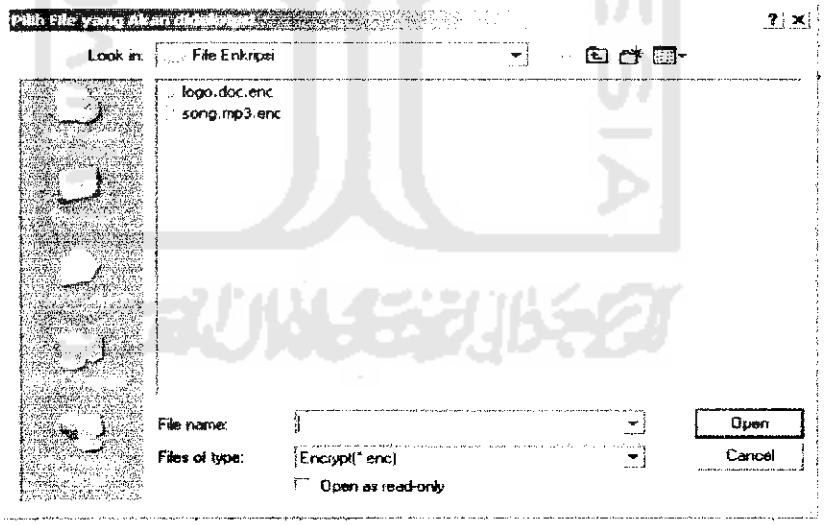
Untuk hasil dari proses enkripsi dapat dilihat pada informasi file yaitu:

- Nama File Original : logo.doc
- Ukuran File Original : 173568 bytes
- Waktu Proses Enkripsi : 6 detik

Dari hasil proses enkripsi dapat diketahui bahwa untuk file logo.doc dengan ukuran file sebesar 173,568 bytes waktu prosesnya yaitu 6 detik. Hasil proses enkripsi ini disimpan dalam file logo.doc.enc. Hasil dari proses enkripsi dengan data diatas dapat dilihat pada gambar 6.1, sedang untuk file yang telah terenkripsi dapat dilihat pada gambar 6.2



Gambar 6.1 Hasil Proses Enkripsi File



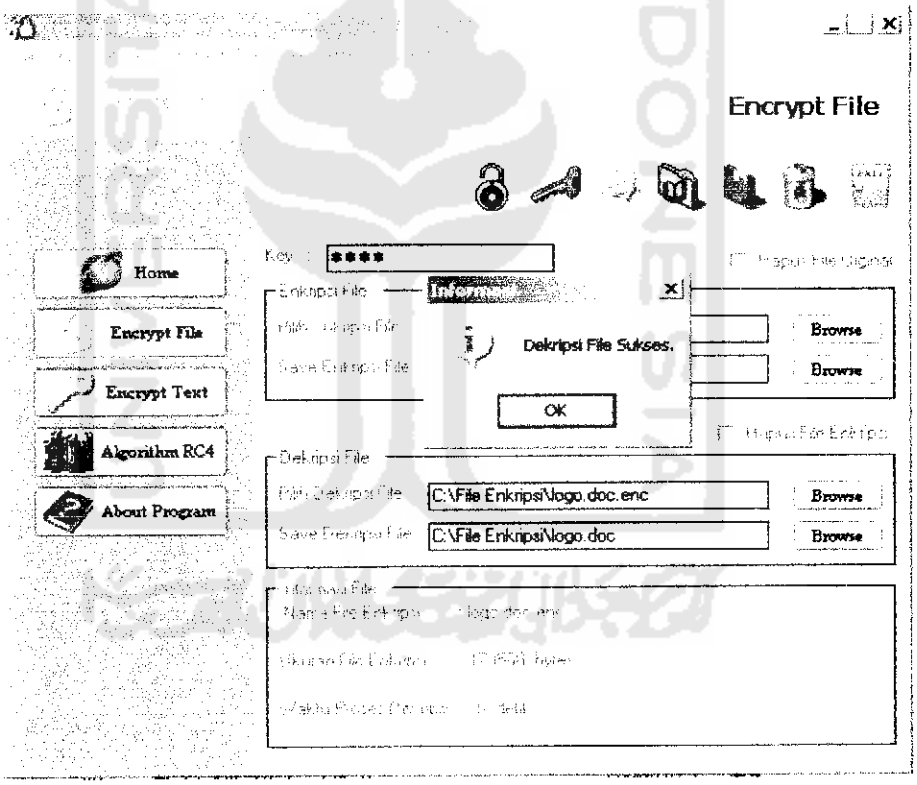
Gambar 6.2 File Hasil Proses Enkripsi

Untuk data yang akan dimasukkan untuk proses dekripsi adalah:

- Key Proses Dekripsi : 1234
- File yang akan didekripsi : C:\File Enkripsi\logo.doc.enc
- Save Dekripsi File : C:\File Enkripsi\logo.doc

Untuk hasil dari proses dekripsi dapat dilihat pada informasi file yaitu:

- Nama File Enkripsi : logo.doc.enc
- Ukuran File Enkripsi : 173568 bytes
- Waktu Proses Dekripsi : 6 detik



Gambar 6.3 Hasil Proses Dekripsi File

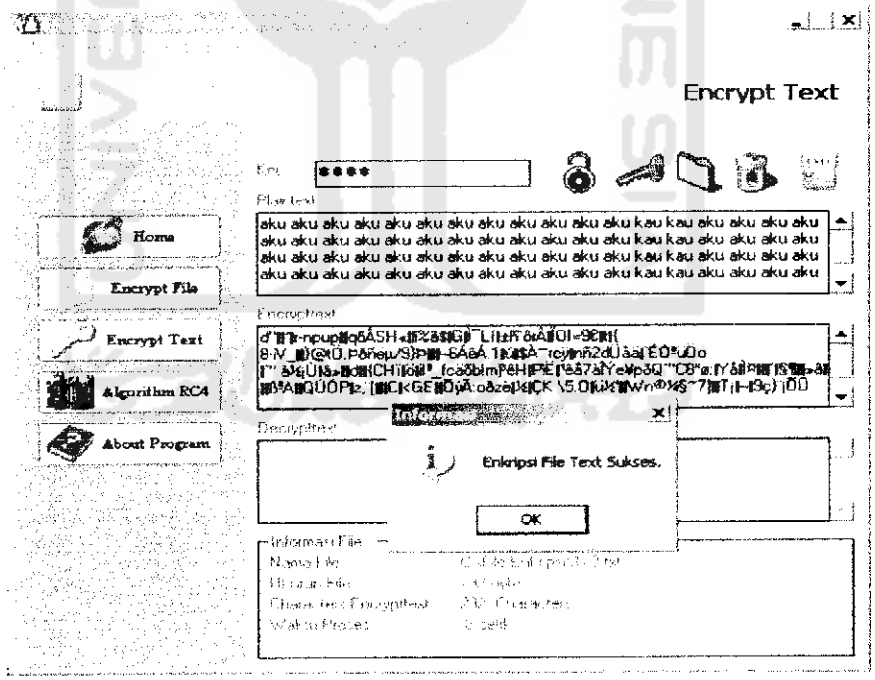
6.2.2 Pengujian Normal Form Encrypt Text

Pada form encrypt text masukkan datanya adalah key proses enkripsi dan plaintext yang akan dienkripsi. Informasi file berupa nama file yang akan dienkripsi , ukuran file, panjang karakter dan waktu proses. Misal data yang akan dimasukkan pada proses enkripsi text adalah :

- Key Proses Enkripsi : 1234
- Nama File Enkripsi : C:\File Enkripsi\123.txt
- Ukuran File : 292 bytes

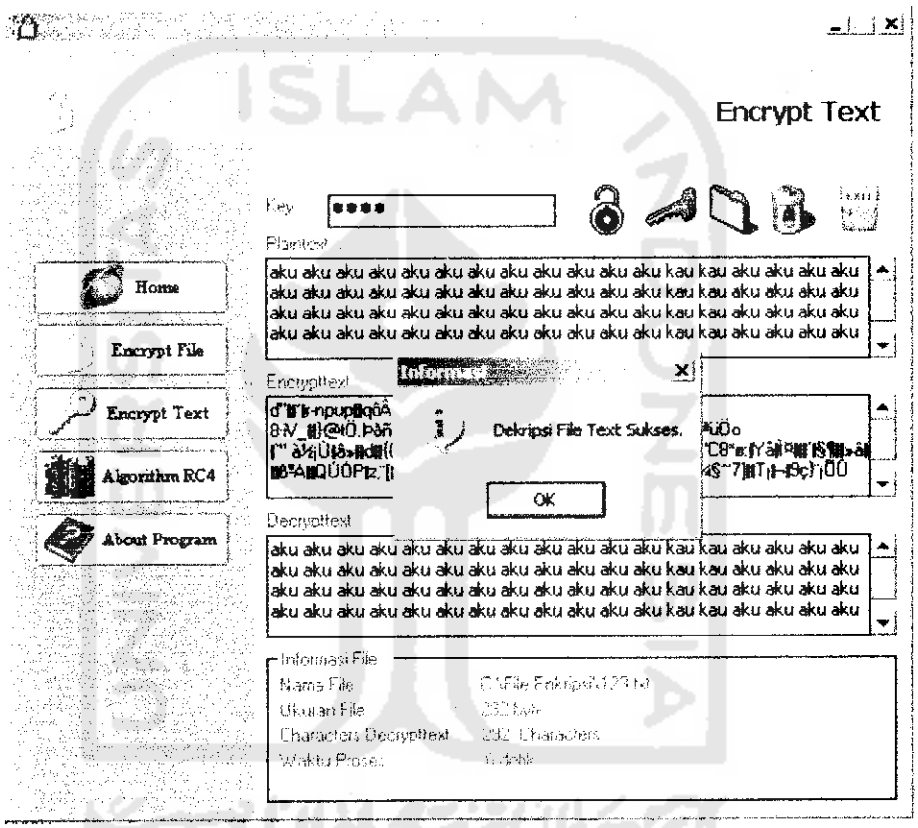
Untuk informasi file yang diperoleh adalah :

- Panjang Karakter Encryptext : 292 karakter
- Waktu Proses Enkripsi : 0 detik



Gambar 6.4 Hasil Proses Enkripsi Text

Hasil dari proses dekripsi text ditampilkan pada memo decrypttext dengan karakter seperti semula (plaintext). Untuk informasi file yang diperoleh yaitu nama file, ukuran file dan panjang karakter yang sama pada saat enkripsi serta waktu proses untuk dekripsi text yaitu 0 detik, hal ini dikarenakan file yang dienkripsi berukuran kecil.



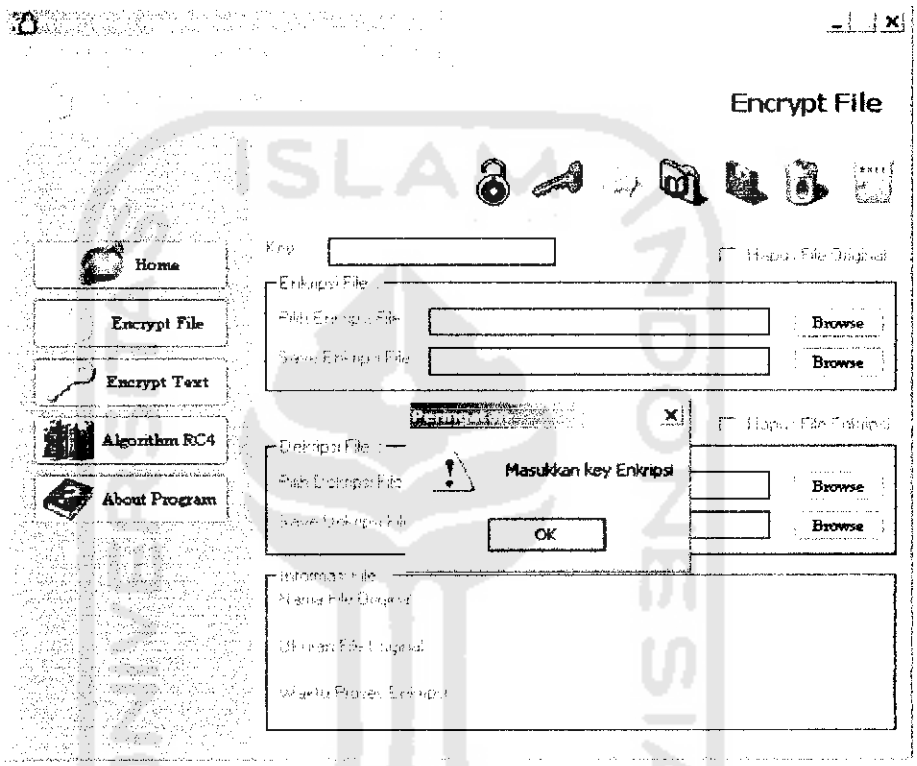
Gambar 6.5 Hasil Proses Dekripsi Text

6.3 Pengujian Tidak Normal

Pengujian tidak normal (*robust testing*) ini dilakukan untuk penanganan kesalahan input data dengan memberikan pesan peringatan kepada *user*.

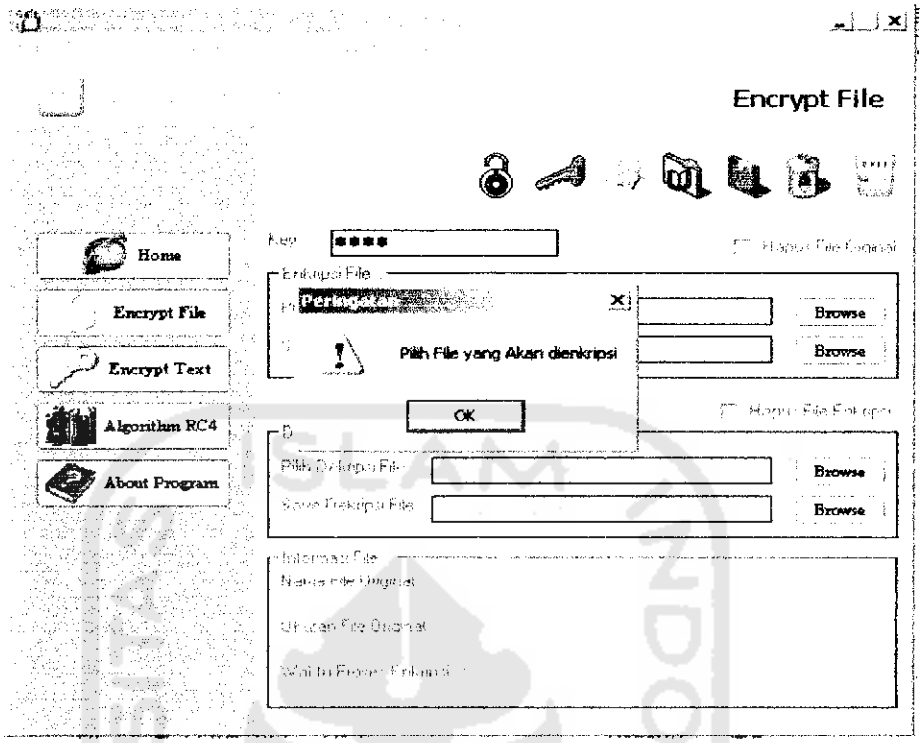
6.3.1 Pengujian Tidak Normal Form Encrypt File

Pada form encrypt file terdapat textbox untuk memasukkan key proses enkripsi, jika textbox ini tidak diisi maka akan menampilkan pesan peringatan untuk memasukkan key enkripsi. Untuk lebih jelasnya lihat gambar 6.6



Gambar 6.6 Pesan Peringatan Key Enkripsi Belum Terisi

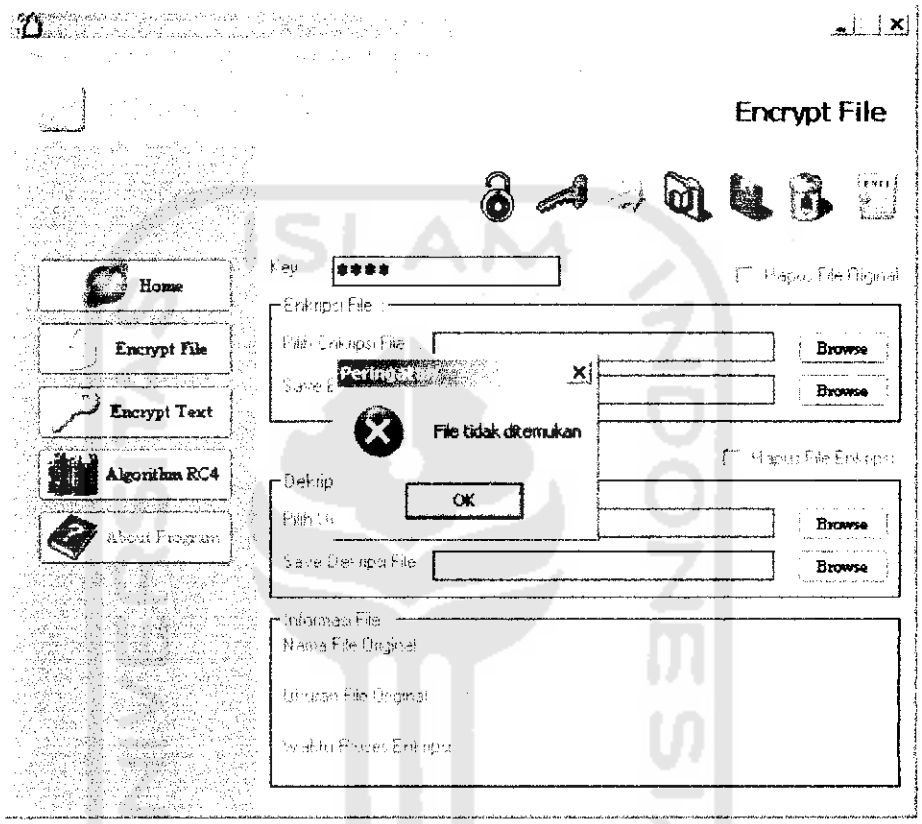
Selain key enkripsi yang harus diisi, nama file yang akan dienkripsi juga harus diisi hal ini untuk memproses file mana yang akan dienkripsi. Setelah nama file yang akan dienkripsi dipilih kemudian memilih file untuk menyimpan hasil dari proses enkripsi. Pesan peringatan jika nama file yang akan dienkripsi tidak diisi dapat dilihat pada gambar 6.7



Gambar 6.7 Pesan Peringatan Nama Enkripsi File Belum Terisi

Gamabr 6.8 Pilih Enkripsi File

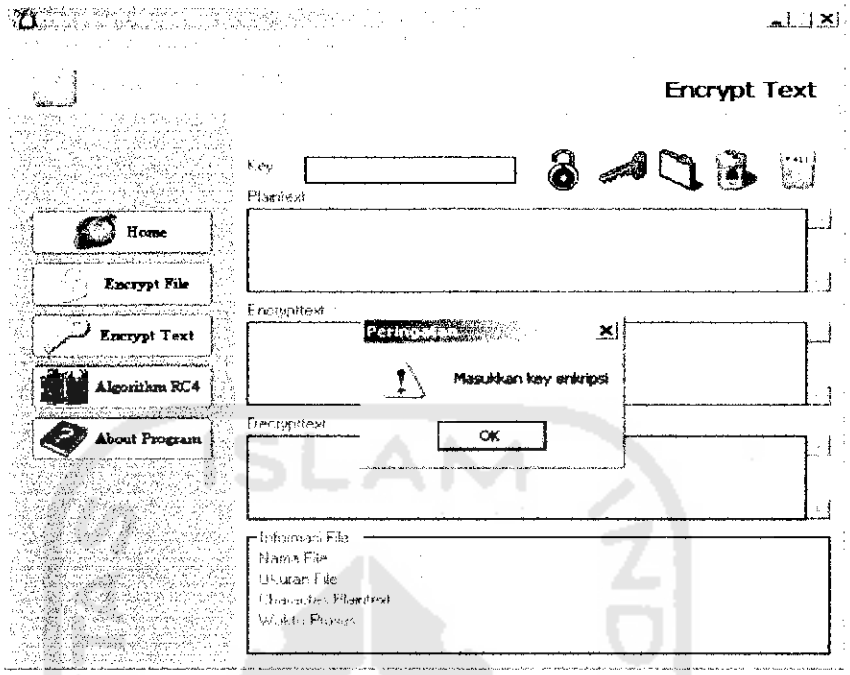
File yang akan dienkripsi harus ada dalam data elektronik atau komputer, jika file tersebut tidak ditemukan maka akan menampilkan peringatan bahwa file tidak ditemukan.



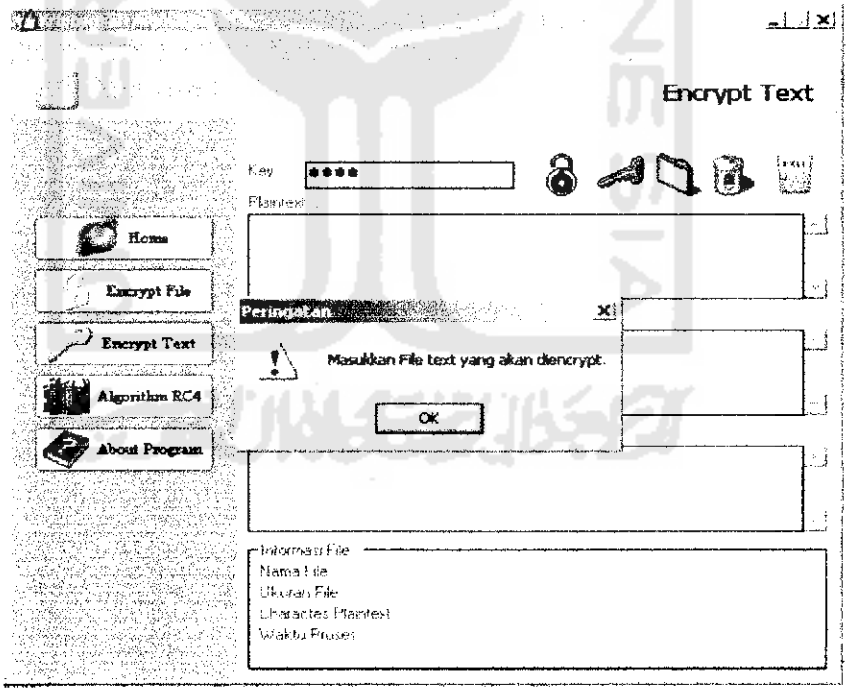
Gambar 6.9 Pesan Peringatan Enkripsi File Tidak Ditemukan

6.3.2 Pengujian Tidak Normal Form Encrypt Text

Pada form encrypt text key enkripsi harus diisi untuk proses enkripsi, kemudian memo plaintext juga harus diisi sebagai pesan yang akan dienkripsi. Untuk key dan memo plaintext yang tidak diisi maka akan menampilkan pesan peringatan. Lihat gambar 6.10 dan gambar 6.11



Gambar 6.10 Pesan Peringatan Key Enkripsi Text Belum Terisi



Gambar 6.11 Pesan Peringatan Plaintext Belum Terisi

6.4 Analisis Perbandingan

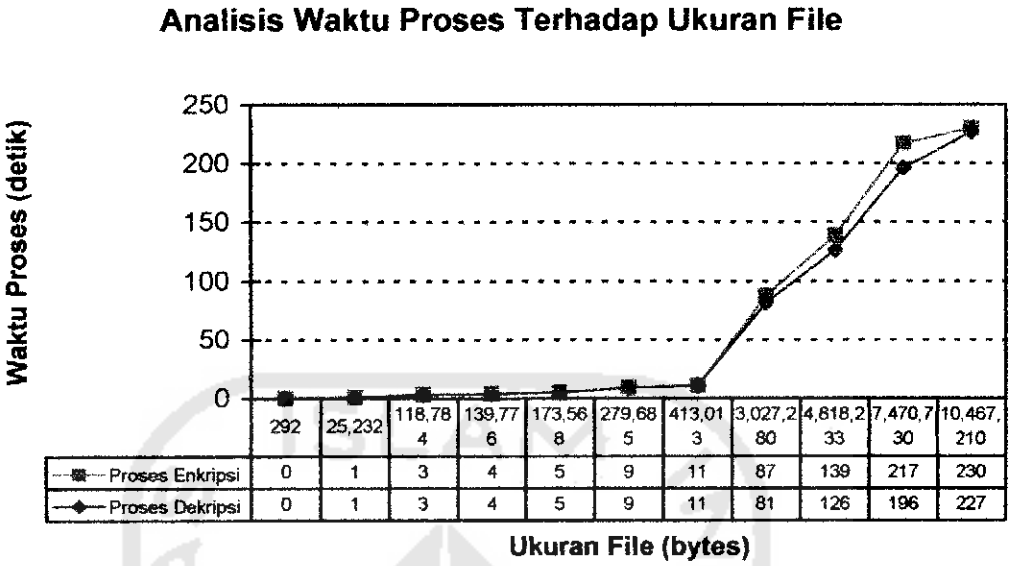
Analisis perbandingan digunakan untuk melihat kinerja dari algoritma RC4. Perbandingan disini berupa analisis waktu proses enkripsi dan dekripsi terhadap ukuran file kemudian perbandingan ukuran file antara file yang belum dienkripsi dengan setelah dienkripsi, analisis panjang kunci terhadap waktu proses. Serta analisis algoritma RC4 terhadap algoritma Blowfish.

6.4.1 Analisis Waktu Proses Terhadap Ukuran File

Waktu proses enkripsi/dekripsi file tergantung pada ukuran file yang akan dienkripsi/dekripsi. Pada uji coba proses enkripsi dengan menggunakan beberapa ekstensi file dengan ukuran yang berbeda-beda dapat disimpulkan bahwa semakin besar ukuran file maka waktu proses enkripsi/dekripsi file semakin lama. Hal ini disebabkan oleh efek cache dan efek penanganan file (*file handling*) oleh sistem operasi. Dapat dilihat pada tabel 6.1.

Tabel 6.1 Analisis Waktu Proses Enkripsi/Dekripsi Terhadap Ukuran File

Nama File	Ukuran File (bytes)	Waktu Proses Enkripsi (detik)	Waktu Proses Dekripsi (detik)
123.txt	292	0	0
Dia1.jpg	25,232	1	1
Process.mdb	118,784	3	3
Setup.exe	139,776	4	4
Logo2.doc	173,568	5	5
Sist.pakar.pdf	279,685	9	9
MateriUAS.zip	413,013	11	11
Song.mp3	3,027,280	87	81
RhapsodyBlue.mp3	4,818,233	139	126
Love for God.mp3	7,470,730	217	196



Gambar 6.12 Grafik Waktu Proses Terhadap Ukuran File

Tes dilakukan sebanyak tiga kali kemudian hasilnya dirata-ratakan. Untuk waktu proses enkripsi dan waktu proses dekripsi ukuran file yang kecil tidak mengalami perubahan, tetapi untuk file yang besar diatas 2 MB maka antara waktu proses enkripsi dan dekripsi mengalami perubahan. Dapat dilihat pada grafik 6.12 yaitu grafik mengalami perubahan ketika ukuran file sebesar 2 MB.

6.4.2 Analisis Ukuran File Terhadap Proses Enkripsi

Analisis dengan membandingkan ukuran file sebelum dienkripsi dan setelah file dienkripsi. Dapat dilihat pada gambar 6.1 dan gambar 6.3 bahwa ukuran file

logo.doc dengan ukuran file logo.doc.enc adalah sama. Untuk lebih jelasnya lihat tabel 6.2

Tabel 6.2 Analisis Ukuran File Terhadap Proses Enkripsi

Nama File	Ukuran File Original (bytes)	Ukuran File Enkripsi (bytes)
123.txt	292	292
Dia1.jpg	25,232	25,232
Process.mdb	118,784	118,784
Setup.exe	139,776	139,776
Logo2.doc	173,568	173,568
Sist.pakar.pdf	279,685	279,685
MateriUAS.zip	413,013	413,013
Song.mp3	3,027,280	3,027,280
RhapsodyBlue.mp3	4,818,233	4,818,233
Love for God.mp3	7,470,730	7,470,730

Dari tabel di atas dapat disimpulkan bahwa ukuran file sebelum proses enkripsi dengan ukuran file setelah proses enkripsi tidak mengalami perubahan karena algoritma RC4 tidak melakukan pengkompresan file. Untuk mengkompresi file diperlukan analisis dan algoritma tersendiri.

6.4.3 Analisis Proses Enkripsi Terhadap Panjang kunci

Pada tabel 6.3 ditunjukkan hasil analisis terhadap file yang sama dengan menggunakan ukuran panjang kunci yang berbeda-beda untuk mengetahui pengaruh panjang kunci terhadap kecepatan proses enkripsi.

Tabel 6.3 Analisis Proses Enkripsi Terhadap Panjang Kunci

Ukuran File (bytes)	Panjang Kunci (karakter)	Waktu Proses (detik)
118,784	1	4
118,784	3	4
118,784	20	4
118,784	50	4
118,784	100	4
118,784	256	4

Dari tabel diatas, terlihat bahwa dengan menggunakan file yang sama dengan ukuran kunci yang berbeda-beda dari 1 karakter sampai panjang kunci maksimal 256 karakter tidak mempengaruhi kecepatan proses enkripsi. Kecepatan proses enkripsi dan dekripsi tidak mempunyai hubungan dengan panjang kunci user. Panjang kunci user hanya memiliki hubungan dengan stream generator sehingga berapapun besar panjang kunci, kecepatan proses enkripsi dan dekripsi tidak terpengaruh. Hal ini disebabkan karena dalam RC4 proses enkripsi dan dekripsi terpisah dengan proses stream generator (proses untuk menghasilkan keystream). Keystream inilah yang kemudian digunakan dalam proses enkripsi dan dekripsi.

6.4.4 Analisis Perbandingan Algoritma RC4 dengan Algoritma Blowfish

Penelitian yang dilakukan oleh saudara Anton Nugroho yaitu aplikasi enkripsi file dengan menggunakan algoritma Blowfish yang berbentuk block cipher, sedang penelitian yang penulis lakukan adalah aplikasi enkripsi file dengan menggunakan

algoritma RC4 yang berbentuk stream cipher dimana dalam proses enkripsi dan dekripsi berbeda.

Untuk enkripsi dengan menggunakan algoritma blowfish merupakan blok cipher 64-bit dengan panjang kunci variabel. Algoritma ini terdiri dari dua bagian: *key expansion* dan enkripsi data. *Key expansion* merubah kunci yang dapat mencapai 448 bit menjadi beberapa array subkunci (*subkey*) dengan total 4168 byte.[ANT04]

Untuk algoritma yang berbentuk block cipher seperti Blowfish hasil dari proses ciphertextnya berpola, jika ada teks yang berulang maka hasil ciphertextnya sama sedang untuk algoritma yang berbentuk stream cipher seperti RC4 hasil dari ciphertextnya terlihat random dimana untuk teks yang berulang hasilnya tidak sama dengan teks yang sebelumnya karena setiap karakter dari plaintext di XORkan dengan *keystream* yang berbeda.

6.5 Keamanan Algoritma RC4

Dalam algoritma enkripsi, panjang kunci yang biasanya dalam ukuran bit, juga menentukan kekuatan dari enkripsi. Kunci yang lebih panjang biasanya lebih aman daripada kunci yang pendek. Jadi enkripsi dengan menggunakan kunci 128-bit lebih sukar dipecahkan dengan algoritma enkripsi yang sama tetapi memiliki kunci 56-bit. Semakin panjang sebuah kunci, semakin besar keyspace yang harus dijalani untuk mencari kunci dengan cara *brute force attack* atau coba-coba karena keyspace yang harus dilihat merupakan pangkat bilangan dari 2. Jadi kunci 128-bit memiliki

keyspace 2^{128}, sedangkan kunci 56-bit memiliki keyspace 2^{56}. Artinya semakin lama kunci baru bisa ditemukan.[AND03].

Pada intinya, keamanan suatu pesan tidak tergantung pada algoritma RC4 sendiri tetapi pada kunci yang digunakan sebagai input ke RC4. Untuk enkripsi algoritma RC4 terdapat beberapa kelebihan maupun kelemahan masing-masing. Adapun kelebihan dan kelemahan pada sistem yang penulis buat dengan menggunakan algoritma RC4 ini antara lain:

- Kelebihan :
  1. Kesulitan mengetahui sebuah nilai dalam tabel
  2. Kesulitan mengetahui lokasi mana di dalam tabel yang digunakan untuk menyeleksi masing-masing nilai.
  3. Penghapusan file sumber dimana selain dienkripsi file juga dihapus untuk memperkuat proses enkripsi.
  4. Hasil dari ciphertextnya random dimana untuk plaintext yang berulang hasil cipertextnya berbeda hal ini karena setiap karakter pada plaintext di XORkan dengan keystream yang berbeda.

- Kekurangan :
  1. Terlalu tingginya kemungkinan terjadi tabel S-box yang sama, hal ini terjadi karena kunci user diulang-ulang untuk mengisi 256 bytes, sehingga 'aaaa' dan 'aaaaa' akan menghasilkan permutasi yang sama.

2. Untuk proses enkripsi file dimana untuk file dengan kapasitas diatas 2 MB akan memakan waktu proses yang cukup lama
3. Kelemahan pada algoritma RC4 sebenarnya kelemahan umum yang ada pada model enkripsi simetris atau kunci pribadi. Kelemahan ini timbul jika terdapat banyak orang yang ingin saling berkomunikasi, karena setiap pasangan maupun file enkripsi mempunyai key berbeda yang harus disepakati sehingga key tiap orang maupun file harus menghafal banyak kunci dan menggunakannya dengan tepat.