ANALISIS PERBANDINGAN ALGORITMA SIMETRIS RIVEST CODE 5 DENGAN ALGORITMA SIMETRIS RIVEST CODE 6)
(Studi Kasus : SMK Negeri Seluma)

Rozali Toyib[1], Ardi Wijaya,[2]
[1)2)]Program Studi Informatika, Fakultas Teknik, Universitas Muhammadiyah Bengkulu
Jl. Bali PO BOX 118. Telp (0736) 227665, Fax (0736) 26161, Bengkulu 38119
E-Mail : rozalitoyib@umb.ac.id[1], ardiwijaya@umb.ac.id[2]

Abstack: Data stored in storage media is often lost or opened by certain parties who are not responsible, so that it is very detrimental to the owner of the data, it is necessary to secure data so that the data can be locked so that it cannot be opened by irresponsible parties. The RC5 and RC6 algorithms are digestive massage algorithms or sometimes also known as the hash function which is an algorithm whose input is a message whose length is not certain, and produces an output message digest from its input message with exactly 128 bits in length. RC6 password is a protection for the user in securing data on a PC or computer. Based on the results of the conclusions taken: For the experiments carried out on the RC5 algorithm the execution time for the generation of keys (set-up key) is very fast, which is about 9-10 ns, a trial carried out on the RC6 algorithm execution time for the key generator (set up key) faster than 10-11 ns. In the encryption and decryption process, the execution time depends on the size or size of the plaintext file. The larger the size of the plaintext file, the longer the execution time.

Abstrak : Data yang tersimpan dalam media penyimpanan sering hilang atau dibuka oleh pihak-pihak tertentu yang tidak bertanggung jawab, sehinga merugikan sekali bagi pemilik data tersebut, untuk itu diperlukan suatu pengamanan data agar data tersebut dapat terkunci sehingga tidak dapat dibuka oleh pihak yang tidak bertanggung jawab.. Algoritma RC5 dan RC6 merupakan algoritma massage digest atau kadang juga dikenal dengan hash function yaitu suatu algoritma yang inputnya berupa sebuah pesan yang panjangnya tidak tertentu, dan menghasilkan keluaran sebuah message digest dari pesan inputnya dengan panjang tepat 128 bit. Password RC6 merupakan salah satu perlindungan kepada user dalam pengamanan data yang berada dalam sebuah Pc atau computer. Berdasarkan hasil pengujian diambil kesimpulan : Untuk uji coba yang dilakukan pada algoritma RC5 waktu eksekusi untuk pembangkitan kunci (set up key) sangat cepat sekali yaitu sekitar 9-10 ns, uji coba yang dilakukan pada algoritma RC6 waktu eksekusi untuk pembangkit kunci (set up key) lebih cepat sekali yaitu 10-11 ns, Pada proses enkripsi dan dekripsi, waktu eksekusi tergantung dari besar atau kecilnya ukuran file plaintext.s emakin besar ukuran file plaintext maka semakin lama waktu eksekusinya.

Keyword : simetris, rivest

I. PENDAHULUAN

Dokumentasi dalam bentuk file baik itu berektensionkan .doc, xls. Ppt dll yang tersimpan kedalam media penyimpanan biasanya disimpan kedalam suatu folder atau direktori sehingga dapat mudah untuk mencari atau pengelompokkanya. Disamping itu juga data yang tersimpan dalam media penyimpanan sering hilang atau dibuka oleh pihak-pihak tertentu yang tidak bertanggung jawab, sehinga merugikan sekali bagi pemilik data tersebut, untuk itu diperlukan suatu pengamanan data dalam bentuk folder. Tujuannya adalah untuk mengamanan data agar data tersebut dapat terkunci sehingga tidak dapat dibuka oleh pihak yang tidak bertanggung jawab.

Penerapan algoritma RC5 dan RC6 telah digunakan dalam berbagai jenis aplikasi keamanan, dan juga sering digunakan untuk memeriksa integritas file sehingga algoritma RC5 dan RC6 dapat menjaga keamanan data dari pengguna-pengguna (user) yang tidak bertanggung jawab.

Algoritma RC5 dan RC6 merupakan algoritma massage digest atau kadang juga dikenal dengan hash function yaitu suatu algoritma yang inputnya berupa sebuah pesan yang panjangnya tidak tertentu, dan menghasilkan keluaran sebuah message digest dari pesan inputnya dengan panjang tepat 128 bit. Password RC6 merupakan salah satu perlindungan kepada user dalam pengamanan data yang berada dalam sebuah Pc atau komputer karena sebuah password adalah kunci yang sangat berharga bagi kita yang sering melakukan aktifitas yang berhubungan dengan perkantoran atau instasi tertentu.

Dari penelitian yang dilakukan sebelumnya banyak membahas tenang satu macam algoritma saja, maka pada penelitian ini penulis akan melakukan penelitian dengan membandingkan dua algoritma pengamanan data yaitu RC5 dan RC 6.

II. TINJAUAN PUSTAKA

2.1. Kriptografi

Kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Dalam ilmu

kriptografi, terdapat dua buah proses yaitu melakukan enkripsi dan dekripsi. Pesan yang akan dienkripsi disebut sebagai plaintext (teks biasa). Disebut demikian karena informasi ini dengan mudah dapat dibaca dan dipahami oleh siapa saja. Algoritma yang dipakai untuk mengenkripsi dan mendekripsi sebuah plaintext melibatkan penggunaan suatu bentuk kunci. Pesan plaintext yang telah dienkripsi (atau dikodekan) dikenal sebagai ciphertext (teks sandi). Di dalam kriptografi kita akan sering menemukan berbagai istilah atau terminology. Beberapa istilah yang harus diketahui yaitu : 1. Pesan, Plainteks, dan Cipherteks Pesan (message) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah (plaintext) atau teks jelas (cleartext). 2. Pengirim dan Penerima Komunikasi data melibatkan pertukaran pesan antara dua entitas. Pengirim (sender) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (receiver) adalah entitas yang menerima pesan. 3. Enkripsi dan dekripsi Proses menyandikan plainteks menjadi cipherteks disebut enkripsi (encryption) atau enciphering (standard nama menurut ISO 7498-2). Sedangkan proses mengembalikan cipherteks menjadi plainteks semula disebut dekripsi (decryption) atau deciphering (standard nama menurut ISO 7498-2). 4. Cipher dan kunci Algoritma kriptogarfi disebut juga cipher, yaitu aturan untuk enkripsi dan dekripsi, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa cipher memerlukan algoritma yang berbeda untuk enkripsi dan dekripsi. Konsep matematis yang mendasari algoritma kriptografi adalah relasi antara dua buah himpunan yang berisi elemen-elemen plainteks dan himpunan yang berisi cipherteks [1].

Cryptography adalah cabang ilmu matematika tentang persandian untuk menjaga keamanan data. Cryptographic system atau cryptosystem adalah suatu fasilitas untuk mengkonversikan plaintext ke ciphertext dan sebaliknya. Plaintext adalah data asli, data yang masih bisa dibaca dan dimengerti. Sedangkan ciphertext adalah data yang tidak bisa dibaca maupun dimengerti. [2].

2.2. Algoritma Simetris

Algoritma simetrik dapat pula disebut sebagai algoritma konvensional, dimana kunci dekripsi dapat ditentukan dari kunci enkripsinya, begitu pula sebaliknya. Pada algoritma simetrik, kunci enkripsi dan kunci dekripsinya sama. Keamanan dari algoritma ini terletak pada kuncinya, jika kunci diberitahukan atau dibocorkan maka siapa saja dapat mengenkrip dan mendekrip data, jadi kunci harus benar-benar rahasia dan aman [3]

Algoritma simetri atau disebut juga algoritma kriptografi konvensional adalah algoritma yang menggunakan kunci yang sama untuk proses enkripsi dan proses dekripsi. Contoh algoritma kunci simetris adalah DES (Data Encryption Standard), blowfish, twofish, MARS, IDEA, 3DES (DES diaplikasikan 3 kali), AES (Advanced Encryption Standard) yang bernama asli Rijndael [4].

Symmetric cryptosystem atau kriptografi simteris atau disebut juga algoritma kriptografi konvensional adalah algoritma yang menggunakan kunci untuk proses enkripsi sama dengan kunci untuk proses dekripsi. Algoritma kriptografi simetris dibagi menjadi 2 kategori yaitu algoritma aliran (Stream Ciphers) dan algoritma blok (Block Ciphers). Pada algoritma aliran, proses penyandiannya berorientasi pada satu bit atau satu byte data. Sedang pada algoritma blok, proses penyandiannya berorientasi pada sekumpulan bit atau byte data (per blok) [5].

2.2.1 Algoritma RC-5

Algoritma Rivest Code-5 merupakan metode enkripsi menggunakan metode simetrik dan pengolahan dalam bentuk blok chiper, jadi kata kunci yang sama digunakan untuk proses enkripsi dan dekripsi. Parameter-parameter yang digunakan dalam Rivest Code-5 adalah sebagai berikut :
Kata kunci (key word) Variabel ini disimbolkan dengan b dengan range 0, 1, 2, ….255. Key word ini dikembangkan menjadi array S yang digunakan sebagai key pada proses untuk enkripsi dan dekripsi [6]

2.2. 2 Algoritma RC-6

RC6 merupakan algoritma cipher blok baru yang didaftarkan ke NIST yang diajukan oleh RSA Security Laboratories. Algoritma ini dirancang oleh Ronald L Rivest, M.J.B. Robshaw, R. Sidney dan Y.L. Yin untuk mengikuti kontes Advanced Ecryption Standard (AES) dan berhasil menjadi salah satu dari lima (5) finalisnya. Design dari berawal dari keinginan untuk meningkatkan performansi dan tingkat keamanan dari RC5 untuk dapat memenuhi standar dari kontes tersebut. RC6 memiliki struktur yang sederhana. RC6 terdiri dari dua jaringan Feistel dimana datanya dicampur dengan rotasi yang bergantung pada isi data tersebut. Dalam sekali putaran RC6, ada beberapa operasi yang terjadi, antara lain : dua (2) aplikasi dari fungsi persamaan $f(x) = x(2x + 1) \bmod 232$, dua (2) rotasi 32-bit yang tidak berubah, dua (2) rotasi 32-bit yang bergantung pada data, dua (2) eksklusif OR dan dua (2) fungsi modulo 232 tambahan. Algoritma cipher ini biasanya memakai 20 putaran [7].

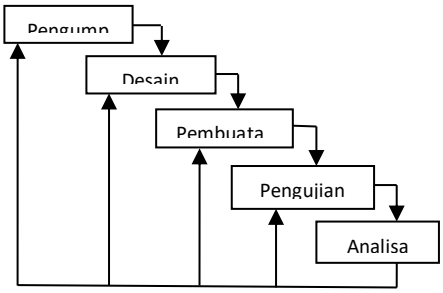
Algoritma RC6 merupakan salah satu kandidat Advanced Encryption Standard (AES) yang diajukan oleh RSA Security Laboratories kepada NIST. Dirancang oleh Ronald L Rivest, M.J.B. Robshaw, R. Sidney dan Y.L. Yin, algoritma ini merupakan pengembangan dari algoritma sebelumnya yaitu RC5 dan telah memenuhi semua kriteria yang diajukan oleh NIST. RC6 adalah algoritma yang menggunakan ukuran blok hingga 128 bit, dengan ukuran kunci yang diguankan bervariasi antara 128, 192 dan 256 bit. Algoritma RC6 dilengkapi dengan beberapa parameter,

sehingga dituliskan sebagai RC6-w/r/b. Parameter w merupakan ukuran kata dalam satuan bit, parameter merupakan bilangan bukan negatif yang menunjukan banyaknya iterasi selama proses enkripsi dan parameter b menunjukan ukuran kunci enkripsi dalam byte. Setelah algoritma ini masuk dalam kandidat AES, maka ditetapkan bahwa nilai w = 32, r=20 dan b bervariasi antara 16, 24 dan 32 byte [8].

III. METODE PENELITIAN

3.1 Metode Pengembangan Sistem

Adapun sketsa tahap analisa pemrosesan adalah seperti gambar berikut :



Gambar 3.1 Sketsa Pemrosesan Data Model *Waterfal*

Keterangan:

1. Pengumpulan data
   Pada tahap Pengumpulan data dalam penelitian dimaksudkan gunamendapatkan bahan, keterangan, kenyataan, dan informasi yang bisa dipercaya. pengumpulan data merupakan tahap yang diperlukan sebelum merancang atau desain program.
2. Desain Program
   Setelah data-data yang diperlukan terkumpul langkah selanjutya adalah perancangan atau mendesain program sebelum program tersebut dibuat.
3. Pembuatan Program
   Setelah selesai mendesain langkah selanjutnya adalah pembutan program, dimana program tersebut di buat sesuai dengan desain yang telah dirancang sesuai dengan kebutuhan.
4. Pengujian
   Pada tahap pengujian sistem ini dilaksanakan apakah sistem sudah siap digunakan, Pengujian ini dilakukan pada setiap pembangunan, yaitu : Pengujian dilakukan dengan prosedur White-box dan Black-box.
5. Analisa
   Tahap ini merupakan tahapan menganalisa, yaitu apakah sistem yang digunakan membantu pakar atau sesuai dengan kebutuhan.

3.2 Analisis

Cara kerja algoritma RC6 yaitu inisialisasi S-Box pertama, S[0], S[1], …….., S[255], dengan bilangan 0 sampai 255. Pertama isi secara berurutan S[0]=0, S[1]=1, …., S[255]=255. Kemudian inisialisasi *array* lain (S-Box lain), misal *array* K dengan panjang 256. Isi *array* K dengan kunci diulangi sampai seluruh *array* K[0], K[1], ……., K[255] terisi seluruhnya. Proses inisialisasi S-Box (*Array* S) dapat dilihat pada tabel 3. 1 dibawah ini :

Tabel 3.1 Proses Inisialisasi S-Box (*Array* S)

Proses Inisialisasi S-Box(*Array* S)
for i = 0 to 255 S[i] = i

Proses inisialisasi S-Box (*Array* K) dapat dilihat pada tabel 3. 2 berikut:

Tabel 3.2 Proses Inisialisasi S-Box (*Array* K)

Proses Inisialisasi S-Box (*Array* K)
array Kunci // *Array* dengan panjang kunci “length”. for i = 0 to 255 K[i] = Kunci [i mod length]

Kemudian lakukan langkah pengacakan S-Box yang dapat dilihat pada tabel 3.3 sebagai berikut :

Tabel 3.3 Proses Pengacakan S-Box

Proses Pengacakan S-Box

i = 0; j = 0;
for i = 0 to 255
{
j = (j + S[i] + K[i]) mod 255
Swap S[i] dan S[j]
}

Setelah itu, buat *pseudo random byte* pada tabel 3. 4 dengan langkah sebagai berikut:

Tabel 3.4 *Pseudo RandomByte*

Pseudo RandomByte
i = (i + 1) mod 255 j = (j + S[i] mod 255 swap S[i] dan S[j] t = (S[i] + S[j] mod 255 K = S[t]

Byte K di-XOR kan dengan *plaintext* untuk menghasilkan *ciphertext* atau di XOR kan dengan *ciphertext* untuk menghasilkan *plaintext*. Enkripsi sangat cepat kurang lebih 10 kali lebih cepat dari DES.

Berikut adalah contoh penerapan algoritma RC6 dengan mode 4 *byte* (agar lebih sederhana). Pertama inisialisasi S-Box dengan panjang 4 *byte*, dengan S[0]=0, S[1]=1, S[2]=2, dan S[3]=3 sehingga *array* S menjadi :

0 1 2 3

Inisialisasi 4 *byte* kunci *array*, Ki. Misalkan kunci terdiri dari 2 *byte* yaitu *byte* 1 dan *byte* 7. Ulang kunci sampai memenuhi seluruh *array* K sehingga *array* K menjadi :

1 7 1 7

Berikutnya mencampur operasi dimana kita akan menggunakan variabel i dan j ke index *array* S[i] dan K[i]. Pertama kita beri nilai inisial untuk i dan j dengan 0. Operasi pencampuran adalah pengulangan rumusan (j + S[i] + K[i] mod 4 yang diikuti dengan penukaran S[i] dengan S[j]. Karena menggunakan *array* dengan panjang 4 *byte* maka algoritma menjadi :

For i = 0 to 4
j = (j + S[i] + K[i] mod 4
swap S[i] dan S[j]

Dengan algoritma seperti diatas maka nilai awal i=0 sampai i=3 akan menghasilkan *array* S seperti berikut :

Iterasi pertama :
i = 0, maka
j = (j + S[i] + K[i]) mod 4
= (j + S[0] + K[0]) mod 4
= (0 + 0 + 1) mod 4
= 1
Swap S[0] dan S[1] sehingga menghasilkan *array* S :
10 2 3

Iterasi kedua :
i = 1, maka
j = (j + S[i] + K[i]) mod 4
= (j + S[1] + K[1]) mod 4
= (1 + 0 + 7) mod 4 = 0
Swap S[1] dan S[0] sehingga menghasilkan *array* S :
0 1 2 3

Iterasi ketiga :
i = 2, maka
j = (j + S[i] + K[i]) mod 4
= (j + S[2] + K[2]) mod 4
= (0 + 2 + 1) mod 4
= 3
Swap S[2] dan S[3] sehingga menghasilkan *array* S :
0 1 3 2

Iterasi keempat :
i = 3, maka
j = (j + S[i] + K[i]) mod 4
= (j + S[3] + K [3]) mod 4
= (3 + 2 + 7) mod 4
= 0
Swap S[3] dan S[0] sehingga menghasilkan *array* S :
2 1 3 0

Setelah didapat hasil *array* S dari iterasi keempat, maka proses selanjutnya yaitu meng-XOR-kan *pseudo randombyte* dengan *plaintext*, misalnya *plaintext* yang dimasukkan adalah "HI".

Karena *plaintext* terdiri dari dua karakter maka terjadi dua iterasi. Iterasi pertama yaitu :

Inisialisasi i dan j dengan i = 0; j = 0.
i = 0; j = 0;
i = (i + 1) mod 4
= (0 + 1) mod 4
= 1
Dan
j = (j + S[i]) mod 4
= (0 + 2) mod 4
= 2
Swap S[i] dan S[j] yaitu S[1] dan S[2] sehingga *array* S menjadi :
2 3 1 0
t = (S[i] + S[j]) mod 4
= (3 + 1) mod 4
= 0
K = S[t] = S[0] = 2

Byte dua/K inilah yang di-XOR-kan dengan *plaintext* "H". Selanjutnya iterasi keduanya yaitu :

i = 1; j = 2
i = (i + 1) mod 4
= (1 + 1) mod 4
= 2
Dan
j = (j + S[i]) mod 4
= (2 + 2) mod 4 = 0
Swap S[i] dan S[j] yaitu S[2] dan S[0] sehingga *array* S menjadi :
1 3 2 0
t = (S[i] + S[j]) mod 4
= (2 +1) mod 4
= 3
K = S[t] = S [3] = 2

Byte K=2 yang akan di-XOR-kan dengan *plaintext* "I"

Proses XOR *pseudo randombyte* dengan *plaintext*, dapat dilihat pada tabel 3.5 dibawah ini :

Tabel 3.5 Proses *XOR Pseudo Random Byte* dengan *plaintext* pada Enkripsi.

	"H"	"I"
Plaintext	0 1 0 0 1 0 0 0	0 1 0 0 1 0 0 1
Pseudo Random Byte	0 0 0 0 0 0 1 0	0 0 0 0 0 0 1 0
Ciphertext	0 1 0 0 1 0 1 0	0 1 0 0 1 0 1 1

Sedangkan proses deskripsi adalah kebalikan dari proses enkripsi, yaitu mengubah *ciphertext* menjadi *plaintext* kembali. Untuk lebih jelas dapat dilihat pada tabel 3.6 dibawah ini :

Tabel 3.6 Proses *XOR Pseudo Random Byte* dengan *plaintext* pada Deskripsi.

	"H"	"I"
Ciphertext	0 1 0 0 1 0 1 0	0 1 0 0 1 0 1 1
Pseudo Random Byte	0 0 0 0 0 0 1 0	0 0 0 0 0 0 1 0
Plaintext	0 1 0 0 1 0 0 0	0 1 0 0 1 0 0 1

Berdasarkan hasil pengujian performa RC6 pada Intel E4500 2.2GHz memori 1 GB, diperoleh hasil seperti terlihat pada tabel. Hasil pengetesan pada tabel 1.7 didapat dengan enkripsi 4 *kbyte* sebanyak 3200 kali, atau setara dengan 100 Mb data. Hasil pengetesan tersebut dapat dilihat pada tabel 3.7 di bawah ini :

Tabel 3.7 Kecepatan Enkripsi RC6

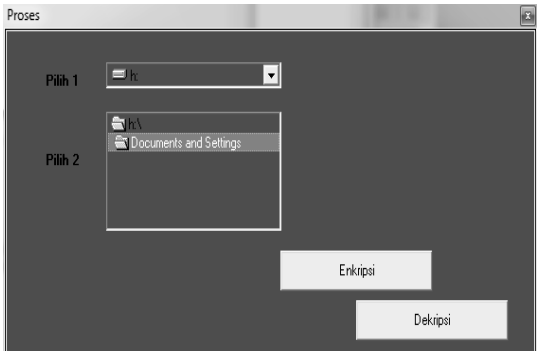
Jml Thread	Kecepatan Thread ke (dalam Mbps)					
	1	2	3	4	5	Total
1	91.42					91.42
2	87.67	86.48				174.15
3	58.71	61.53	58.71			178.95
4	41.29	56.63	48.48	50.39		196.79
5	36.99	37.42	43.53	41.02	36.36	195.32

IV. HASIL DAN PEMBAHASAN

4.1 Menu dalam Proses Pada Enkripsi dan Dekripsi

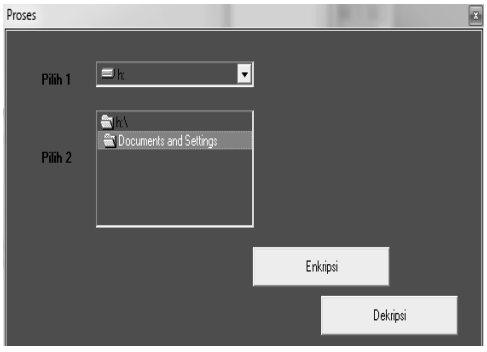
4.1.1 Proses Enkripsi

Form Enkripsi dan Dekripsi, terdapat tampilan Pilih Folder Data, Pilih Data Teks, Hasil dan Pilih Proses.



Gambar 4.1 Halaman Utama

Untuk menjelaskan bagaimana langkah dari proses ini, penulis akan memberikan contoh untuk enkripsi Data.



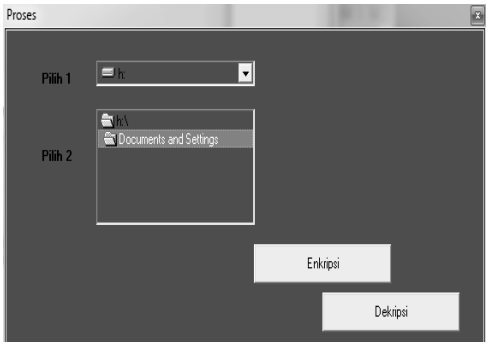
Gambar 4.2 Proses Enkripsi Data

Setelah Data ditemukan, maka tombol Enkipsi ditekan akan memberikan pemberitahuan nama kunci dari data tersebut seperti pada Gambar 4.3

Gambar 4.3 Pesan Proses Enkripsi

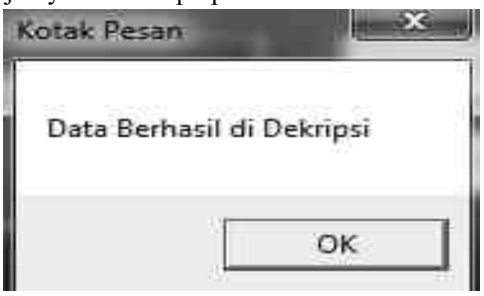
4.1.2 Proses Dekripsi

Pada proses Dekripsi hampir Tampilan menu Dekripsi sama dengan proses Enkripsi, pertama menentukan dimana letak file yang yang telah di Enkripsi dan selanjutnya menekan tombol Dekripsi. Ditunjukkan pada Gambar 4.4.



Gambar 4.4 Proses Dekripsi Data

Selanjutnya akan tampil pesan bahwa Data Sudah di Dekripsi.



Gambar 4.5 Pesan Proses Dekripsi

4.2 Pembahasaan

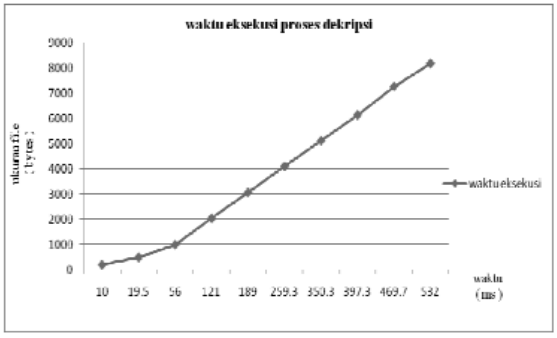
4.2.1 Pengujian

Pengujian tahap ini dilakukan pada saat menggunakan metode operasi file dimana pengujiannya akan dibagi menjadi 3 bagian yaitu waktu eksekusi untuk proses set up key, proses enkripsi, dan proses dekripsi. Untuk pembangkitan kunci (set up key) membutuhkan waktu antara 9 – 10 nano second, sedangkan waktu eksekusi pada proses enkripsi dilakukan 10 kali pengujian dengan mneggunakan file.txt yang memiliki ukuran file yang berbeda beda. Sehingga hasil percobaannya dapat dilihat pada tabel dibawah ini :

Tabel 4.1 Waktu eksekusi pada proses enkripsi

No	Nama file (.txt)	ukuran file (bytes)	Waktu Eksekusi Proses Enkripsi dalam milisecond										
		plaintext	1	2	3	4	5	6	7	8	9	10	average
1	data1	200	10	15	15	10	10	15	10	10	10	10	11.5
2	data2	500	20	15	20	20	20	20	15	20	20	20	19
3	data3	1000	46	50	50	50	60	50	50	50	50	50	50.6
4	data4	2049	140	110	120	110	120	130	110	130	110	130	121
5	data5	3076	171	180	190	190	180	198	175	187	180	171	182.2
6	data6	4104	234	250	250	240	270	230	250	270	256	245	249.5
7	data7	5121	296	343	343	335	302	330	331	320	312	320	323.2
8	data8	6144	353	358	375	365	379	382	383	370	378	388	373.1
9	data9	7275	420	440	456	455	482	477	457	423	455	470	453.5
10	data10	8189	479	510	483	540	468	508	512	475	458	480	491.3

Pada pengujian waktu eksekusi pada proses dekripsi, cara pengujiannya juga hamper sama dengan proses enkripsi, sehingga dari data tabel bisa digambarkan grafiknya sebagai berikut ini :



Pada pengujian ini akan bagi menjadi 2 bagian yaitu pengujian pada proses enkripsi dan proses dekripsi. Pada pengujian proses enkripsi, karakter yang akan diproses, akan di enkripsi terlebih dahulu dengan sebuah kunci menjadi suatu ciphertext yang berupa data interger. Ketika akan diproses kembali menggunakan socket, data integer akan diubah terlebih dahulu menjadi data bertipe string.

Pada pengujian keberhasilan proses enkripsi dan dekripsi Rivest Code-5 dan Rivest Code-6 dengan metode operasi file, pengujiannya dilakukan dengan cara melakukan enkripsi file.txt yang berisi dari beberapa karakter dan akan menghasilkan file ciphertext.txt yang berisi beberapa karakter yang susah untuk dibaca. Sedangkan di poses dekripsi, file ciphertext.txt tadi didekripsi sehingga dapat menghasilkan file.txt kembali.

4.2.2 Hasil Pengujian

Berdasarkan hasil pengujian yang dilakukan sebagai berikut :

1. Untuk uji coba yang dilakukan pada algoritma RC5 waktu eksekusi untuk pembangkitan kunci (set up key) sangat cepat sekali yaitu sekitar 9-10 ns.
2. Untuk uji coba yang dilakukan pada algoritma RC6 waktu eksekusi untuk pembangkit kunci (set up key) lebih cepat sekali yaitu 10-11 ns.
3. Pada proses enkripsi dan dekripsi, waktu eksekusi tergantung dari besar atau kecilnya ukuran file plaintext. Semakin besar ukuran file plaintext maka semakin lama waktu eksekusinya. Dimana untuk file dengan ukuran sebesar 1 kb membutuhkan waktu eksekusi sebesar 50,6 ms pada proses enkripsi.
4. Tidak terdapat perbedaan antara besar file sebelum masuk enkripsi dengan sesudah dekripsi. Dimana untuk file dengan ukuran sebesar 500 bytes sebelum masuk proses enkripsi, akan dihasilkan file sebesar 500 bytes dari proses dekripsi.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

1. Untuk uji coba yang dilakukan pada algoritma RC5 waktu eksekusi untuk pembangkitan kunci (set up key) sangat cepat sekali yaitu sekitar 9-10 ns, Untuk uji coba yang dilakukan pada algoritma RC6 waktu eksekusi untuk pembangkit kunci (set up key) lebih cepat sekali yaitu 10-11 ns.
2. Pada proses enkripsi dan dekripsi, waktu eksekusi tergantung dari besar atau kecilnya ukuran file plaintext. Semakin besar ukuran file plaintext maka semakin lama waktu eksekusinya. Dimana untuk file dengan ukuran sebesar 1 kb membutuhkan waktu eksekusi sebesar 50,6 ms pada proses enkripsi.
3. Tidak terdapat perbedaan antara besar file sebelum masuk enkripsi dengan sesudah dekripsi. Dimana untuk file dengan ukuran sebesar 500 bytes sebelum masuk proses enkripsi, akan dihasilkan file sebesar 500 bytes dari proses dekripsi.

5.2 Saran

1. Algoritma Rivest Code-5 dan algoritma Rivest Code 6 merupakan algoritma yang termasuk dalam kriptogafi simetris, yang merupakan algoritma yang dapat bekerja jika kunci private dan kunci public harus simetris atau sama
2. Uji coba ini dilakukan pada file yang berektensi text seperti : doc dan txt, untuk itu perlu mencoba penggunaan data dengan tipe lain

DAFTAR PUSTAKA

[1] Fresly Nandar Pabokory, , Indah Fitri Astuti, Awang Harsa Kridalaksana (2015), Implementasi Kriptografi Pengamanan Data Pada Pesan Teks, Isi File Dokumen, Dan File Dokumen Menggunakan Algoritma Advanced Encryption Standard, Universitas Mulawarma, Jurnal Informatika Mulawarman, Vol. 10 No. 1 Februari

[2] Munawar, Perancangan Algoritma Sistem Keamanan Data Menggunakan Metode Kriptografi Asimetris(2012), Jurnal Komputer dan Informatika (KOMPUTA), Universitas Komputer Indonesia, Edisi. I Volume. 1, Maret, Bandung .

[3] Dafid (2006)Kriptografi Kunci Simetris Dengan Menggunakan Algoritma Crypton, Jurnal Algoritma, STMIK MDP, Volume 2 Nomor 3, Palembang.

[4] Syamsinar (2017), Implementasi Kombinasi Algoritma Asimetris Rivest Shamir Adleman Dan Algoritma Simetris Advanced Encryption Standard Pada Aplikasi Pesan Singkat, Skripsi, Jurusan Teknik Informatika Fakultas Sains Dan Teknologi Universitas Islam Negeri (Uin) Alauddin Makassar

[5] Basri (2016), Kriptografi Simetris Dan Asimetris Dalam Perspektif Keamanan Data Dan Kompleksitas Komputasi, Jurnal Ilmiah Ilmu Komputer, Universitas Al Asyariah Mandar, Vol. 2, No. 2,September, Sulbar.

[6] Andrizal (2008), Algoritma Enkripsi Rivest Code 5 (RC-5), Dept. Teknik Elektro Option Teknik Komputer ITB, Bandung.

[7] Riza M Yunus, Dr.H, Harun Sujadi, Karnia (2015) Sistem Keamanan Pesan Dengan Algoritma Rivest Code 6 (Rc-6) Menggunakan Java Pada Smartphone Berbasis Android, Jurnal J-Ensitec, Universitas Majalengka, Vol 02 No.01, November, Majalengka.

[8] Martofori Endriko (2013), Pengamanan Pengiriman Pesan Dalam Gambar Menggunakan Steganografi Dengan Discrete Cosine Transform (Dct) Dan Kriptografi Rivest Code 6 (Rc6), Tugas Akhir, Jurusan Teknik Informatika Fakultas Sains Dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau, Pekanbaru