APLIKASI KRIPTOGRAFI FILE MENGGUNAKAN ALGORITMA BLOWFISH

Suriski Sitinjak [1)],Yuli Fauziah [2)], Juwairiah [3)]
[1,2,3)]Jurusan Teknik Informatika UPN "Veteran" Yogyakarta
Jl. Babarsari no 2 Tambakbayan 55281 Yogyakarta Telp (0274)-485323
e-mail : juwai_riah@yahoo.com

Abstrak

Untuk menjaga keamanan data ataupun informasi yang tersimpan dalam bentuk file, salah satu caranya dengan menggunakan metode kriptografi untuk mengenkripsi data file tersebut sehingga tidak dilihat oleh pihak yang tidak berhak. Salah satu algoritma kriptografi adalah algoritma Blowfish yang merupakan algoritma kriptografi modern kunci simetris berbentuk cipher block. Aplikasi yang dibangun ini dapat mengenkripsi file (plaintext) dalam bentuk teks, gambar, suara, video, juga archieve seperti .zip dan .rar. Enkripsi dilakukan dengan menggunakan kunci tertentu, sehingga menghasilkan chipertext (file yang sudah dienkrip atau disandikan) yang tidak dapat dibaca ataupun dimengerti. Ciphertext tersebut dapat dikembalikan seperti semula jika didekripsi menggunakan kunci yang sama sewaktu mengenkripsi file tersebut. Kunci yang digunakan maksimum 56 karakter.. Metode yang digunakan untuk membangun aplikasi ini adalah metode waterfall. Perangkat lunak yang digunakan untuk User-System Interface-nya adalah Visual Basic 6.0.

Keyword : *kriptografi, kunci simetrik, algoritma blowfish, enkripsi, dekripsi*

1. PENDAHULUAN

Kemudahan akses media komunikasi membawa pengaruh terhadap keamanan informasi yang menggunakan media komunikasi sebagai media penyampaian. Informasi menjadi sangat rentan untuk diketahui, diambil atau bahkan dimanipulasi dan disalahgunakan oleh pihak lain yang tidak berhak. Selama pengiriman dan ketika sampai di tujuan, informasi tersebut harus tetap rahasia dan terjaga keasliannya atau tidak dimodifikasi. Penerima informasi harus yakin bahwa informasi tersebut memang benar berasal dari pengirim yang tepat, begitu juga sebaliknya, pengirim yakin bahwa penerima informasi adalah orang yang sesungguhnya. Selain itu penerima tidak ingin pengirim membantah pernah mengirim informasi tersebut, dan jika hal tersebut terjadi penerima perlu membuktikan ketidakbenaran penyangkalan tersebut. Untuk permasalahan-permasalahan keamanan tersebut diperlukan suatu metode untuk menjaga keamanan informasi. Salah satu metodenya adalah kriptografi.

Kriptografi akan merahasiakan informasi dengan menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Saat ini banyak bermunculan algoritma kriptografi yang terus dianalisis, dicoba dan disempurnakan untuk mencari algoritma yang dianggap memenuhi standar keamanan. Beberapa algoritma kriptografi yang dikenal antara lain *DES, Rijndael, Blowfish, RC4, Vigenere Cipher, Enigma, IDEA* dan lainnya.

Blowfish merupakan salah satu algoritma yang tidak dipatenkan dan cukup kuat karena memiliki ruang kunci yang besar dan panjangnya bisa beragam, sehingga tidak mudah diserang pada bagian kuncinya. Suatu sistem kriptografi yang baik terletak pada kerahasiaan kunci dan bukan pada kerahasiaan algoritma yang digunakan. Blowfish pada strategi implementasi yang tepat akan lebih optimal, dapat berjalan pada memori kurang dari 5 KB dan kesederhanaan pada algoritmanya. Untuk itu dibangun sebuah aplikasi yang dapat digunakan untuk mengamankan data atau informasi berupa *file* dengan menggunakan metode Blowfish ini. Selain itu diharapkan pula aplikasi yang dibangun ini dapat melihat kinerja algoritma Blowfish dari segi waktu prosesnya.

II. DASAR TEORI

Aplikasi

Aplikasi atau juga disebut program aplikasi adalah program yang dibuat oleh pemakai yang ditujukan hanya untuk melakukan suatu tugas khusus (Kadir, 2003).

Kriptografi

Kriptografi berasal dari bahasa Yunani yaitu *cryptós* yang artinya "*secret*" (yang tersembunyi) dan *gráphein* yang artinya "*writting*" (tulisan). Jadi, kriptografi berarti "*secret writting*" (tulisan rahasia). Definisi yang dikemukakan oleh Bruce Schneier (1996), kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan (*Cryptography is the art and science of keeping messages secure*).

Terminologi dalam Kriptografi

Ada beberapa istilah-istilah yang penting dalam kriptografi, yaitu :

1. **Pesan** (*Plaintext* dan *Ciphertext*) : **Pesan** (*message*) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya. Pesan asli disebut **plainteks** (*plaintext*) atau teks-jelas (*cleartext*). Sedangkan pesan yang sudah disandikan disebut **cipherteks** (*chipertext*)
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. **Penyadap** (*eavesdropper*) adalah orang yang mencoba menangkap pesan selama ditransmisikan.
4. **Kriptanalisis** dan **Kriptologi** : **Kriptanalisis** (*cryptanalysis*) adalah ilmu dan seni untuk memecahkan chiperteks menjadi plainteks tanpa mengetahui kunci yang digunakan. Pelakunya disebut **kriptanalis**. **Kriptologi** (*cryptology*) adalah studi mengenai kriptografi dan kriptanalisis.
5. **Enkripsi** dan **Dekripsi :** Proses menyandikan plainteks menjadi cipherteks disebut **enkripsi** (*encryption*) atau *enciphering*. Sedangkan proses mengembalikan cipherteks menjadi plainteks semula dinamakan **dekripsi** (*decryption*) atau *deciphering*.
6. ***Cipher*** dan **Kunci :** Algoritma kriptografi disebut juga ***cipher*** yaitu aturan untuk *enchipering* dan *dechipering*, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. **Kunci** (*key*) adalah parameter yang digunakan untuk transformasi *enciphering* dan *dechipering.* Kunci biasanya berupa *string* atau deretan bilangan.

Kriptografi Kunci Simetri (Kriptografi Kunci-Privat)

Pada sistem kriptografi kunci-simetri, kunci untuk enkripsi sama dengan kunci untuk dekripsi, oleh karena itulah dinamakan kriptografi simetri. Keamanan sistem kriptografi simetri terletak pada kerahasiaan kuncinya. Ada banyak algoritma kriptografi modern yang termasuk ke dalam sistem kriptografi simetri, diantaranya adalah *DES* (*Data Encryption Standard*), *Blowfish*, *Twofish*, *Triple-DES*, *IDEA*, *Serpent*, *AES* (*Advanced Encryption Standard*).

Algoritma kriptografi (*cipher*) simetri dapat dikelompokkan menjadi dua kategori, yaitu (Lung dan Munir, 2005):

1. *Cipher* aliran (s*tream cipher*)
   Algoritma kriptografi beroperasi pada plainteks/cipherteks dalam bentuk bit tunggal, yang dalam hal ini rangkaian bit dienkripsikan/didekripsikan bit per bit.
2. *Cipher* blok (*block cipher*)
   Algoritma kriptografi beroperasi pada plainteks/cipherteks dalam bentuk blok bit, yang dalam hal ini rangkaian bit dibagi menjadi blok-blok bit yang panjangnya sudah ditentukan sebelumnya.

Algoritma Blowfish

Blowfish diciptakan oleh seorang *Cryptanalyst* bernama Bruce Schneier, Presiden perusahaan *Counterpane Internet Security, Inc* (Perusahaan konsultan tentang kriptografi dan keamanan komputer) dan dipublikasikan tahun 1994. Dibuat untuk digunakan pada komputer yang mempunyai microposesor besar (32-bit keatas dengan *cache* data yang besar). Blowfish merupakan algoritma yang tidak dipatenkan dan *licensefree*, dan tersedia secara gratis untuk berbagai macam kegunaan (Syafari, 2007).

Pada saat Blowfish dirancang, diharapkan mempunyai kriteria perancangan sebagai berikut (Schneier, 1996):

1. Cepat, Blowfish melakukan enkripsi data pada *microprocessors* 32-bit dengan *rate* 26 *clock cycles* per *byte*.
2. *Compact* (ringan), Blowfish dapat dijalankan pada memori kurang dari 5K.
3. Sederhana, Blowfish hanya menggunakan operasi-operasi sederhana: penambahan, XOR, dan *lookup* tabel pada operan 32-bit.
4. Memiliki tingkat keamanan yang bervariasi, panjang kunci yang digunakan oleh Blowfish dapat bervariasi dan bisa sampai sepanjang 448 bit.

Dalam penerapannya sering kali algortima ini menjadi tidak optimal. Karena strategi implementasi yang tidak tepat. Algoritma Blowfish akan lebih optimal jika digunakan untuk aplikasi yang tidak sering berganti kunci, seperti jaringan komunikasi atau enkripsi *file* otomatis. Selain itu, karena algoritma ini membutuhkan memori yang besar, maka algoritma ini tidak dapat diterapkan untuk aplikasi yang memiliki memori kecil seperti *smart card*. Panjang kunci yang digunakan, juga mempengaruhi keamanan penerapan algoritma ini.

Algoritma Blowfish terdiri atas dua bagian, yaitu ekspansi kunci dan enkripsi data (Schneier, 1996).

a. Ekspansi kunci (*Key-expansion*)

Berfungsi merubah kunci (minimum 32-bit, maksimum 448-bit) menjadi beberapa array subkunci (*subkey*) dengan total 4168 byte (18x32-bit untuk P-array dan 4x256x32-bit untuk *S-box* sehingga totalnya 33344 bit atau 4168 byte). Kunci disimpan dalam K-array:

K1, K2, ... Kj $1 \le j \le 14$

Kunci-kunci ini yang dibangkitkan (*generate*) dengan menggunakan subkunci yang harus dihitung terlebih dahulu sebelum enkripsi atau dekripsi data. Sub-sub kunci yang digunakan terdiri dari :

P-array yang terdiri dari 18 buah 32-bit subkunci,

P1, P2, ..., P18

S-box yang terdiri dari 4 buah 32-bit, masing-masing memiliki 256 entri :

S1,0, S1,1, ..., S1,255
S2,0, S2,1, ..., S2,255
S3,0, S3,1, ..., S3,255
S4,0, S4,1, ..., S4,255

Langkah-langkah perhitungan atau pembangkitan subkunci tersebut adalah sebagai berikut:

1. Inisialisasi *P-array* yang pertama dan juga empat *S-box*, berututan, dengan string yang telah pasti. String tersebut terdiri dari digit-digit heksadesimal dari *phi*, tidak termasuk angka tiga di awal. Contoh :
   P1= 0x243f6a88
   P2= 0x85a308d3
   P3= 0x13198a2e
   P4= 0x03707344
   dan seterusnya sampai *S-box* yang terakhir (daftar heksadesimal digit dari phi untuk P-array dan *S-box* bisa lihat Lampiran).
2. XOR-kan P1 dengan 32-bit awal kunci, XOR-kan P2 dengan 32-bit berikutnya dari kunci, dan seterusnya untuk semua bit kunci. Ulangi siklus seluruh bit kunci secara berurutan sampai seluruh *P-array* ter-XOR-kan dengan bit-bit kunci. Atau jika disimbolkan : P1 = P1 $\oplus$ K1, P2 = P2 $\oplus$ K2, P3 = P3 $\oplus$ K3, . . . P14 = P14 $\oplus$ K14, P15 = P15 $\oplus$ K1, . . . P18 = P18 $\oplus$ K4.
   Keterangan : $\oplus$ adalah simbol untuk XOR.
3. Enkripsikan string yang seluruhnya nol (*all-zero string*) dengan algoritma Blowfish, menggunakan subkunci yang telah dideskripsikan pada langkah 1 dan 2.
4. Gantikan P1 dan P2 dengan keluaran dari langkah 3.
5. Enkripsikan keluaran langkah 3 menggunakan algoritma Blowfish dengan subkunci yang telah dimodifikasi.
6. Gantikan P3 dan P4 dengan keluaran dari langkah 5.
7. Lanjutkan langkah-langkah di atas, gantikan seluruh elemen *P-array* dan kemudian keempat *S-box* secara berurutan, dengan hasil keluaran algoritma Blowfish yang terus-menerus berubah.

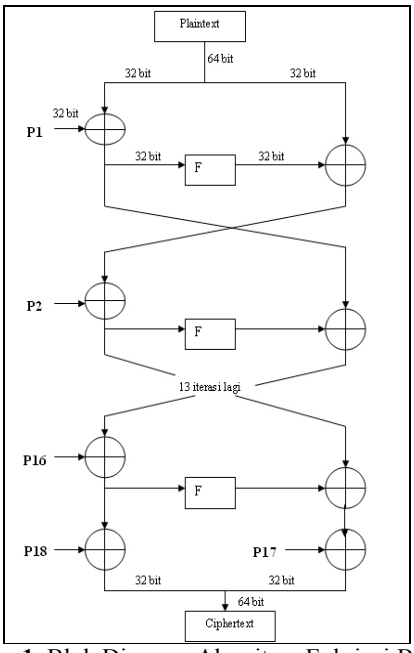
Total keseluruhan, terdapat 521 iterasi untuk menghasilkan subkunci-subkunci dan membutuhkan memori sebesar 4KB.

b. Enkripsi Data

Terdiri dari iterasi fungsi sederhana (*Feistel Network*) sebanyak 16 kali putaran (iterasi), masukannya adalah 64-bit elemen data X. Setiap putaran terdiri dari permutasi kunci-*dependent* dan substitusi kunci- dan data-*dependent*. Semua operasi adalah penambahan (*addition*) dan XOR pada variabel 32-bit. Operasi tambahan lainnya hanyalah empat penelusuran tabel array berindeks untuk setiap putaran. Langkahnya adalah seperti berikut.

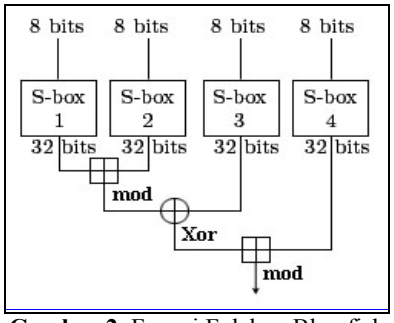
1. Bagi X menjadi dua bagian yang masing-masing terdiri dari 32-bit: X_L, X_R.
2. Lakukan langkah berikut
   For i = 1 to 16:
   $X_L = X_L \oplus P_i$
   $X_R = F(X_L) \oplus X_R$
   Tukar X_L dan X_R
3. Setelah iterasi ke-16, tukar X_L dan X_R lagi untuk melakukan membatalkan pertukaran terakhir.
4. Lalu lakukan
   $X_R = X_R \oplus P_{17}$
   $X_L = X_L \oplus P_{18}$
5. Terakhir, gabungkan kembali X_L dan X_R untuk mendapatkan cipherteks.

Untuk lebih jelasnya, gambaran tahapan pada jaringan feistel yang digunakan Blowfish adalah seperti pada Gambar 1.



Gambar 1. Blok Diagram Algoritma Enkripsi Blowfish

Pada langkah kedua, telah dituliskan mengenai penggunaan fungsi F. Fungsi F adalah: bagi XL menjadi empat bagian 8-bit: a,b,c dan d.
$F(XL) = ((S1,a + S2,b \bmod 2^{32}) \text{ XOR } S3,c) + S4,d \bmod 2^{32}$ .....(2.1)
Agar dapat lebih memahami fungsi F, tahapannya dapat dilihat pada Gambar 2.

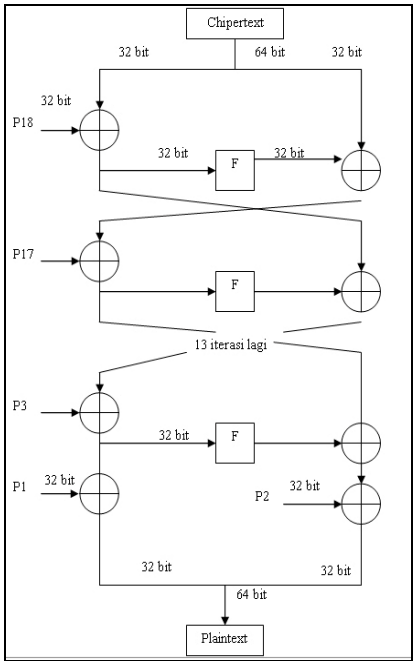


Gambar 2. Fungsi F dalam Blowfish

Dekripsi sama persis dengan enkripsi, kecuali bahwa P_1, P_2,…, P_{18} digunakan pada urutan yang berbalik (*reverse*). Algoritmanya dapat dinyatakan sebagai berikut (Schneier, 1996) :

for i = 1 **to** 16 **do**
  XRi = XLi-1 ⊕ P19-i;
  XLi = F[XRi] ⊕ XRi-1;
XL17 = XR16 ⊕ P1;
XR17 = XL16 ⊕ P2;

Blok diagram dekripsi seperti pada Gambar 3.



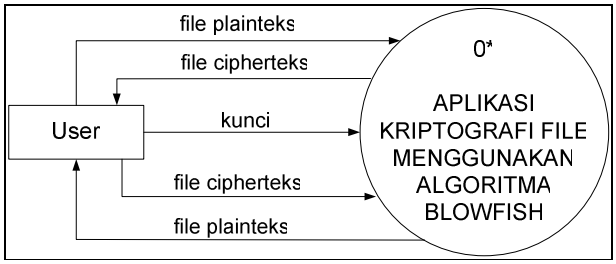
Gambar 3. Blok Diagram Dekripsi Blowfish

III. HASIL DAN PEMBAHASAN

3.1 Perancangan Proses

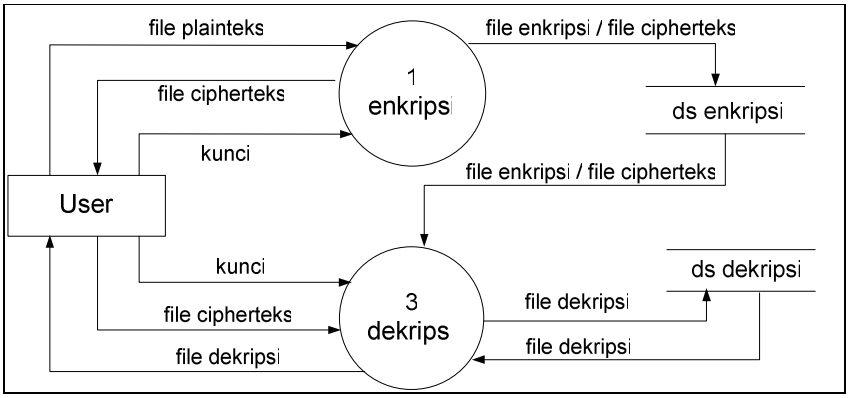
Perancangan proses dapat digambarkan dalam bentuk *Data Flow Diagram* (DFD).

DFD Level 0



Gambar 4. DFD Level 0

DFD Level 1



Gambar 5. DFD Level 1

3.2 Implementasi Program

Halaman utama Aplikasi Kriptografi Blowfish
Pada halaman utama ini, ada 4 tombol, yaitu : Enkripsi/Dekripsi, Tentang Programmer, Manual, dan keluar aplikasi.

Gambar 6. Halaman utama Aplikasi Kriptografi Blowfish

Halaman Enkripsi/ Dekripsi
Untuk melakukan enkripsi, sebelumnya user memilih file sumber (yang akan dienkripsi), file tujuan (untuk menyimpan hasil yang sudah dienkripsi), kunci (maksimum 56 karakter). Untuk melakukan dekripsi, sama seperti proses enkripsi. Hasil outputnya dapat dilihat ukuran file sumber, ukuran file tujuan, dan waktu proses.

Gambar 7. Halaman Enkripsi/Dekripsi

3.3. Pengujian Aplikasi Program

Pada bagian ini dilakukan pengujian aplikasi untuk mengenkripsi *file* dan setelah proses enkripsi selesai dilakukan akan dilihat hasilnya kemudian dilakukan pengujian apakah *file* tersebut bisa dikembalikan seperti semula. Pengujian dilakukan pada beberapa *file* dengan ekstensi berbeda, yaitu ekstensi-ekstensi *file* yang didukung oleh aplikasi ini kemudian akan dilihat perubahan dari setiap ekstensi serta hasil enkripsi dari masing-masing *file*. Pengujian dilakukan pada ukuran *file* yang berbeda dan membandingkan waktu proses untuk masing-masing proses enkripsi/dekripsi. Ekstensi *file* yang dapat dienkripsi dan perubahan ekstensinya dapat dilihat pada Tabel 1.

Tabel 1. Ekstensi *File* yang didukung oleh Aplikasi blowCypt v1.0 dan Perubahan Ekstensi setelah Proses Enkripsi

Jenis *file*	Ekstensi	Ekstensi hasil enkripsi
File teks	txt	txe
Adobe PDF File	pdf	pde
Rich Text Format	rtf	rte
Microsoft Office File	doc	doe
	xls	xle
	mdb	dbe
	ppt	pte
File web	htm	hte
File gambar	bmp	bme
	jpg	jpe
	gif	gie
Suara/audio	wav	wae
	mp3	m3e
	wma	wme
Video	mpg	mpe
	avi	vie
	dat	dte
	wmv	wve
Arsip	zip	zie
	rar	rae

3.4. Hasil Pengujian terhadap Ukuran *File* dan Waktu Proses

Hasil pengujian pada beberpa *file* di atas dapat dirangkum dalam tabel berikut yang menunjukkan ukuran dan waktu proses untuk masing-masing *file* pada tiap proses enkripsi dan dekripsi.

Tabel 2. Hasil Proses Enkripsi beberapa *File* yang berekstensi berbeda

No	Nama *file* plainteks	Ukuran *file* plainteks (byte)	Ukuran *file* cipherteks (byte)	Waktu proses enkripsi (detik)
1	latEnkrip.txt	2128	2144	0.077375
2	Algoritma Pendukung Kriptografi.pdf	35860	35872	0.64025
3	Manual Program.rtf	29198	29216	0.515125
4	Enkripsi Data.doc	59904	59920	0.968625
5	data QC NP CHART.xls	35328	35344	0.609125
6	Studio.mdb	1671168	1671184	12.937
7	Tipe & Mode Algoritma Simetri.ppt	232960	232976	3.48425
8	ASCII Code - The extended ASCII table.htm	50744	50760	0.85925
9	FACTORY.BMP	44278	44296	0.781125
10	boo.jpg	8608	8624	0.233875
11	peter.gif	26107	26120	0.468375
12	Mexican Hat Dance.wav	288078	288096	3.921875
13	What A Wonderful World.mp3	2877664	2877680	21.18712
14	Alter Bridge - Open your eyes(Acoustic).wma	1230697	1230712	10.4525
15	Mike Portnoy -- As I Am 'On Studio'.mpg	59874885	59874904	408.5772
16	Jack Johnson and Friends - Upside down.avi	51773440	51773456	360.8438

17	bee.dat	22783868	22783880	162.3906
18	Nobody To Play With.wmv	4417647	4417664	32.63987
19	bfsh-sch.zip	6737	6752	0.202375
20	_my_files.rar	257133	257152	3.811625

Tabel 3. Hasil Proses Dekripsi beberapa *File* Hasil Enkripsi

No	Nama *file* cipherteks	Ukuran *file* cipherteks (byte)	Ukuran *file* hasil dekripsi (byte)	Waktu proses dekripsi (detik)
1	latEnkrip.txe	2144	2128	0.077625
2	Algoritma Pendukung Kriptografi.pde	35872	35860	0.65575
3	Manual Program.rte	29216	29198	0.546875
4	Enkripsi Data.doe	59920	59904	0.9835
5	data QC NP CHART.xle	35344	35328	0.62475
6	Studio.dbe	1671184	1671168	13.14063
7	Tipe & Mode Algoritma Simetri.pte	232976	232960	3.562125
8	ASCII Code - The extended ASCII table.hte	50760	50744	0.82775
9	FACTORY.BME	44296	44278	0.764875
10	boo.jpe	8624	8608	0.233875
11	peter.gie	26120	26107	0.4835
12	Mexican Hat Dance.wae	288096	288078	3.937125
13	What A Wonderful World.m3e	2877680	2877664	21.43737
14	Alter Bridge - Open your eyes(Acoustic).wme	1230712	1230697	10.42113
15	Mike Portnoy -- As I Am 'On Studio'.mpe	59874904	59874885	420.5466
16	Jack Johnson and Friends - Upside down.vie	51773456	51773440	363.859
17	bee.dte	22783880	22783868	162.9843
18	Nobody To Play With.wve	4417664	4417647	32.1405
19	bfsh-sch.zie	6752	6737	0.202875
20	_my_files.rae	257152	257133	3.812375

Dari hasil di atas dapat dilihat bahwa besarnya ukuran *file* mempengaruhi waktu atau lamanya proses enkripsi/dekripsi. Dari keduapuluh ekstensi yang dapat diproses oleh aplikasi ini dan dilakukan percobaan pada sebuah *file* yang mewakili masing-masing ekstensi, terlihat bahwa semakin besar ukuran *file*, maka semakin banyak waktu yang diperlukan untuk proses tersebut. Waktu proses untuk enkripsi dan dekripsi untuk masing-masing *file* sedikit berbeda, diakibatkan ukuran antara *file* plainteks dan *file* cipherteksnya sedikit berbeda. Ukuran *file* cipherteks sedikit lebih besar dibandingkan plainteksnya. Perbedaan ukuran antara plainteks dan cipherteks ini mengakibatkan waktu proses yang diperlukan untuk dekripsi sedikit lebih besar dibandingkan untuk proses enkripsi. Penambahan jumlah byte dalam *file* cipherteks diakibatkan penambahan beberapa byte untuk mode enkripsi CBC (*Cipher Block Chaining*) pada proses enkripsi. Tetapi ketika cipherteks didekripsi kembali, ukuran *file* kembali seperti plainteksnya.

IV. KESIMPULAN

Berdasarkan keseluruhan proses yang dilakukan untuk membangun Aplikasi Kriptografi *File* menggunakan Algoritma Blowfish ini dapat disimpulkan bahwa aplikasi ini telah berhasil dibangun dan dapat berfungsi sesuai tujuan, yaitu mengamankan data ataupun informasi yang berupa *file* (plainteks) dengan mengacak *file* tersebut sehingga tidak dapat dibaca atau dimengerti. Aplikasi ini juga telah berhasil mengembalikan *file* yang telah diacak tersebut (cipherteks) seperti semula dengan menggunakan kunci yang sama sewaktu enkripsi. Selain itu, aplikasi ini dapat digunakan untuk melihat kinerja algoritma Blowfish dalam pengimplementasiannya, yaitu bagaimana kecepatan proses enkripsi/dekripsi jika dikaitkan dengan ukuran dari sebuah *file*.

Kecepatan proses enkripsi/dekripsi bergantung pada besarnya ukuran *file*, semakin besar ukuran *file* semakin banyak waktu yang dibutuhkan untuk enkripsi/dekripsi. Terjadi penambahan byte pada *file* hasil enkripsi yang mengakibatkan ukuran *file* enkripsi dan *file* plainteks sedikit berbeda, tetapi ketika *file* enkripsi dikembalikan (didekripsi) ukuran *file* akan kembali seperti ukuran *file* plainteksnya.

V. DAFTAR PUSTAKA

Ariyus, Dony, 2006, *Kriptografi Keamanan Data dan Komunikasi*, Graha Ilmu, Yogyakarta.

Harpiandi, 2003, *Belajar Sendiri Pemrograman Database dengan ADO Menggunakan Visual Basic 6.0*, Elex Media Komputindo, Jakarta.

Hartono, Jogiyanto, 1999, *Pengenalan Komputer Dasar Ilmu Komputer, Pemrograman, Sistem Informasi dan Intelegensi Buatan,* Andi, Yogyakarta.

Kadir, Abdul, 2003, *Pengenalan Sistem Informasi*, Andi, Yogyakarta.

Lung, C., Munir, R., 2005, *Studi dan Implementasi AES dengan Empat Mode Operasi Block Cipher*. <http://www.informatika.org/~rinaldi/ta.htm>, (5 Desember 2005, diakses tanggal 10 Juni 2008).

Munir, Rinaldi. 2004. *Bahan Kuliah IF5054 Kriptografi*. <www.informatika.org/~rinaldi/Kriptografi>, tanggal akses mulai 24 Oktober 2008.

Munir, Rinaldi. 2006. *Pengantar Kriptografi*. <www.informatika.org/~rinaldi/Kriptografi>, tanggal akses mulai 20 Oktober 2008.

Pressman, R, 2002, *Rekayasa Perangkat Lunak Pendekatan Praktisi*, Andi Offset, Yogyakarta.

Randy, Adhitya, 2006, *Studi dan Perbandingan Algoritma Blowfish dan Twofish*. <www.webmail.informatika.org/~rinaldi/Kriptografi/2006-2007/Makalah1/Makalah>, diakses tanggal 10 Juni 2008.

Syafari, Anjar, 2007, *Sekilas Tentang Enkripsi Blowfish*, <www.ilmukomputer.com>, diakses tanggal 10 Juni 2008.

Schneier, Bruce, 1994. *Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish),* Springer-Verlag.

Schneier, Bruce, 1996, *Applied Cryptography, Second Edition,* John Wiley & Son, New York.

Wahana, 2003, Memahami Model Enkripsi dan Security Data, Andi Offset, Yogyakarta.