@lgoritma
Jurnal Ilmiah STMIK GI MDP
Volume 2 Nomor 3, Oktober 2006

Kriptografi Kunci Simetris Dengan Menggunakan Algoritma *Crypton*

Dafid

STMIK MDP Palembang
dafid@stmik-mdp.net

Abstrak: *Kriptografi* dapat digunakan sebagai suatu teknik untuk sistem keamanan pada sistem komunikasi data komputer. Algoritma *Crypton* yang termasuk dalam algoritma *Simetrik* merupakan algoritma enkripsi blok data dengan ukuran blok data yang dienkripsi pada mode EBC dan CBC adalah 128 bit sementara panjang kunci bervariasi dari 64 bit hingga 256 bit. Pada penelitian ini algoritma *Crypton* dianalisa kinerjanya dengan simulasi pada *Personal Computer* (PC) menggunakan bahasa pemrograman Microsoft Visual C++ 6.0. Hasil Pengujian menunjukkan bahwa Algoritma *Crypton* memiliki *Avalanche Effect* yang baik, penambahan ukuran file hasil yang besarnya tidak selalu sama antara file yang satu dengan yang lainnya, tidak memiliki kunci lemah serta waktu untuk melakukan proses enkripsi dan dekripsi adalah sama baik dalam mode EBC maupun CBC.

Kata Kunci : Kriptografi, Algoritma Simetrik, Algoritma *Crypton*.

1 PENDAHULUAN

Di era informasi seperti sekarang ini, data atau informasi yang bersifat penting dan rahasia telah menjadi aset yang sangat berharga. Data atau informasi yang berharga tersebut tentunya akan menimbulkan resiko bilamana diakses oleh pihak-pihak yang tidak berhak (*unauthorized person*). Oleh karena itu sudah seharusnya proses pengamanan data mendapat perhatian khusus.

Berbagai cara telah dikembangkan untuk melindungi data dari pihak-pihak yang tidak berhak. Salah satu teknik yang dipakai adalah dengan menggunakan kriptografi (*cryptography*), yaitu ilmu yang menyandikan suatu data menjadi kode tertentu yang sulit dimengerti. Dengan menggunakan kriptografi data asli yang dikirim (*plaintext*) diubah ke dalam bentuk data tersandi (*ciphertext*), kemudian data tersandi tersebut dapat dikembalikan ke bentuk data sebenarnya hanya dengan menggunakan kunci (*key*) tertentu yang hanya dimiliki oleh pihak yang sah saja.

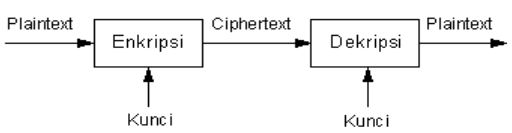
Saat ini telah banyak bermunculan berbagai algoritma kriptografi yang tentunya setiap algoritma menawarkan kelebihan dan kekurangan masing-masing. Salah satunya adalah algoritma CRYPTON yang akan dibahas pada penelitian ini dan mungkin dapat dijadikan pertimbangan dalam penggunaan dan perkembangan algoritma yang telah digunakan sebelumnya.

2 TINJAUAN PUSTAKA

Kata kriptografi (*Cryptography*) berasal dari bahasa Yunani yaitu dari kata Cryptos yang artinya tersembunyi dan Graphein yang artinya menulis. Kriptografi dapat diartikan sebagai suatu ilmu ataupun seni yang mempelajari bagaimana sebuah data dikonversi ke bentuk tertentu yang sulit untuk dimengerti.(Bruce Schneier, 1996). Kriptografi bertujuan untuk menjaga kerahasiaan informasi atau data supaya tidak dapat diketahui oleh pihak yang tidak berhak (*unauthorized person*).

Suatu data yang tidak disandikan disebut *plaintext* atau *cleartext*. Sedangkan data yang telah tersandikan disebut *ciphertext*. Proses yang dilakukan untuk mengubah *plaintext* menjadi *ciphertext* disebut enkripsi (*encryption*) atau *encipherment*. Sedangkan proses untuk mengubah *ciphertext* kembali ke *plaintext* disebut dekripsi (*decryption*) atau *decipherment*. Dalam kriptografi diperlukan parameter yang digunakan untuk proses konversi

data yaitu suatu set kunci. Enkripsi dan dekripsi data dikontrol oleh sebuah kunci atau beberapa kunci. Secara sederhana istilah-istilah diatas dapat digambarkan sebagai berikut :



Gambar 1: Proses Enkripsi/Dekripsi Sederhana

Berdasarkan kunci yang dipakai, algoritma kriptografi dapat dibedakan atas dua jenis yaitu algoritma simetrik (*symmetric*) dan asimetrik (*asymmetric*)

2.1 Algoritma Simetrik

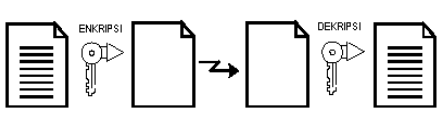
Algoritma simetrik dapat pula disebut sebagai algoritma konvensional, dimana kunci dekripsi dapat ditentukan dari kunci enkripsinya, begitu pula sebaliknya. Pada algoritma simetrik, kunci enkripsi dan kunci dekripsinya sama.

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.

Proses enkripsi dan dekripsi dari fungsi algoritma ini dapat dinotasikan sebagai berikut

E_k (P) = C (Proses Enkripsi)
D_k (C) = P (Proses Dekripsi)

Dimana E adalah fungsi enkripsi, D adalah fungsi dekripsi, k adalah kunci enkripsi dan dekripsi, P adalah *plaintext* (pesan yang sebenarnya) dan C adalah *ciphertext* (hasil enkripsi dari *plaintext*). Proses enkripsi dan dekripsi dengan algoritma simetrik dapat digambarkan sebagai berikut :



Gambar 2: Enkripsi dan Dekripsi Algoritma Simetrik

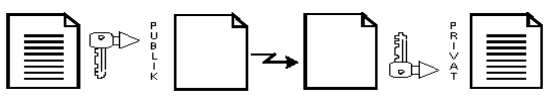
2.2 Algoritma Asimetrik

Algoritma Asimetrik (*Asymmetric* atau *Public Key*) adalah algoritma yang menggunakan kunci yang berbeda untuk proses enkripsi dan dekripsi. Algoritma ini disebut juga algoritma kunci umum (*public key algorithm*) karena kunci untuk enkripsi dibuat umum atau dapat diketahui oleh setiap orang, tapi kunci untuk dekripsi hanya diketahui oleh orang yang berwenang mengetahui data yang disandikan. Jadi hanya orang tertentu saja yang berhak terhadap kunci dekripsi, walaupun kunci enkripsi dapat diketahui dan digunakan oleh orang lain.

Kunci enkripsi pada algortima ini disebut kunci publik (*public key*) dan kunci dekripsi sering disebut dengan kunci pribadi atau kunci rahasia (*private key*). Proses enkripsi dengan kunci publiknya (misalkan “ek”) dan proses dekripsi dengan kunci rahasianya (misalkan “dk”) akan menghasilkan persamaan sebagi berikut:

E_{ek} (P) = C (Proses Enkripsi)
D_{dk} (C) = P (Proses Dekripsi)

Proses enkripsi dan dekripsi dengan algoritma asimetrik dapat digambarkan sebagai berikut :



Gambar 3: Enkripsi dan Dekripsi Algoritma Asimetrik

Pada dasarnya algoritma simetrik terdiri atas dua metode yaitu metode blok *cipher* dan aliran *cipher*. Pada subbab ini pembahasan lebih ditekankan pada metode kriptografi blok *cipher*, karena algoritma kriptografi CRYPTON yang dibahas disini termasuk metode kriptografi blok *cipher*.

2.3 Metode Blok *Cipher*

Pada blok *cipher*, *plaintext* yang akan disandikan dipecah menjadi blok-blok dengan

panjang yang sama. Blok *cipher* menyandikan setiap *plaintext* tersebut menjadi blok *ciphertext* dengan proses enkripsi yang identik dan keseluruhan blok *plaintext* disandikan dengan kunci yang sama.

Dalam sub bab ini metode blok *cipher* meliputi teknik penyandian dan mode operasi yang digunakan untuk menyandikan sebuah *plaintext*.

2.3.1 Teknik Penyandian

Pada blok *cipher*, penyandian dilakukan pada sebuah blok *plaintext* dan berorientasi pada satu bit atau satu karakter pada blok *plaintext* tersebut. Algoritma kriptografi modern merupakan algoritma berbasis kunci. Teknik penyandian pada algoritma blok *cipher* modern juga mengandalkan kunci untuk kerahasiaannya. Beberapa teknik penyandian blok *cipher* modern diantaranya :

1. *Cipher* Berulang

Cipher berulang (*iterated cipher*) mengenkripsi blok *plaintext* dengan sebuah proses yang mengalami beberapa putaran (*round*) atau iterasi untuk mendapatkan blok *ciphertext*. Pada masing-masing putaran diterapkan transformasi atau fungsi putaran (*round function*) yang sama pada *plaintext* dengan menggunakan sub kunci. Fungsi tersebut biasanya merupakan gabungan dari proses substitusi, permutasi, transposisi, atau ekspansi terhadap blok *plaintext*. Kumpulan subkunci biasanya dihasilkan dari secret key dengan fungsi khusus dan biasanya disebut dengan daftar kunci (*key schedule*).

Jumlah putaran dalam *cipher* berulang bergantung pada tingkat keamanan yang diinginkan. Dalam banyak kasus, penambahan jumlah putaran akan memperbaiki keamanan, tetapi untuk beberapa penyandian, jumlah putaran untuk mencapai keamanan yang memadai akan menjadikan penyandian tidak efisien.

2. Jaringan Fiestel

Fiestel *Cipher* merupakan metode yang dikemukakan pertama kali oleh Horst Fiestel pada awal tahun 1970-an. Metode ini sering juga disebut Fiestel Network. Fiestel *cipher* beroperasi terhadap panjang blok data tetap sepanjang n (n adalah bilangan genap), kemudian membagi menjadi 2 blok dengan panjang masing-masing n/2, yang dinotasikan L dan R. Fiestel *cipher* menerapkan metode *cipher* berulang dengan masukan pada putaran ke-i didapat dari keluaran putaran sebelumnya, secara matematis :

$$L_i = R_{i-2}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

$i = 1, 2, 3, \ldots, r$
(r adalah jumlah putaran)

Yang dalam hal ini Ki adalah kunci putaran pada putaran ke-i dan f adalah fungsi transformasi.

Blok *plaintext* adalah gabungan L dan R awal atau secara formal *plaintext* dinyatakan dengan (L0,R0). Sedangkan blok *ciphertext* didapatkan dari L dan R hasil putaran terakhir setelah terlebih dahulu dipertukarkan atau secara formal *ciphertext* dinyatakan dengan (Rr, Lr). Feistel *cipher* banyak diterapkan karena metode ini reversible untuk proses enkripsi dan dekripsi sehingga tidak perlu mendekripsikan algoritma baru untuk dekripsi *ciphertext* menjadi *plaintext*. Hal ini didapatkan karena operator XOR ini mempunyai sifat-sifat unik. Dari sifat unik tersebut maka didapatkan persamaan :

$$L_{i-1} \oplus f(R_{i-1}, K_i) \oplus f(R_{i-1}, K_i) = L_{i-1}$$

Dari persamaan diatas dapat dilihat bahwa sifat reversible tidak tergantung pada fungsi transformasi f yang digunakan sehingga fungsi ini dapat dibuat serumit yang diinginkan.

2.3.2 Mode Operasi

Secara umum, terdapat empat mode enkripsi data baku yang dapat diterapkan pada enkripsi blok dengan sembarang ukuran blok. Mode baku tersebut adalah *Electronic Code Book* (ECB), *Cipher Block Chaining* (CBC), *Cipher Feedback* (CFB) dan *Output Feedback* (OFB). Tetapi yang akan dibahas disini hanya ECB dan CBC saja.

1. *Electronic Code Book* (ECB)

Pada metode ini setiap blok *plaintext* dienkripsi secara independent menjadi blok *ciphertext*. Secara matematis dapat dinyatakan

$$C_i = E_k (P_i)$$
$$P_i = D_k (C_i)$$

2. *Cipher Block Chaining* (CBC)

Pada proses enkripsi mode CBC, blok *plaintext* terlebih dahulu di-XOR-kan dengan blok *ciphertext* hasil enkripsi blok sebelumnya. Blok pertama *plaintext* di-XOR-kan dengan suatu initialization vector (vektor awal) yang besarnya sama dengan blok *plaintext*. Secara matematis dapat dinyatakan :

$$C_i = E_k (P_i \oplus C_{i-1})$$
$$P_i = C_{i-1} \oplus D_k (C_i)$$

2.4 Tinjauan Matematis

Penggunaan ilmu Matematika sangat dominan sekali dalam ilmu kriptografi. Beberapa diantaranya adalah operasi XOR dan operasi AND

2.4.1 Operasi XOR

Pada algoritma kriptografi simetrik, operasi aritmetika yang sering dipakai adalah operasi XOR (*Exclusive* OR) dengan simbol $\oplus$. Operasi XOR ini melibatkan bilangan 0 dan 1 saja. Seluruh kemungkinan nilai operasi XOR ini dapat dilihat pada table 1

2.4.2 Operasi AND

Operasi aritemetika yang juga sering dipakai adalah operasi AND dengan simbol $\wedge$. Operasi AND ini melibatkan bilangan 0 dan 1 saja. Seluruh kemungkinan nilai operasi AND ini dapat dilihat pada tabel 1.

Tabel 1: Hasil Operasi XOR dan Operasi AND

A	B	A⊕B	A⊕B
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

2.5 Proses *Padding*

Proses *padding* adalah proses penambahan byte-byte dummy berupa karakter NULL pada byte-byte sisa yang masih kosong pada blok terakhir *plaintext*, sehingga ukurannya menjadi sama dengan ukuran blok penyandian. Ukuran data yang akan disandikan sebagian besar tidak merupakan kelipatan ukuran blok penyandian. Hal ini mengakibatkan blok terakhir mungkin akan memiliki ukuran yang lebih kecil dari blok penyandian. Karena pada metode *Block Cipher* mengharuskan blok yang akan disandikan memiliki panjang yang tetap maka pada blok terakhir tersebut harus ditambahkan byte-byte tertentu sehingga ukurannya menjadi sama dengan ukuran blok penyandian.

2.6 Kunci Lemah

Beberapa kunci yang digunakan pada proses penyandian dapat merupakan kunci lemah. Dalam algoritma kriptografi yang baik, jumlah kunci lemah adalah sangat kecil. Kunci lemah (*weak key*) adalah sebuah kunci yang mengakibatkan proses enkripsi tidak berbeda dengan proses dekripsi. Bila kunci lemah 'K' maka secara matematis dapat dinyatakan :

$$E_K(P) = D_K(P)$$

Sehingga dipenuhi pula persamaan :

$$E_K(E_K(P)) = P$$
$$D_K(D_K(P)) = P$$

3 IMPLEMENTASI ALGORITMA CRYPTON

Tahap implementasi merupakan tahap pengkodean, dimana dari algoritma yang ada diubah menjadi suatu program dengan menggunakan bahasa pemograman tertentu. Bahasa pemrograman yang digunakan adalah Microsoft Visual C++ versi 6.0 yang dapat berjalan pada lingkungan Windows 9x.

Dalam menguji program, dilakukan simulasi terhadap beberapa file yang mempunyai ukuran yang berbeda pada mode EBC dan mode CBC. Hal ini untuk mendapatkan kemungkinan-kemungkinan yang terjadi dalam proses enkripsi maupun dekripsi data. Untuk keperluan tersebut digunakan file bertipe *text*. Adapun spesifikasi komputer yang digunakan adalah Pentium 233 MMX, RAM 48 MB dan menggunakan Sistem Operasi Windows 98.

3.1 Tinjauan Umum Algoritma Crypton

Crypton adalah blok *cipher* 128 bit dimana tiap blok data direpresentasikan ke dalam array berukuran 4 x 4 byte. Tiap blok data tersebut diproses dengan menggunakan rangkaian putaran transformasi. Proses enkripsi melibatkan 12 kali iterasi dengan putaran transformasi yang sama. Proses dekripsi memiliki proses yang sama dengan proses enkripsi hanya saja diterapkan penjadwalan kunci yang berbeda. Tiap putaran transformasi terdiri dari empat tahap yaitu : substitusi byte, permutasi bit, transposisi byte dan penambahan kunci. Crypton hanya menggunakan operasi sederhana diantaranya AND, XOR dan shift. Algoritma Crypton terdiri dari dua bagian yaitu bagian penjadwalan kunci dan bagian enkripsi data.

3.2 Simulasi Mode ECB

Pada mode ECB proses enkripsi dekripsi dilakukan tanpa adanya tambahan proses *initial value* dalam menghasilkan *ciphertext*. Dalam proses enkripsi dekripsi ini, *plaintext* dan *ciphertext* serta kunci yang diinputkan semuanya dalam format karakter. *Plaintext* dan *ciphertext* diinputkan dalam bentuk file. Sedangkan kunci diinputkan dalam bentuk input karakter biasa. Panjang kunci yang diinputkan dapat bervariasi antara 1 – 256 bit. Namun untuk alasan keamanan sebaiknya panjang kunci minimum 64 bit. Ukuran *plaintext* tidak selalu memenuhi kelipatan blok 128 bit. Oleh karena itu dilakukan proses *padding* terhadap blok terakhir *plaintext* yang tidak memenuhi blok 128 bit. Blok yang kurang dari 128 bit tersebut pada bagian akhirnya diisi dengan nol hingga mencapai 128 bit. Proses *padding* juga dilakukan terhadap kunci yang tidak memenuhi ukuran 256 bit.

Pada pengujian diberikan

Plaintext : abcdefghijklmnop
Kunci : abcdefgh

Dari simulasi didapatkan *ciphertext*:

§ÆÅ;_¦[¦k¦¦ÜDLmº

Selanjutnya dilakukan proses dekripsi dengan

Ciphertext : §ÆÅ;_¦[¦k¦¦ÜDLmº
Kunci : abcdefgh

Dari simulasi didapatkan *plaintext*:

abcdefghijklmnop

Pada proses dekripsi, *plaintext* dapat ditemukan kembali bila menggunakan kunci yang sama dengan proses enkripsi. Bila kunci yang digunakan tidak sama, maka *plaintext* yang semula tidak dapat ditemukan kembali. Bila dekripsi menggunakan kunci yang berbeda seperti dibawah ini :

Ciphertext : §ÆÅ;_¦[¦k¦¦ÜDLmº
Kunci : stuvwxyz
Dari simulasi didapatkan *plaintext*:

á-+Üq8]!9¦+_2_¬

Ternyata *plaintext* yang dihasilkan tidak memberikan hasil seperti *plaintext* semula.

3.3 Simulasi Mode CBC

Pada mode CBC proses enkripsi dekripsi dilakukan dengan adanya tambahan proses initial value dalam menghasilkan *ciphertext*. Ukuran initial value ini adalah sebesar 1 blok penyandian yaitu 128 bit. Initial value akan dibuat secara otomatis dengan membuat blok data berukuran 128 bit yang datanya diambildari sebagian kunci. Dalam proses enkripsi dekripsi ini, *plaintext* dan *ciphertext* serta kunci yang diinputkan semuanya dalam format karakter. *Plaintext* dan *ciphertext* diinputkan dalam bentuk file. Sedangkan kunci diinputkan tidak dalam bentuk file melainkan dalam bentuk input karakter biasa. Panjang kunci yang diinputkan dapat bervariasi antara 1 – 256 bit. Namun untuk alasan keamanan sebaiknya panjang kunci minimum 64 bit. Ukuran *plaintext* tidak selalu memenuhi kelipatan blok 128 bit. Oleh karena itu dilakukan proses *padding* terhadap blok terakhir *plaintext* yang tidak memenuhi blok 128 bit. Blok yang kurang dari 128 bit tersebut pada bagian akhirnya diisi dengan nol hingga mencapai 128 bit. Proses *padding* juga dilakukan terhadap kunci yang tidak memenuhi ukuran 256 bit.

Pada pengujian diberikan

Plaintext : abcdefghijklmnop
Kunci : abcdefgh

Dari simulasi didapatkan *ciphertext*:

q+a↑ -GxìmR:_&-T

Selanjutnya dilakukan proses dekripsi dengan

Ciphertext : q+a↑ -GxìmR:_&-T
Kunci : abcdefgh

Dari simulasi didapatkan *plaintext*:

abcdefghijklmnop

Pada proses dekripsi, *plaintext* dapat ditemukan kembali bila menggunakan kunci yang sama dengan proses enkripsi. Apabila dilakukan proses dekripsi dengan menggunakan kunci yang berbeda maka *plaintext* yang dihasilkan juga berbeda.

Ciphertext : q+a↑ -GxìmR:_&-T
Kunci : stuvwxyz

Dari simulasi didapatkan *plaintext*:

_ÅWEX+Ñg│¦__X– "

Ternyata *plaintext* yang dihasilkan tidak memberikan hasil seperti *plaintext* semula.

3.4 Analisa Pengaruh Perubahan Bit

Karakteristik yang digunakan untuk melihat pengaruh perubahan bit pada suatu algoritma kriptografi adalah *Avalanche Effect*, yaitu perubahan yang kecil pada *plaintext* maupun key akan menyebabkan perubahan yang signifikan terhadap *ciphertext* yang dihasilkan. Dengan kata lain perubahan satu bit pada *plaintext* maupun *key* akan menghasilkan perubahan banyak bit pada *ciphertext*. Umumnya bit pada *ciphertext* mengalami perubahan dari jumlah bit *plaintext* sebesar 50%. Suatu *avalanche effect* dikatakan baik jika perubahan bit yang dihasilkan berkisar antara 45-60% (sekitar separuhnya). Semakin banyak perubahan yang terjadi mengakibatkan akan semakin sulit bagi kriptanalis untuk dapat melakukan kriptanalisis. Bila perubahan bit yang terjadi hanya sedikit, hal ini dapat mempermudah dalarn mencari *plaintext* atau kunci.

Pengujian terhadap perubahan bit (*Avalanche test*) pada algoritma Crypton dilakukan dengan cara mengubah sebesar 1 bit terhadap *plaintext* atau kunci yang diberikan. Pada pengujian yang pertama yaitu pengujian terhadap *plaintext*, digunakan dua buah *plaintext* yang berbeda dengan sebuah kunci yang sama. Pada pengujian yang kedua yaitu pengujian terhadap kunci, digunakan dua buah kunci yang berbeda dengan sebuah *plaintext* yang sama.

Plaintext dan kunci yang digunakan tersebut yaitu sebagai berikut :

<u>Pengujian I</u>

Plaintext 1:
0x00000000000000000000000000000000

Plaintext 2:
0x10000000000000000000000000000000
Kunci:
0x00000000000000000000000000000000

Pengujian II

Plaintext:
0x00000000000000000000000000000000
Kunci 1:
0x00000000000000000000000000000000
Kunci 2:
0x10000000000000000000000000000000

Dari hasil pengujian terlihat bahwa baru setelah tiga putaran perbedaan yang terjadi mencapai 60 bit atau hampir setengah dari besar *ciphertext* (128 bit). Kemudian pada akhir proses enkripsi terdapat 63 bit yang berbeda. Pada putaran pertama, perbedaan yang terjadi langsung mencapai 64 bit atau hampir setengah dari besar *ciphertext* (128 bit). Kemudian pada akhir proses enkripsi terdapat 64 bit yang berbeda.

Dari kedua hasil pengujian diatas menunjukkan bahwa Crypton memiliki pengaruh perubahan bit yang kuat, dengan kata lain Crypton menunjukkan suatu *Avalanche Effect* yang baik.

3.5 Analisa Perubahan Ukuran File

Pada bagian sebelumnya telah dijelaskan bahwa algoritma Crypton yang digunakan dalam kriptografi ini memiliki ukuran blok data sebesar 128 bit (16 byte). Oleh karena itu data yang masuk ke dalam sistem harus berukuran kelipatan 128 bit (16 byte). Namun apabila ukuran data yang masuk tidak mencukupi 16 byte maka dilakukan proses *padding*. Proses *padding* ini menambahkan byte-byte dummy ke dalam blok terakhir *plaintext* sehingga menyebabkan terjadinya pengembangan ukuran data. Pengembangan ukuran data ini dapat dengan jelas dilihat dari proses enkripsi file.

Penambahan ukuran file hasil enkripsi besarnya tidak selalu sama antara file yang satu dengan file yang lainnya. Hal ini tergantung dari besar byte *padding* yang digunakan dan mode operasi yang digunakan. Untuk proses enkripsi pada mode ECB (*Electronic Code Book*) jumlah byte yang ditambahkan maksimum sebesar 15 byte yang setara dengan 1 blok penyandian data (16 byte). Sedangkan pada mode CBC (*Chaining Block Cipher*) jumlah byte yang ditambahkan maksimum sebesar 31 byte yang setara dengan 2 blok penyandian data (32 byte). Hal ini dikarenakan pada mode EBC penambahan maksimum jumlah byte terjadi hanya diakibatkan oleh proses *padding* sebesar 15 byte sedangkan pada mode CBC penambahan jumlah byte terjadi selain diakibatkan oleh proses *padding* sebesar 15 byte juga diakibatkan oleh adanya *initial value* sebesar 16 byte.

3.6 Analisa Kunci Lemah

Kunci lemah (*weak key*) dalam sistem kriptografi merupakan kunci yang dapat mengembalikan *plaintext* dari *ciphertext* tanpa melalui proses dekripsi. Suatu *plaintext* yang dienkripsi dengan menggunakan kunci lemah akan diperoleh suatu *ciphertext*. Kemudian bila *ciphertext* tersebut dienkripsi kembali dengan kunci lemah tersebut maka akan didapatkan *plaintext* semula. Pada sistem kriptografi Crypton hal ini sudah dihindari.

Algoritma Crypton melakukan proses dekripsi dengan cara menggunakan penjadwalan kunci yang berbeda dari proses enkripsi. Metode penjadwalan kunci pada Crypton menjamin bahwa semua kunci yang digunakan tidak memliki kunci lemah. Pada algoritma Crypton digunakan jumlah rotasi dan konstanta putaran yang berbeda sehingga diharapkan tidak terdapat kunci lemah. Percobaan dilakukan terhadap beberapa kunci dengan *plaintext*:

0x00000000000000000000000000000000.

Dari beberapa kunci yang diujicobakan, ternyata tidak ditemukan *plaintext* pada enkripsi kedua. Hal itu menunjukkan bahwa pada Crypton tidak memiliki kunci lemah. Oleh sebab itu, untuk sementara dapat dikatakan bahwa Crypton tidak memiliki kunci lemah, namun tidak menutup kemungkinan akan ditemukannya kunci lemah pada algoritma ini.

Dari hasil pengujian diatas dapat diambil beberapa kesimpulan serta saran sebagai berikut:

4 KESIMPULAN

1. Keamanan enkripsi dan dekripsi dengan menggunakan kunci simetris pada dasarnya terletak pada kuncinya sendiri, artinya bahwa kunci yang digunakan untuk mengenkripsi dan dekripsi adalah kunci *private key*, dimana kunci tersebut tidak boleh dipublikasikan kepada umum.

2. Semakin kompleks metode pengacakan yang digunakan maka semakin sulit untuk membongkar pesan yang terenkripsi ke dalam bentuk aslinya.

3. Penambahan ukuran besar file pada proses enkripsi disebabkan oleh proses *padding* (penambahan byte-byte *dummy* pada blok *plaintext* yang masih kosong).

4. Proses enkripsi dan dekripsi memerlukan waktu yang sama untuk data dan metoda yang sama.

5 SARAN

1. Untuk pengembangan lebih lanjut, sebaiknya program yang dibuat dapat mengenkripsi data tidak hanya pada data yang bertipe teks, melainkan juga data yang bertipe bukan teks misalnya data berupa gambar.

2. Diharapkan juga supaya program yang dibuat memiliki kemampuan mengenkripsi ukuran data yang lebih besar tanpa mengurangi performansi program.

3. Diupayakan program yang dibuat diimplementasikan tidak hanya pada komputer *stand alone* (*Personal Computer*) juga pada komputer berbasis jaringan.

DAFTAR PUSTAKA

[1] Abdul Kadir. 1997. *Pemrograman Dasar Turbo C Untuk IBM PC*. Yogyakarta : Andi Offset.

[2] Bruce Schneier. 1996. *AppliedCryptograp by: Protocols, Algorithms, and Source Code in C*. USA : John Wiley & Sons, Inc.

[3] Chae Hoon Lim. CRYPTON : A New 128-bit Block Cipher. Korea : Information and Communication Research Center Future Systems, Inc., http://crypt.future.co.kr/chlim. Diakses pada 15/08/2006.

[4] Chae Hoon Lim.. A Note on Imple - menting CRYPTON – Endian-Neutral Implementation –. Korea : Cryptography & Network Security Center Future Systems, Inc., http://www.future.co.kr/fs-tr01-02. Diakses pada 15/08/2006.

[5] Knused. 1994. *Block Cipher – Analysis design and application*. Phd Dissertation: Aarhus University.