Analisis Keamanan Algoritma Kriptografi RC6

Rudianto [1)]

1) Jurusan Teknik Informatika ITB, Bandung 40116, email : if14099@students.if.itb.ac.id

Abstract - RC6 adalah algoritma kriptografi *symmetric-key* yang merupakan perkembangan dari algoritma RC5 yang disertakan dalam *Advances Encrytion Standard* (AES). RC6 menggunakan esensi dari *data-dependent rotations*. RC6 adalah algoritma yang menggunakan ukuran blok hingga 128 bit, dengan ukuran kunci yang digunakan bervariasi antara 128, 192 dan 256 bit. RC6 memecah blok 128 bit menjadi 4 buah blok 32-bit, dan mengikuti aturan enam operasi dasar yaitu : penjumlahan, pengurangan, exclusive OR, perkalian bilangan integer, geser bit ke kanan, dan geser bit ke kiri. Proses utama dalam algoritma ini adalah penjadwalan kunci, dan enkripsi-dekripsi. Berdasarkan proses-proses tersebutlah RC6 RC6 merupakan salah satu algoritma kriptografi yang paling canggih dan masih dapat dikatakan belum terpecahkan. Beberapa metode kriptanalisis yang sering digunakan, mulai dari yang kovensional sampai yang modern telah digunakan tetapi belum ada yang mampu memecahkannya dalam waktu yang singkat. Dalam percobaannya penulis terlebih dahulu menganalisis performansi dari algoritma RC6 kemudian menggunakan metode konvensional, exhaustive search, *linear* dan *differential crytanalysis* dengan hanya menggunakan sekali putaran untuk melakukan serangan. Penulis juga melakukan beberapa modifikasi terhadap algoritma ini dengan cara mengurangi salah satu komponen dari metode ini untuk melihat seberapa besar pengaruhnya terhadap kekuatan algoritma ini.

Kata Kunci*: RC6, keamanan, kriptografi*

1. PENDAHULUAN

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 $f(x) = x(2x + 1) \bmod 2^{32}$, 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 2^{32} tambahan. Algoritma cipher ini biasanya memakai 20 putaran.

RC6, bila dibandingkan dengan RC5, menggunakan 4 (empat) *working registers*, dan menyertakan operasi perkalian integer sebagai operasi primitif tambahan. Operasi perkalian meningkatkan penyebaran untuk tiap putarannya sehingga meningkatkan faktor keamanan, mengurangi putaran, dan meningkatkan performa hasil.

Tingkat keamanan pada algoritma ini terletak pada kekuatan rotasi yang berdasarkan data, penggunaan eksklusif OR yang bergantian, fungsi modulo dan fungsi persamaan yang menggunakan rotasi yeng tetap. Dengan menghilangkan salah satu atau beberapa aspek di atas, maka cipher yang dihasilkan akan menjadi lebih lemah terhadap beberapa serangan yang sudah diketemukan sebelumnya. Beberapa jenis serangan modern terhadap algoritma ini hanya dapat dilakukan secara teori tanpa praktek serangan sesungguhnya.

Pada awalnya penulis ingin menggunakan simulator yang telah ada, akan tetapi source yang didapatkan dari [3] tidak dapat digunakan, maka penulis memutuskan membuat simulator sederhana untuk melakukan pengujian sederhana. Berdasarkan simulator tersebut, berdasarkan beberapa sampel data, dianalasis perfornansi dari algoritma RC6, sebelum dilakukan serangan.

2. ALGORITMA SIMULATOR RC6

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 negatuf yang menunjukan banyaknya iterasi selama proses enskripsi dan parameter b menunjukkan ukuran kunci enkripsi dalam byte. Setelah algoritma ini masuk ke dalam AES, maka ditetapkan bahwa nilai w = 32, r = 20, dan b bervariasi antara 16, 24, dan 32 byte.

- Operasi dasar

RC6-w/r/b memecah blok 128-bit menjadi 4 buah blok 32-bit, dan mengikuti aturan enam operasi dasar sebagai berikut :

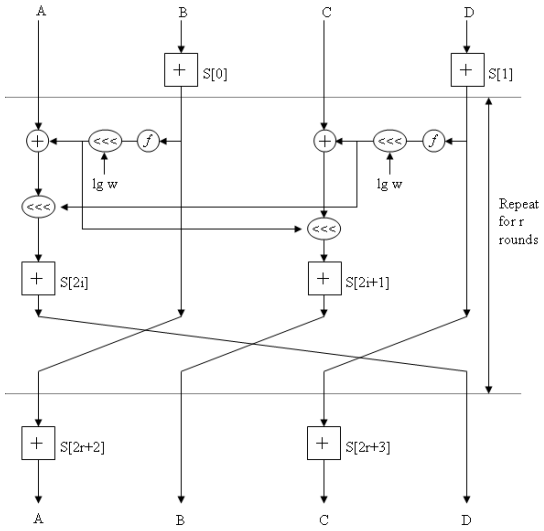
a + b	operasi penjumlahan bilangan integer
a – b	operasi pengurangan bilangan integer
a ⊕ b	operasi eksklusif OR (XOR)
a x b	operasi perkalian bilangan integer
a <<< b	a dirotasikan ke kiri sebanyak b
a >>> b	a dirotasikan ke kanan sebanyak b

- Algoritma enkripsi

Karena RC6 memecah blok 128 bit menjadi 4 buah blok 32 bit, maka algoritma ini bekerja dengan 4 buah

register 32-bit A, B, C, D. Byte yang pertama dari plainteks atau cipherteks ditempatkan pada byte A, sedangkan byte yang terakhirnya ditempatkan pada byte D. Dalam prosesnya akan didapatkan (A, B, C, D) = (B, C, D, A) yang diartikan bahwa nilai yang terletak pada sisi kanan berasal dari register di sisi kiri. Diagram blok berikut akan lebih menjelaskan proses enkripsi yang terjadi pada algoritma RC6.

Gambar 1



Algoritma RC6 menggunakan sub kunci yang dibangkitkan dari kunci dan dinamakan dengan S[0] hingga S[43]. Masing-masing sub kunci panjangnya 32 bit. Proses enkripsi pada algoritma RC6 dimulai dan diakhiri dengan proses *whitening* yang bertujuan untuk menyamarkan iterasi yang pertama dan yang terakhir dari proses enkripsi dan dekripsi. Pada proses *whitening* awal, nilai B akan dijumlahkan dengan S[0] dan nilai D dijumlahkan dengan S[i]. Pada masing-masing iterasi pada RC6 menggunakan dua (2) buah sub kunci. Sub kunci pada iterasi yang pertama menggunakan sub-sub kunci lanjutannya. Setelah iterasi ke-20 selesai, dilakukan proses *whitening* akhir dimana nilai A dijumlahkan dengan S[42], dan nilai C dijumlahkan dengan S[43].

Setiap iterasi pada algoritma RC6 mengikuti aturan sebagai berikut, nilai B dumasukkan ke dalam fungsi f, yang didefinisikan sebagai f(x) = x(2x + 1), kemudian diputar ke kiri sejauh 5 bit. Hasil yang didapat pada proses ini dimisalkan sebagai u. Nilai u kemudian di XOR dengan C dan hasilnya menjadi nilai C.

Nilai t juga digunakan sebagai acuan bagi C untuk memutar nilainya ke kiri. Begitu pula dengan nilai U, juga digunakan sebagai acuan bagi nilai A untuk melakukan proses pemutara ke kiri.

Kemudian sub kunci S[2i] pada iterasi dijumlahkan dengan A, dan sub kunci S[2i+1] dijumlahkan dengan C. Keempat bagian dari blok kemudian akan dipertukarkan dengan mengikuti aturan bahwa nilai A ditempatkan pada D, nilai B ditempatkan pada A, nilai C ditempatkan pada B, dan nilai (asli) D ditempatkan pada C. Demikian iterasi tersebut terus berlangsung hingga dua puluh (20) kali.

- Algoritma dekripsi

Proses dekripsi cipherteks pada algoritma RC6 merupakan pembalikan dari proses enkripsi. Pada proses *whitening*, bila proses enkripsi menggunakan operasi penjumlahan, maka pada proses dekripsi menggunakan operasi pengurangan. Sub kunci yang digunakan pada proses *whitening* setelah iterasi terakhir diterapkan sebelum iterasi pertama, begitu juga sebaliknya sub kunci yang diterapkan pada proses *whitening* sebelum iterasi pertama digunakan pada *whitening* setelah iterasi terakhir. Akibatnya, untuk melakukan dekripsi, hal yang harus dilakukan semata-mata hanyalah menerapkan algoritma yang sama dengan enkripsi, dengan tiap iterasi menggunakan sub kunci yang sama dengan yang digunakan pada saat enkripsi, hanya saja urutan sub kunci yang digunakan terbalik.

- Penjadwalan kunci

Pengguna memasukkan sebuah kunci yang besarnya b byte, dimana 0≤b≤255 byte kunci ini kemudian ditempatkan dalam array c w-bit kata L[0] … L[c-1]. Byte pertama kunci akan ditempatkan sebagai pada L[0], byte kedua pada L[1], dan seterusnya (catatan, bila b=0 maka c=1 dan L[0]=0). Masing-masing nilai kata w-bit akan dibangkitkan pada penambahan kunci round 2r+4 dan akan ditempatkan pada array S[0,…,2r+3].

Konstanta P32 = B7E15163 dan Q32 = 9E3779B9 (dalam satuan heksadesimal) adalah “konstanta ajaib” yang digunakan pada penjadwalan kunci pada RC6. Nilai P32 diperoleh dari perluasan bilangan biner e-2, dimana e adalah sebuah fungsi logaritma. Sedangkan nilai dari Q32 diperoleh dari perluasan bilangan biner ∅-1, dimana ∅ dapat dikatakan sebagai “*golden ratio*” (rasio emas).

- Struktur data

Berikut ini algoritma – algoritma yang digunakan untuk mengimplementasikan simulator kriptografi RC6. Dengan algoritma-algoritma ini dibuat sebuah simulator kriptografi RC6 yang dapat menghasilkan data analisis performansinya.

1. Algoritma Pembangkit Sub Kunci

```
KAMUS
Type Word32 : 32 bit {tipe data 32 bit}
Kunci : string {kunci yang dimasukkan oleh
pemakai}
i, j, c, s, v : integer
A : integer
B : integer
S : array [0..43] of word32
L : array [0..43] of word32
Function ROTL (X:Word32; Y : integer) →
Word32
(fungsi untuk merotasi bit sebanyak variabel
kedua)
ALGORITMA
```

```
Input (Kunci)
S[0] ←P32
For i← 1 to 43 do
S[i] ← S[i-1]+Q32
Endfor

A←B←i←j← 0
V←44
If (c>v) then
V←c
V←v*3
For s←1 to v do
A← S[i] ← ROTL ((S[i]+A+B),3)
B←L[j] ← ROTL (L[j]+A+B,A+B)
I←(i+1) mod 44
J← (j+1) mod c
Endfor
```

2. Algoritma Baca File Masukan

```
Procedure baca_file
{I.S : File belum dibuka}
{F.S : File dibaca per 16 karakter dan
ditampung dalam buffer}
KAMUS
Nama_file : string {nama file masukan}
Buff : array [0..15] of char
i : integer
ALGORITMA
Input (nama_file)
Open (nama_file)
i←0
while (i≤15) and not (EOF) do
Read (nama_file, Buff[i])
Endwhile
Close (nama_file)
```

3. Algoritma whitening awal

```
Procedure whitening_awal
{I.S : blok kedua dan keempat belum
dijumlahkan dengan sub kunci}
{F.S : blok kedua dan keempat yang telah
dijumlahkan dengan sub kunci}
KAMUS
Type word32 : 32 bit (tipe data sebesar 32
bit)
X : word32 array [0..3] {blok
enkripsi/plainteks}
S : array [0..43] of word32 {sub kunci}
ALGORITMA
X[1] ← X[1] + S[0]
X[3] ← X[3] +S[1]
```

4. Algoritma Iterasi

```
Procedure iterasi
{I.S : keempat blok setelah whitening awal
belum diproses}
{F.S : keempat blok yang telah diproses dan
saling dipertukarkan}
KAMUS
Type word32 : 32 bit {tipe data sebesar 32
bit}
X : word32 array [0..3] {blok
enkripsi/plainteks}
Function ROTL(X:Word32; Y : integer)→ word32
{Merotasi bit kekiri sebanyak variabel kedua}
Temp : word32
u, t: word32
i : integer
ALGORITMA
For i ← 1 to 20 do
t ← ROTL ((X[1]*(2*X[1]+1)), 5)
u ← ROTL ((X[3]*(2*X[3]+1)), 5)
X[0] ← (ROTL((X[0] XOR t), u )) + S[2*i]
X[0] ← (ROTL((X[2] XOR u), t )) + S[2*i + 1]
Temp ← X[0]
X[0] ← X[1]
X[1] ← X[2]
X[2] ← X[3]
X[3] ← Temp
End for
```

5. Algoritma Whitening Akhir

```
Procedure Whitening_akhir
{I.S : blok pertama dan ketiga belum
dijumlahkan dengan sub kunci}
{F.S : blok pertama dan ketiga yang telah
dijumlahkan dengan sub kunci}
KAMUS
Type word32 : 32 bit (tipe data sebesar 32
bit)
X : word32 array [0..3] {blok
enkripsi/plainteks}
S : array[0..43] of word32 {sub kunci}
ALGORITMA
X[0] ← X[0] + S[42]
X[2] ← X[2] + S[43]
```

3. JENIS SERANGAN

Dalam algoritma cipher blok ada beberapa jenis serangan yang terdefinisi, antara lain :

1. *Exhaustive Key Search*

Penyerang mencoba semua kemungkinan kunci satu persatu dan mengecek apakah plainteks memiliki kecocokan dengan cipherteks yang menjadi sampel. Untuk sebuah blok cipher dengan k-bit kunci dan n-bit blok, jumlah pasangan cipherteks dan plainteks pengujian yang diperlukan untuk menentukan kunci yang tepat adalah sekitar k/n. Tambahan, jika ada plainteks yang sifatnya redundan, serangan hanya dapat bekerja dengan baik hanya jika sebagian blok cipherteks diketahui. Jumlah blok cipherteks yang dibutuhkan bergantung pada frekuensi kata-kata yang sering muncul.

2. *The matching cipherteks attack*

Serangan ini didasarkan pada fakta bahwa setidaknya ada sebuah blok cipher berukuran m-bit yang muncul dari hasil enkripsi yang berasal dari $2^{m/2}$ blok plainteks sehingga dapat diketahui sedikit informasi mengenai plainteksnya.

3. *Differential crytanalysis*

Cara ini merupakan salah satu metode kriptanalisis konvensional yang paling umum dan sering digunakan, yang dipublikasikan leh Bilham dan Shamir pada tahun 1990. Kriptanalisis ini biasa digunakan untuk melawan metode-metode kriptografi yang dibangun dari perulangan fungsi yang tetap. Salah satu caranya adalah dengan mencari selisih antara dua (2) bit string, misalnya X dan X'

persamaan (1) $\Delta X = X \Phi (X')^{-1}$

dimana Φ adalah kumpulan operasi terhadap kumpulan bit string yang digunakan untuk mengombinasikan kunci dengan plainteks dalam fungsi putaran dan dimana $(X)^{-1}$ adalah *inverse* elemen dari X. Ide di balik metode ini adalah selisih dari plainteks dan cipherteks, yang didapatkan dari

hasil kombinasi dengan kunci, selalu sama besarnya.

4. *Truncated differentials*

Untuk beberapa cipherteks, dimungkinkan dan sangat bermanfaat, memprediksi hanya sebagian nilai saja dengan menggunakan *differential crytanalysis* untuk setiap putarannya

5. *Impossible differentials*

Salah satu tipe dari *differential crytanalysis* yang kemungkinannya adalah 0. Ide utamanya adalah menspesifikkan bahwa ketidak-mungkinan terhadap beberapa putaran terhadap cipher serangan. Kemudian dengan menebak beberapa kunci dalam putaran yang tidak tercakup dalam fungsi, dapat dilakukan pembuangan terhadap beberapa nilai kunci yang salah.

6. *Higher-order differentials*

Sebuah s^{th}*-order differentials* didefinisikan secara rekursif sebagai sebuah fungsi *differentials* dari fungsi $(s\text{-}1)^{th}$*-order differentials*, dimana s^{th}*-order differentials* berisi kumpulan 2^n teks yang mengandung *predetermined differentials*.

6. *Linear crytanalysis*

Linear crytanalysis ditemukan oleh Matsui pada tahun 1993. *Linear crytanalysis* merupakan serangan *known-plainteks* dimana penyerang mengeksploitasi pendekatan persamaan linier dari beberapa bit plainteks, beberapa bit cipherteks, dan beberapa bit kunci.

7. Kriptanalisis Mod n

Serangan ini merupakan generalisasi dari serangan linier. Serangan ini dapat digunakan untuk cipherteks dimana beberapa kata terbiaskan dalam persamaan modulo n, dimana n adalah integer yang bernilai kecil. Telah terbukti bahwa algoritma kriptografi yang menggunakan hanya rotasi bit dan menambahkan modulo dari 2^{32} sangat rapuh terhadap serangan ini.

8. *Related-key attacks*

Knudsen telah menggunakan metode mendapatkan hasil enkripsi menggunakan satu kunci terhadap sebuah plainteks yang terpilih dan berhasil mengurangi kunci secara *exhaustive search* sampai dengan empat (4) kali lebih cepat. Serangan ini membutuhkan hasil enkripsi dari beberapa kunci yang berbeda, dalam beberapa kasus juga memerlukan plainteks yang sama, oleh karena itu, cara ini dianggap kurang realistik.

4. HASIL DAN PEMBAHASAN

1. Hasil kajian analisis performansi

- Struktur Cipher

Mekanisme yang digunakan pada algoritma RC6 ini adalah pembagian blok dari 128 bit menjadi empat buah blok yang masing-masing besarnya 32 bit. Fungsi f(x)= x(2x+1) yang diikuti dengan pergeseran 5 bit ke kiri digunakan untuk memberikan tingkat keamanan data yang tinggi. Pada RC6 tidak ditemui adanya Feistel Network seperti pada DES, namun fungsi f(x)=x(2x+1) dan pergeseran 5 bit ke kiri sudah dapat menghasilkan tingkat pengacakan data yang tidak kalah dengan algoritma yang menggunakan jaringan feistel.

- Banyak iterasi

Penentuan banyaknya iterasi didasarkan atas azas keseimbangan. Dimana jika jumlah round sedikit akan menyebabkan cipher menjadi mudah untuk dipecahkan. Sedangkan jika jumlah iterasi semakin banyak, akan menyebabkan kecepatan proses enkripsi/dekripsi akan semakin berkurang.
Sebenarnya jumlah round sebanyak 20 cukup beralasan. Akan tetapi berdasarkan analisis keamanan yang sangat mendalam, diketahui serangan terbaik terahadap RC6 dapat mencapai 8 round. Untuk alasan ini, pencipta RC6 memeberikan *security margin* yang cukup besar yaitu sebanyak 12 round untuk memberikan perlindungan keamanan yang tinggi terhadap RC6.

- Penjadwalan Kunci

Penjadwalan kunci pada RC6 menggunakan sebuah sBox untuk melakukan iterasi awal terhadap S[0]…S[43]. Nilai konstanta P32=B7E15163 (heksadesimal) didefinisikan sebagai inisialisasi awal terhadap S[0] dan inisialisasi awal S[1] hingga S[43] didapatkan dengan menambahkan Q32=9E3779B9 (heksadesimal) terhadap inisialisasi awal subkunci-subkunci sebelumnya. Pada proses utama penjadwalan kunci, digunakan beberapa operasi seperti pergeseran bit dan operasi penjumlahan yang menyebabkan proses penjadwalan kunci pada RC6 mempunyai kekuatan yang sama dengan RC6 cipher itu sendiri.

- Perubahan Ukuran Arsip

Arsip cipherteks mempunyai ukuran yang lebih besar dari arsip plainteks. Hal ini terjadi karena adanya proses pedding. Pada mode EBC maupun pada metode CBC, perubahan maksimum besarnya arsip cipherteks adalah sebesar satu blok penyandian data (16 byte). Hal ini terjadi semata-mata karena adanya proses padding (16 byte). Walaupun pada metode CBC terdapat proses inisial vektor, namun yang terjadi hanyalah operasi XOR antara inisial vektor dan blok 128 bit sehingga tidak akan merubah besar ukuran cipherteks menjadi lebih besar dibanding mode EBC.

- *Avalanche Effect*

Salah satu karakteristik untuk menentukan baik atau tidaknya suatu algoritma kriptografi adalah dengan melihat *avalanche effect*-nya. Perubahan yang kecil pada plainteks maupun key akan menyebabkan perubahan yang signifikan terhadap cipherteks yang dihasilkan. Atau dengan kata lain, perubahan satu bit pada palintext maupun key akan menghasilkan

perubahan banyak bit pada cipherteks. Suatu avalanche effect dikatakan baik jika perubahan bit yang dihasilkan berkisar antara 45-60% (sekitar separuhnya, 50 % adalah hasil yang sangat baik). Hal ini dikarenakan perubahan tersebut berarti membuat perbedaan yang cukup sulit untuk kriptanalis melakukan serangan.
RC6 memperlihatkan sebuah *avalanche effect* yang baik. Hasil yang ditunjukan ini sesuai dengan parameter yang ditetapkan yaitu 50% dari besar blok penyandian. Hal tersebut ditunjukan dari hasil yang diperlihatkan dibawah ini :

Plainteks 1 : $00000000000000000000000000000000_{16}$
Plainteks 2 : $10000000000000000000000000000000_{16}$
Kunci : $00000000000000000000000000000000000000000000000000000000000000000000_{16}$
Cipherteks 1 : $8F5FBD0510D15FA893FA3FDA6E857EC2_{16}$
Cipherteks 2 : $2321D38ACCAAEA5DBA682E7BDCE095AB_{16}$

Dari hasil diatas terlihat bahwa perbedaan bit yang dihasilkan pada cipherteks 1 dan cipherteks 2 adalah sebanyak 72 bit atau sekitar 56% dari besar blok (128 bit). Sedangkan untuk perubahan kunci satu bit akan ditunjukan pada hasil dibawah ini.

Plainteks $00000000000000000000000000000000_{16}$
Kunci 1 $00000000000000000000000000000000000000000000000000000000000000000000_{16}$
Kunci 2 $10000000000000000000000000000000000000000000000000000000000000000000_{16}$
Cipherteks 1 $8F5FBD0510D15FA893FA3FDA6E857EC2_{16}$
Cipherteks 2 $11395D4BFE4C8258979EE2BF2D24DFF4_{16}$

Dari hasil tersebut tampak bahwa perbedaan kunci satu bit menghasilkan perbedaan bit sebesar 64 bit atau sekitar 50 % dari besar blok (128 bit).

- Kunci Lemah dan Kunci Setengah Lemah

Pada algoritma RC6 sampai saat ini masih belum ditemukan adanya kunci lemah. Semua kunci mempunyai kekuatan yang sama dan dapat digunkan untuk enkripsi ganda (kunci lemah mempunyai sifat yaitu jika sebuah plainteks dienkripsi ganda menggunkan kunci lemah akan menghasilkan plainteks itu sendiri). Metode penjadwalan kunci apda RC6 menjamin bahwa semua kunci yang digunakan tidak memliki kunci lemah. Beberapa kunci yang bernilai ekstrim (termasuk empat buah kunci lemah yang dimiliki algoritma DES) tidak berpengaruh pada algoritma RC6. Kunci setengah lemah adalah sepasang kunci yang mempunyai sifat yaitu jika sebuah plainteks dienkripsi dengan suatu kunci dan didekripsi kembali dengan kunci pasangannya akan menghasilkan plainteks itu kembali. Beberapa pasangan kunci yang bernilai ekstrim (termasuk pasangan kunci setengah lemah yang dimiliki DES) tidak berpengaruh pada algoritma RC6.

2. Hasil percobaan kriptanalisis

Percobaan dilakukan pada komputer berspesifikasi 2.2 MHz Petium Celeron dengan 256 memori RAM yang berjalan di bawah OS Windows XP. Simulator dibuat menggunakan bahasa pemrograman Java dan dikompilasi dengan JDK1.6.
Berdasarkan percobaan yang telah dilakukan dengan algoritma RC6 yang paling sederhana, penulis tidak dapat menemukan kunci yang tepat hanya dengan menggunakan metode *exhaustive search*. Penulis memperkirakan serangan yang terbaik yang dapat dilakukan terhadap RC6 adalah *exhaustive search* untuk enkripsi kunci b-byte. Usaha yang diperlukan untuk melakukan ini minimal sebanyak 2^{1408} operasi, sesuatu yang tidak mungkin dilakukan oleh PC. Dengan metode kriptanalisis yang lebih canggih, yaitu *differential* dan *linear crytanalysis* masih dimungkinkan untuk menemukan hasil dengan syarat jumlah putaran yang cukup sedikit, tidak mencapai 20 putaran, yang merupakan standar dari RC6. Kesulitan lainnya adalah sulit untuk menemukan karakteristik dari fungsi perulangan atau pendekatan dari fungsi liniernya.

3. Referensi hasil percobaan lain

Untuk mencoba jenis serangan yang sudah tergolong jenis modern, dibutuhkan data yang jumlahnya sangat besar dan mengandung 2^n blok pasangan plainteks-cipherteks yang sudah diketahui dan sebuah proses paralel untuk menemukan kunci dari 2^n kemungkinan. Jika diasumsikan cipherteks dapat ditemukan dalam kecepatan satu (1) terabyte / detik, maka dibutuhkan waktu lebih dari satu (1) tahun bagi 50 komputer yang bekerja secara paralel untuk mendekripsi 2^{64} blok data.

4. Modifikasi terhadap RC6 dan penurunan tingkat keamanannya

Penulis mencoba menghilangkan beberapa proses yang ada di dalam algortima ini. Pada percobaan sebelumnya, penulis sudah mengurangi jumlah putaran sehingga masih mampu dihasilkan kunci yang tepat, untuk percobaan berikutnya penulis mencoba menghilangkan proses putaran dan menarik kesimpulan bahwa kunci sangat mudah dipecahkan.
Pencobaan menghilangkan fungsi persamaan berhasil memperlihatkan bahwa RC6 menjadi sangat mudah untuk dipecahkan. Percobaan terakhir adalah menghilangkan proses eksklusif OR dan menggantinya dengan proses modulasi 2^{32}. Cipher yang dihasilkan masih sulit untuk diserang dan dicari kunci yang tepatnya, tetapi, secara teori, modifikasi ini dapat menyebabkan algoritma ini sangat rapuh terhadap jenis serangan kriptanalisis *mod n.*

5. KESIMPULAN

Algoritma kriptografi RC6 adalah algoritma yang sangat kuat. Algoritma ini memiliki performansi yang sangat baik walaupun besar cipherteks selalu sedikit lebih besar daripada besar plainteks, memiliki *avelache effect* yang kecil, dan tidak memiliki kunci lemah ataupun kunci setengah lemah. Sampai dengan saat ini, belum ada serangan yang secara signifikan dapat memecahkan kunci dari algoritma RC6 standar dalam tempo waktu yang singkat. Algoritma yang terdiri atas rotasi yang berdasarkan data, penggunaan eksklusif OR yang bergantian, fungsi modulo dan fungsi persamaan yang menggunakan rotasi yang tetap. Jika salah satu komponennya dihilangkan, maka kunci dapat ditemukan dengan relatif mudah.

DAFTAR REFERENSI

[1] Munir, Rinaldi, Ir.,M.T. 2007. *Diktat Kuliah IF-5054 Kriptografi*. Informatika-ITB : Bandung.

[2] S. Contini, R.L. Rivest, M.J.B. Robshaw dan Y.L. Yin. *The Security of the RC6 Block Chiper*. (e-books)

[3] http://www.processorexpert.com/lnk_mcHC08__Download_Embedded_Beans.php#

[4] http://www.vicman.net/dir/13327/Download-Hot-Crypt.htm

[5] http://RSA.com/rsalabs/aes