Kompleksitas Waktu Algoritma Kriptografi RC4 Stream Cipher

Nur Adi Susliawan Dwi Caksono - 13508081

Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung
Jl.Ganesha No 10 Bandung
e-mail: if18081@students.if.itb.ac.id

ABSTRAK

Kriptografi berasal dari dua suku kata yaitu kripto dan grafi. Kripto artinya menyembunyikan, sedangkan grafi artinya ilmu. Kriptografi (Cryptography) adalah suatu ilmu yang mempelajari sistem sandi untuk menjamin kerahasiaan dan keamanan data, dilakukan oleh seorang kriptographer

Di makalah ini akan dijelaskan mengenai kriptografi dengan metode RC4 Stream Cipher. Di makalah ini, Anda akan mendapatkan penjelasan mengenai Algoritma RC4 Stream Cipher dalam proses enkripsi dan proses dekripsi, Kompleksitas waktu dari Algoritma Kriptografi RC4 Stream Cipher serta keamanan dari Algoritma Kriptografi RC 4 Stream Cipher.

Kata kunci: Kriptografi, Algoritma RC4 Stream Cipher, Enkripsi, Dekripsi, Kompleksitas Waktu

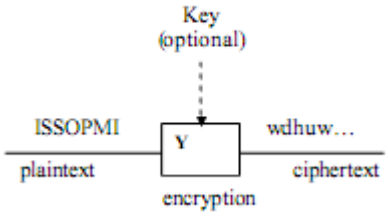
1. PENDAHULUAN

Dalam komunikasi data, terdapat suatu metode pengamanan data yang dikenal dengan kriptografi (*Crypthography*). Kriptografi merupakan suatu metode pengamanan data yang dapat digunakan untuk menjaga kerahasiaan , keaslian dan keutuhan data, serta keaslian pengirim data. Metode ini bertujuan agar informasi penting yang bersifat terbatas atau rahasia yang dikirim melalui sarana telekomunikasi tidak dapat diketahui atau dimanfaatkan oleh pihak yang tidak berhak. Dalam dunia kriptografi terdapat istilah enkripsi dan dekripsi. Enkripsi adalah suatu proses yang melakukan perubahan data dari yang bisa dimengerti (plaintext) menjadi sebuah kode yang tidak dapat dimengerti (ciphertext), sedangkan Dekripsi adalah proses kebalikan dari enkripsi. Secara umum operasi enkripsi dan dekripsi secara matematis dapat ditulis sebagai berikut :

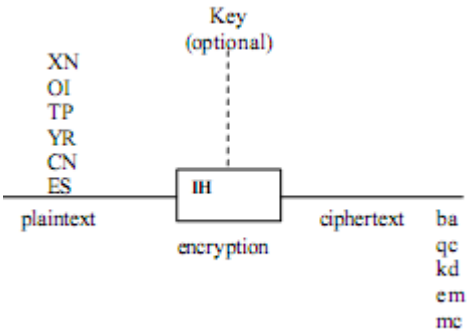
$$EK(M) = C \text{ \{proses enkripsi\}}$$

$$DK(M) = M \text{ \{proses dekripsi\}}$$

Pada proses enkripsi pesan M dengan suatu kunci K disandikan menjadi pesan C. Pada proses dekripsi, pesan C dengan kunci K disandikan menjadi pesan semula (M).
Secara umum dalam proses enkripsi dan dekripsi dikenal dua macam cipher berdasarkan cara kerja penyandiannya, yaitu stream cipher dan block cipher.
Stream Cipher adalah suatu sistem dimana proses enkripsi dan dekripsinya dilakukan dengan cara bit per bit. Pada sistem ini, aliran bit kunci dihasilkan oleh suatu pembangkit acak. Aliran bit kunci pada sistem ini dikenakan operasi XOR dengan aliran bit –bit ciphertext. Keamanan sistem ini tergantung dari pembangkit kunci, jika pembangkit kunci memiliki aliran bit-bit 0 maka ciphertext yang dihasilkan akan sama dengan plaintext, sehingga untuk mengatasi ini diperlukan suatu pembangkit bit-bit kunci yang acak dan tidak berulang.
Sedangkan pada SistemBlock Cipher, plaintext dibagi menjadi beberapa blok yang memilikiukuran yang sama dan tetap. Kemudian setiap bloknya dienkripsi dan didekripsi sekaligus. Cara ini bekerja lebih cepat karena plaintext dibagi atas beberapa blok.



Gambar 1.1 Sistem Stream Cipher



Gambar 1.2 Sistem Block Cipher

RC4 itu sendiri merupakan salah satu jenis stream cipher, yaitu memproses unit atau input data, pesan atau informasi pada satu saat. Unit atau data pada umumnya sebuah byte atau bahkan kadang kadang bit (byte dalam hal RC4). Dengan cara ini enkripsi atau dekripsi dapat dilaksanakan pada panjang yang variabel. Algoritma ini tidak harus menunggu sejumlah input data, pesan atau informasi tertentu sebelum diproses, atau menambahkan byte tambahan untuk mengenkrip. Contoh stream cipher adalah RC4, Seal, A5, Oryx, dan lain-lain. Tipe lainnya adalah block cipher yang memproses sekaligus sejumlah tertentu data (biasanya 64 bit atau 128 bit blok), contohnya : Blowfish, DES, Gost, Idea, RC5, Safer, Square, Twofish, RC6,Loki97,danlain-lain.

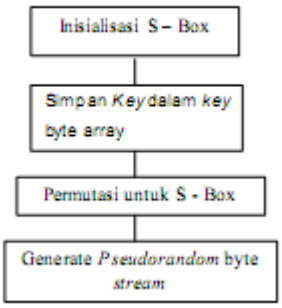
RC4 merupakan enkripsi stream simetrik proprietary yang dibuat oleh RSA Data Security Inc (RSADSI). Penyebarannya diawali dari sebuah source code yang diyakini sebagai RC4 dan dipublikasikan secara *'anonymously'* pada tahun 1994. Algoritma yang dipublikasikan ini sangat identik dengan implementasi RC4 pada produk resmi. RC4 digunakan secara luas pada beberapa aplikasi dan umumnya dinyatakan sangat aman. Sampai saat ini diketahui tidak ada yang dapat memecahkan/membongkarnya, hanya saja versi ekspor 40 bitnya dapat dibongkar dengan cara "brute force" (mencoba semua kunci yang mungkin). RC4 tidak dipatenkan oleh RSADSI, hanya saja tidak diperdagangkan secara bebas (trade secret) (Bruce Scheiner,1996).

2. MEKANISME DASAR KERJA RC4

RC4 memiliki sebuah S-Box, S0, S1....S255 yang berisi permutasi dari bilangan 0 sampai 255. Algoritma ini menggunakan 2 buah indeks, misalkan i dan j. Indeks i digunakan untuk memastikan bahwa suatu elemen berubah dan indeks j akan memastikan bahwa suatu elemen berubah secara random. Sedangkan key pada algoritma ini bersifat opsional. Inti dari algoritma ini ialah membangkitkan byte pseudorandom dari key yang akan dioperasikan dengan operasi XOR terhadap plain text untuk menghasilkan ciphertext

2.1 Proses Enkripsi

Seperti yang telah dijelaskan diatas di proses enkripsi, pseudorandom byte dari key akan dibangkitkan yang nantinya akan dioperasikan dengan operasi XOR dengan plaintext. Dibawah ini merupakan bagan yang menggambarkan rangkaian proses yang dijalankan untuk mengenkripsi data



Gambar 2.1.1 Rangkaian Proses RC4 Stream Cipher

Secara garis besar algoritma RC4 *Stream Cipher* ini terbagi menjadi 2 bagian yaitu : *key setup* dan *stream generation*. Berikut ini akan diberikan contoh penerapan langsung dari Algoritma RC4 Stream Cipher sehingga lebih mudah untuk dimengerti.

Plaintext : enkripsi
ASCII plaintext : 101 110 107 114 105 112 115 105
Key : mahabbah
ASCII Key : 109 97 104 97 98 98 97 104

2.1.1 Key Setup

Pada bagian ini terdapat 3 tahapan proses didalamnya, yaitu :

1) Inisialisasi S-Box

Pada tahapan ini, S-Box akan diisi dengan nilai sesuai indeksnya untuk mendapatkan S-Box awal. Algoritmanya adalah sebagai berikut :

a) For i = 0 hingga i = 255 do
b) Assign nilai S[i] dengan nilai i

Dari Algoritma diatas akan didapat urutan nilai S-Box sebagai berikut

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	240	241	242	243	245	246	247	248	249	250	251	252	253	254	255

Gambar 2.1.1.1 S-Box Inisialisasi Awal

2) Menyimpan Key dalam Key Byte Array

Pada bagian ini, kunci (*key*) yang akan kita gunakan untuk mengenkripsi atau dekripsi akan diinputkan kedalam array berukuran 256 byte secara berulang sampai array terisi penuh.

Algoritma untuk menyimpan key dalam key byte array adalah sebagai berikut :

a) Assign j dengan nilai 1

b) For i = 0 sampai i = 255 do

  Jika (j > panjang kunci) maka

    Assign j dengan 1

c) Assign k ke-i dengan nilai ASCII karakter kunci ke j

d) Nilai j dinaikkan 1

e) Tambahkan i dengan 1, kembali ke bagian b)

Dari Algoritma diatas maka akan didapatkan urutan array key sebagai berikut

109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104
109	97	104	97	98	98	97	104	109	97	104	97	98	98	97	104

Gambar 2.1.1.2 Urutan Array Key (K-Box)

Dari gambar 2.1.2 diatas dapat disimpulkan algoritma diatas mengurutkan byte chiper key secara berulang dari array ke-0 hingga array ke 255.

3) Permutasi pada S-Box

Pada tahapa n ini, akan dibangkitkan suatu nilai yang akan dijadikan aturan dalam permutasi S-Box. Misalkan untuk nilai i = 2 akan dihasilkan nilai j = 26, sehingga nilai S(2) akan ditukar dengan nilai pada S(26). Algoritmanya adalah sebagai berikut:

a) Assign nilai j dengan 0

b) For i = 0 sampai i = 255 lakukan

  Assign y dengan hasil operasi (j+S(i)+K(i))mod256

  Assign nilai j dengan y

  Tukar S(i) dengan S(j)

Dari algoritma diatas akan diperoleh nilai S-Box yang telah mengalami proses transposisi sehingga urutan dari S-Box sekarang ialah sebagai berikut

109	26	57	76	157	245	52	49	181	31	45	103	84	240	73	192
61	235	28	98	3	205	160	128	130	59	22	38	92	122	137	30
247	121	4	135	156	148	231	170	228	16	68	208	159	53	210	20
176	168	220	214	136	19	87	42	120	219	27	169	183	116	200	96
204	110	64	216	89	222	44	126	105	213	164	188	23	47	194	33
209	65	162	142	107	195	119	80	166	238	237	72	246	102	243	241
35	40	77	215	141	100	74	115	146	56	189	163	113	55	67	229
236	185	83	46	18	140	118	50	233	248	99	8	172	203	54	132
13	124	152	252	151	153	147	139	179	190	82	154	224	161	86	0
34	149	29	225	78	14	48	158	25	104	150	184	218	187	95	193
79	43	155	223	143	232	177	251	66	114	202	10	9	88	6	226
239	250	178	71	106	51	11	60	125	242	17	90	197	91	249	173
94	15	201	101	108	234	227	41	75	117	165	174	230	138	207	62
70	244	129	145	182	131	112	7	171	253	111	198	134	180	167	211
39	255	2	1	32	85	254	21	63	217	212	175	12	93	123	191
36	186	37	133	144	24	69	81	206	5	196	199	97	221	58	127

Gambar 2.1.1.3 S-Box setelah ditransposisi

2.1.2 Stream Generation

Pada tahapa ini dihasilkan nilai pseudorandom yang akan dioperasikan dengan operasi XOR untuk menghasilkan ciphertext.

Algoritmannya adalah sebagai berikut :

a) Isi indeks i dan j dengan nilai 0

b) For i = 0 sampai i = panjang plaintext

  Assign i dengan hasil operasi (i+1) mod 256

  Assign j dengan hasil operasi (j+S(i)) mod 256

  Swap S(i) dengan S(j)

  Assign nilai t dengan hasil operasi ((S(i)) + (S(j) mod 256)) mod 256

  Assign nilai y dengan S(t)

  Nilai y diXOR kan terhadap kode plain text

Sehingga ciphertext yang akan terbentuk ialah sebagai berikut

Õ	G	Ó	•	•	[	Đ

Gambar 2.1.2.1 Ciphertext string 'enkripsi'

2.2 Proses Dekripsi

Untuk mendapatkan pesan semula, kunci aliran yang dibangkitkan harus sama dengan kunci aliran pada proses enkripsi.

Algoritma Proses Dekripsi dapat Anda lihat dibawah ini:

- Inisialisasi i dengan 0
- Inisialisasi j dengan 0
- For idx dari 0 hingga Panjang Cipherteks -1 do

  Assign j dengan (i+1) mod 256

  Assign j dengan (j + S[i]) mod 256

  Swap (S[i], S[j])

  Assign t dengan (S[i] + S[j]) mod 256

  Assign K dengan S[t]

Assign P dengan Hasil Operasi K XOR C[idx]

Keterangan:
C : array of karakter cipherteks
K : keystream yang dibangkitkan
P : plainteks
RC4 digunakan untuk pengenkripsian file dalam banyak aplikasi seperti RSA SecurPC (salah satu produk RSA Security, Inc).

3. KOMPLEKSITAS WAKTU RC4 STREAM CIPHER

Algoritma adalah suatu konsep matematis dan analisis untuk menyelesaikan suatu masalah dan persoalan. Algoritma adalah urutan langkah yang tepat dan pasti dalam memecahkan suatu masalah secara logis. Algoritma juga dapat dianalisis efesiensi dan kompleksitasnya. Efisiensi dalam algoritma sangat dipertimbangkan karena suatu masalah dapat diselesaikan dengan berbagai macam cara. Algoritma yang baik adalah algoritma yang efisien dimana algoritma tersebut dinilai dari aspek kebutuhan waktu dan ruang membutuhkan jumlah yang sedikit. Kemangkusan algoritma dapat diukur dari orde yang terdapat dalam persamaan kompleksitas waktu. Kompleksitas waktu diukur dari jumlah tahapan komputasi yang dibutuhkan dalam menjalankan algoritma dimana kompleksitas waktu tersebut merupakan fungsi dari jumlah masukan n.
Hal-hal yang mempengaruhi kompleksitas waktu ialah sebagai berikut :

1. Jumlah masukan data untuk suatu algoritma (n)
2. Waktu yang dibutuhkan untuk menjalankan algoritma tersebut
3. Ruang memori yang dibutuhkan untuk menjalankan algoritma yang berkaitan dengan struktur data dari program.

Kompleksitas mempengaruhi performa dan kinerja dari suatu algoritma. Kompleksitas waktu dibagi menjadi 3 jenis, yaitu worst case, best case, dan average case. Masing-masing jenis kompleksitas ini menunjukkan kecepatan atau waktu yang dibutuhkan algoritma untuk mengeksekusi sejumlah kode.
Pengukuran kinerja kualitatif algoritma biasanya dilakukan dengan menyatakan kinerja sebagai salah satu persamaan sederhana yang menunjukkan hubungan antara input dan kinerja. Cara tradisional untuk menyatakan kinerja algoritma adalah dengan menggunakan notasi Big-O (O).
Kompleksitas waktu dibedakan atas tiga macam, yaitu :
1. Tmax(n) adalah kompleksitas waktu untuk kasus terburuk (worst case),
adalah kebutuhan waktu maksimum.
$T(n) = O(\log n)$
Untuk setiap B, N > 0, logBN = K, if BK = N
Jika (base) B tidak disebut, maka default-nya adalah 2 dalam konteks ilmu computer (binary representation).
Contoh :
Log 32 = 5 (karena 25 = 32)
Log 1024 = 10
Log 1048576 = 20
Log 1 milyard = sekitar 30
2. Tmin(n) : kompleksitas waktu untuk kasus terbaik (best case), dengan kata lain Tmin(n) adalah kebutuhan waktu minimum dari suatu algoritma.
$T(n) = O(n \log n)$
Misalkan T(n) adalah waktu untuk menyelesaikan masalah dengan ukuran input n.
Maka T(1) = 1 (1 adalah quantum time unit ketika melakukan proses base case; ingat konstanta tidak terlalu penting).
Dua buah pemanggilan pengulangan, masing – masing beukuran n/2. waktu yang dibutuhkan untuk menyelesaikan masing – masing nya adalah T(n/2)
T(1) = 1 = 1*1
T(2) = 2 * T(1) + 2 = 4 = 2 * 2
T(4) = 2 * T(2) + 4 = 12 = 4 * 3
T(8) = 2 * T(4) + 8 = 32 = 8 * 4
T(16) = 2 * T(8) + 16 = 80 = 16 * 5
T(32) = 2 * T(16) + 32 = 192 = 32 * 6
T(64) = 2 * T(32) + 64 = 484 = 64 * 7
$T(N) = N(1+\log N) = N + N \log N = O(N \log N)$
3. Tavg(n): kompleksitas waktu untuk kasus rata-rata (average case)
Adalah kebutuhan waktu secara rata-rata. Sehingga total waktu yang dibutuhkan (kompleksitas waktu) oleh algoritma RC4 adalah $O(\log n) + O(n \log N) = O(n \log n + \log n)$.

4. KEAMANAN ALGORITMA RC4 STREAMCIPHER

Permasalahan yang didapat dalam Algoritma RC4 Stream Cipher ini ialah sebagai berikut :

a) Permasalahan yang pertama dihadapi pada algoritma ini ialah terlalu tingginya kemungkinan terjadi S-Box yang sama karena nilai pseudorandom yang sama seringkali dibangkitkan secara berulang, hal ini terjadi karena kunci user diulang-ulang untuk mengisi 256 byte array. Meskipun metode ini memungkinkan penggunaan variabel yang panjangnya dapat mencapai 256 byte. Namun pada kenyataannya jarang sekali ada yang menngunakan kunci sepanjang itu. Selain karena sulit mencari kombinasinya juga sulit untuk mengingatnya. Sehingga jika kunci yang digunakan sebanyak 8 byte, maka kunci ini akan diulang sebanyak 32 kali untuk mengisi key byte array sampai penuh.
b) Enkripsi RC4 adalah XOR antara data bytes dan pseudorandom byte stream yang dihasilkan kunci, maka penyerang akan mungkin menentukan beberapa byte pesan orisinal dengan meng-XOR dua set cipher byte, bila beberapa byte plaintext diketahui (atau mudah ditebak). Diasumsikan A berhasil menyadap 2

buah message berbeda yang dienkripsi menggunakan algoritma stream cipher dengan menggunakan kunci sama. A kemudian meng-XORkan kedua chiphertext yang berhasil disadapnya untuk menghilangkan pengaruh rangkaian kunci. Jika A berhasil mengetahui plaintext dari salah satu message terenkripsi tersebut maka A akan dengan mudah menemukan plaintext yang lain tanpa mengetahui rangkaian kuncinya.

Untuk mengatasi permasalahan diatas, terdapat beberapa cara yang bisa dilakukan, antara lain :

a) Gunakan kunci yang panjang (minimal panjang kunci 3 karakter dan maksimal 255 karakter)
b) Usahakan untuk tidak menggunakan kunci yangsama untuk mengenkripsi file yang berbeda.
c) Mengacak (mengubah susunan) plaintext sebelum diubah ke dalam chipher, sehingga jika seorang penyerang memperoleh 1 byte data dari plaintext, maka ia tidak dapat memperoleh data yang lainnya dengan cara meng-XORkan dua buah ciphertext dan byte data yang ia ketahui.
d) Mengubah metode pengisian key ke dalam key array. Caranya adalah key cukup diisikan sekali dalam array kemudian sisa variabel array key yang lainnya akan diisi dengan nilai yang dibangkitkan secara random.

5. KESIMPULAN

Kesimpulan yang dapat diambil dari makalah ini adalah sebagai berikut :

1. Algoritma RC4 Stream chipher merupakan salah satu jenis stream cipher, yaitu memproses unit atau input data, pesan atau informasi pada satu saat. Unit atau data pada umumnya sebuah byte atau bahkan kadang kadang bit (byte dalam hal RC4). Dengan cara ini enkripsi atau dekripsi dapat dilaksanakan pada panjang yang variabel. Algoritma ini tidak harus menunggu sejumlah input data, pesan atau informasi tertentu sebelum diproses, atau menambahkan byte tambahan untuk mengenkrip.
2. Dalam proses enkripsi, terdapat 4 langkah penting, yaitu yang pertama ialah dengan melakukan inisialisasi S-Box Awal yang dalam makalah ini berjumlah 256 byte, kemudian dilakukan Insialisasi Array Of Key, kemudian Permutasi S-Box yang dinisialisasi dengan variabel input dari Array of Key. Dan Kemudian dilakukan proses Operasi XOR dengan Array S-Box yang sudah dipermutasi ini untuk menghasilkan ciphertext
3. Kompleksitas waktu untuk algoritma ini yang didasarkan pada perhitungan Big-O ialah O(n log n + log n)
4. Secara garis besar, kemungkinan algoritma ini diserang oleh pihak yang tidak berhak cukup besar sehingga diperlukan tindakan untuk mencegahnya, yaitu diantaranya ialah dengan menggunakan kunci yang memiliki karakter panjang, dan tidak menggunakan kunci yang sama untuk mengenkrip file yang berbeda

REFERENSI

[1] Juliasari, N., "Penerapan Teknik Enkripsi Blok dan RC4 Stream Cipher pada Database Nasabah Koperasi", Budi Luhur, 2002.

[2] Michael Howard, RC4 Usage Errors Leave Your Data Exposed,http://security.devx.com/bestdefense/2001/mh0201/mh0201-4.asp, diakses tanggal 18 Desember 2002.

[3] Menezes, A., Oorshoot, P.,Vanstone, S., "Handbook ofApplied Cryptography", Boca Raton, FL, CRC Press, 1997.

[4] Raharjo, Budi, "Keamanan Sistem Informasi Berbasis Internet", HandbookKeamanan, PT Insan Infonesia - Bandung & PT INDOCISC – Jakarta, 2004.