Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 155 - 161

Penerapan Algoritma RC6 dan Algoritma Elias Delta Code Pada Aplikasi Pengamanan dan Kompresi Short Message Service (SMS) Berbasis Android

Fadrid Jaya Immanuel Harefa

Program Studi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia
Email:fadridharefa@gmail.com

Abstrak−Pada penelitian ini dibangun suatu perangkat lunak yang dapat berguna untuk meningkatkan keamanan pesan yang terjadi pada komunikasi melalui SMS. SMS merupakan suatu layanan yang diberikan oleh telepon seluler kepada penggunanya untuk melakukan komunikasi melalui pengiriman pesan singkat dengan biaya yang murah. SMS sangan popular, selain dikarenakan biayanya yang murah, pesan yang dikirimkan dapat diterima oleh penerima dengan baik dan cepat. Dengan adanya kompresi dan pengamanan ini maka SMS berhasil dikirm dan diamankan dengan hasil bit yang lebih kecil dari SMS sebelumnya. Hasil dari kompresi dan keamanan dengan menggunkan Algoritma Elias Delta Code dan Algoritma RC6 munculnya karakter atau symbol yang akan sampai kepada nomor yang hanya dapat digunakan dengan android.

Kata Kunci: Kompresi, Keamanan SMS dan Elias Delta Code.

Abstract− In this study software was developed that could be useful to improve the security of messages that occur in communication via SMS. SMS is a service provided by cellular phones to its users to communicate through sending short messages at a low cost. SMS is very popular, aside from being cheap, messages sent can be received by the recipient well and quickly. With this compression and security, the SMS was successfully sent and secured with smaller bits than the previous SMS. The results of compression and security using the Elias Delta Code Algorithm and RC6 Algorithm the appearance of characters or symbols that will arrive at numbers that can only be used with Android.

Keywords: Compression, SMS Security and Elias Delta Code.

1. PENDAHULUAN

Perkembangan Teknologi pada masa sekarang ini telah mengalami perubahan yang sangat pesat. Seperti kita ketahui alat komunikasi salah satunya adalah Handphone (HP) menjadi kebutuhan utama bagi orang yang membutuhkan informasi dan komunikasi diseluruh penjuru dunia. Melalui Handphone yang berbentuk pesan teks dalam mengirim informasi, maka akan membutuhkan media penyimpanan data yang sangat besar. SMS (Short Message Service) merupakan pesan tulis yang ingin disampaikan pengirim pada penerima melalui media telepon maupun telepon genggam (Handphone). Maka keamanan dibutuhkan bagi setiap orang mengirim SMS pada tujuan untuk tidak ketahui orang lain.

Dengan adanya proses kompresi terhadap teks SMS maka akan terjadi pemampatan terhadap teks SMS sehingga dapat menghemat biaya pengiriman SMS. Pada saat melakukan pengiriman pesan pada tujuan dan tidak diketahui pihak lain, maka harus menggunakan algoritma Elias delta code karena algoritma tersebut akan mengkompresi data pada saat mengirim pesan. Sehingga dapat menghemat kapasitas penyimpanan dengan menggunakan Teknik kompresi bertujuan untuk menghasilkan yang ukuran lebih kecil data dari data aslinya[1]. Maka data yang sudah di kompresi harus diperlukan keamanan terhadap data dengan menerapkan algoriitma RC6. Dengan kunci privat yang dikenal dengan kesederhanannya., untuk aspek keamanannya akan digunakan pada proses enkripsi dan dekripsi dari algoritma RC6[2].

2. METODOLOGI PENELITIAN

2.1 Kompresi

Kompresi merupakan teknik memperkecil atau memadatkan ukuran data atau *file*, sehingga *file* yang berukuran besar menjadi lebih kecil dan mengurangi kebutuhan akan penyimpanan. Proses kompresi merupakan proses yang mengarah pada minimisasi jumlah bit *rate* untuk representasi digital seperti citra, video dan audio, sehingga menghasilkan ukuran suatu data yang padat dan mampat tetapi tetap menjaga kuantitas informasi dalam data tersebut. Kompresi citra adalah proses memperkecil atau meminimalisasi jumlah tiap bit yang merepresentasikan suatu citra dengan ukuran data citra menjadi lebih kecil[3]

Ada dua teknik yang dapat dilakukan dalam melakukan kompresi[4]

1. *Lossless Compression*
   Lossless compression merupakan teknik kompresi citra yang tidak menghilangkan informasi sebelumnya, dimana hasil dekompresi dari citra yang terkompresi sama dengan citra aslinya.
2. *Lossy Compression*
   Lossy compression merupakan teknik kompresi citra yang akan menghilangkan beberapa informasi, dimana hasil dekompresi dari citra yang terkompresi tidak sama dengan citra aslinya, tetapi masih bisa ditolerir oleh persepsi mata.

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 155 - 161

2.2 Algoritma RC6

Algoritma RC6 merupakan salah satu kandidat *Advanced Encryption Standard* (AES) yang diajukan oleh RSA Laboratoriest kepada NIST. Dirancang oleh Ronald L Rivest, Algoritma ini merupakan pengembangan dari Algoritma sebelumnya yaitu RC5 dan telah memenuhi semua kriteria yang diajukan oleh NIST. Algoritma RC6 adalah versi yang dilengkapi dengan beberapa parameter, sehingga dituliskan sebagai RC6 w/r/b, dimana parameter w merupakan ukuran kata dalam satuan bit, r adalah bilangan bulat bukan negatif yang menunjukan banyaknya iterasi selam proses enkripsi, dan b menunjukan ukuran kunci enkripsi dalam byte. Ketika algoritma ini masuk sebagai kandidat AES, maka di tetapkan nilai parameter w=32, r=20 dan b bervariasai antara 16,24, dan 32 byte[2].

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

1. A + B : penjumlahan integer modulo 2w
2. A + B : pengurangan integer modulo 2w
3. A ⊕ B : Operasi eksklusif OR (XOR) dari w-bit kata
4. A x B : Perkalian integer modulo 2w
5. A <<< B : Rotasi terhadap A ke kiri sebanyak nilai yang diperoleh dari bit-bit orde bawah sejumlah 2log(w) dari B
6. A >>> B : Rotasi terhadap A ke kanan sebanyak yang diperoleh dari bit-bit orde bawah sejumlah 2log(w) dari B

2.3 Algoritma *Elias Delta Code*

Elias delta code adalah sebuah algoritma kompresi yang dibuat oleh Elias menggunakan kode yang telah dia buat sebelumnya, yaitu Elias Gamma code, sebagai building block. Kode Gamma, Elias menambah Panjang kode dalam unary (α). dalam kode berikutnya, δ (delta), ditambahkan pada Panjang kode dalam biner (β). Dengan demikian, Elias Delta Code, yang juga untuk bilangan bulat positif, sedikit lebih kompleks untuk dibangun[1]

Adapun aturan untuk mengkodekan sebuah bilangan dengan menggunakan Elias Delta Code adalah sebagai berikut:

1. Tuliskan n dalam bilangan biner (binary). Bit yang paling kiri (paling signifikan) akan menjadi 1.
2. Hitung jumlah bit-nya, hapus bit paling kiri dari n dan tambahkan perhitungan dalam bilangan biner (binary) pada bagian kiri dari n setelah bit paling kiri dari n dihapus.
3. Hitung jumlah bit-nya, hapus bit paling kiri dari n dan tambahkan perhitungan dalam bilangan biner (*binary*) pada bagian kiri dari n setelah bit paling kiri dari n dihapus.
4. Kurangi 1 dari perhitungan pada langkah ke-2 dan tambahkan jumlah nol ke kode.

Ketika langkah-langkah ini diterapkan pada integer ke-17, hasilnya adalah:17=10001 (lima bit). Hapus angka 1 yang paling kiri dan tambahkan 5=101 sehingga hasilnya 101|0001. Tiga bit sudah ditambahkan, kemudian tambahkan 2 nol untuk mendapatkan kode delta 00|101|0001.

Decode dengan *elias delta code* dilakukan dalam langkah-langkah berikut :

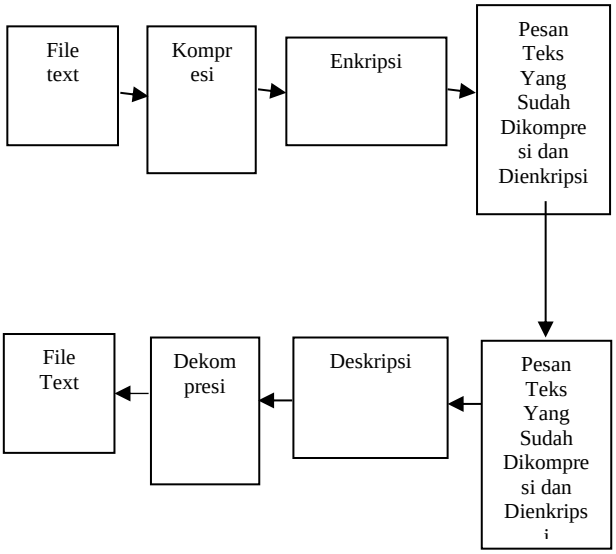
1. Baca bit dari kode sampai proses *decode* dengan *elias gamma code* dapat dilakukan. Proses ini dapat dilakukan dengan beberapa langkah berikut ini:
   a. Hitung jumlah nol terdepan dari kode tersebut lalu gantikan perhitungan tersebut dengan C.
   b. Periksa bit bagian kiri 2C + 1 (C nol, diikuti dengan 1, lalu diikuti dengan bit C selebihnya). Ini merupakan decode elias gamma code M + 1.
2. Baca bit M berikut. Sebut ini sebagai L.
3. Bilangan bulat yang di decode adalah 2M + L.

3. HASIL DAN PEMBAHASAN

Analisa merupakan tahap awal yang harus dilakukan untuk memecahkan masalah yang dihadapi. berbagai macam cara agar dapat menyimpan sejumlah data, informasi data dalam media penyimpanan yang terbatas. Oleh karena itu dibutuhkan teknik kompresi danpengamanan data. Salah satu metode yang cukup efisien adalah dengan menggunakan teknik kompresi dan Teknik kriptografi. Menerapkan teknik kompresi dan kriptografi dalam data akan membuat kerahasiaan data yang nantinya akan lebih terjaga.

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 155 - 161



Gambar 1. Prosedur Kompresi Dan RC6 Pesan Teks

3.1 Penerapan Algoritma RC6

Proses enkripsi dan dekripsi, dibutuhkan suatu key scheduling untuk mendapatkan kunci-kunci yang akan digunakan dalam proses enkripsi dan dekripsi tersebut. Algoritma RC6 terdiri dari 3 tahap utama, yaitu :

1. Penempatan kunci yang di-input pengguna kedalam array L.
2. Inisialisasi kunci yang ditempatkan dalam array S.
3. Kombinasi L dan S.

   Setelah mendapatkan kunci-kunci, blok data yang diolah sebesar 128bit dikelompokkan 4 buah blok. Masing-masing blok data berukuran 32bit kemudian disimpan dalam register A, B, C, dan D lalu dioperasikan dengan kunci-kunci berukuran 32 bit yang telah diperoleh dari algoritma RC6. Sebelumnya pada register A sampai dengan register A terisi banyak 4 byte. Dengan proses penempatan byte pada register B, C, dan D sehingga byte terakhir blok data terdapat pada byte paling kiri (most significant) register Secara umum enkripsi dengan menggunakan algoritma RC6 memiliki langkah-langkah terurut sebagai berikut:

1. *Whitening* awal
2. Iterasi
3. *Whitening* akhir

Dekripsi algoritma RC6 merupakan kebalikan dari proses enkripsi. Ukuran blok tersebut sama dengan ukuran matriks kunci yang telah ditentukan. Proses enkripsi algoritma RC6 mempunyai langkah-langkah yang dilakukan dalam proses enkripsi adalah dengan cara mengenkripsi terlebih dahulu pesan teks dan menghasilkan nilai untuk diproses. Pada enkripsi algoritma RC6 ini diberikan kunci sebesar 4 byte dan plain 128 bit (16 *byte*), kunci dan plaim yang menjadi contoh masing-masing adalah sebagai berikut: “fadr”

Plain; “70 65 68 82 73 68 68 73 77 65 78 65 84 73 78 71”

Pembangunan kunci internal dalam proses enkripsi dengan algoritma RC6 terdiri dari tiga tahap yaitu :

1. Konversi kunci rahasia dari bytes ke words

   Langkah pertama adalah menyalin kunci rahasia K [0..b-1] kedalam sebuah array, L [0.c-1], dimana c = pembulatan keatas (b/u) dan u =w/8 penyalinan tersebut dilakukan secara little endian. Untuk kasus dimana b = 0, maka c = 1dan L [0] = 0 dalam kasus ini user memasukan kunci berupa kata “riof” sehingga pada prosesnya adalah sebagai berikut:

   If C = 0 then
   C = 1
   End if
   For I = b – 1 downto 0 do
   L [i/u] = (L [i/u] << 8) + K [i]
   End for

   Dengan input K [0] = “**fadr**” makan nilai L yang di dapat adala :

   L [0] = “01000110010000010100010001010010”

2. Inisialisasi Array S

   Insialisasi array S yang nantinya akan digunakan sebagai kunci internal agar memiliki pola pseudo – randombit tertentu menggunakan progresi arimatika modulo 2w yang ditentukan dengan P32 dan Q32. Berikut adalah proses inisialisasi array S ini:

   S [0] = P32

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 155 - 161

For I = 0 to 2 r + 3 do
S [i] = S [i-1] + Q32
Dengan proses ini, inisialisasi S[ ] yang dihasilkan adalah sebagai berikut:
S [0] = B7E15163
For I = 0 to 42 do
S [0] = B7E15163
S [1] = S [1-1] + 9E3779B9
S [1] = S [0] + 9E3779B9
S [1] = B7E15163 +
S [1] = 15618CB1C
S [2] = 15618CB1C + 9E3779B9
S [2] = 1F45044D5
S [3] = 1F45044D5 + 9E3779B9
S [3] = 29287BE8E
S [4] = 29287BE8E + 9E3779B9
S [4] = 330BF3847
S [5] = 330BF3847 + 9E3779B9
S [5] = 3CEF6B200
.
.
S [43] = 1AACFB49BD + 9E3779B9
S [43] = 1B4B32C376

Langkah pertama adalah membagi plainteks ke dalam 4 *block* yaitu A, B, C, D yang masing-masing terdiri dari 32 bit.

Tabel 1. Pembagian Plainteks Kedalam 4 Block

Blok Data	Bilangan Biner	Register
70	01000110	
65	01000001	A
68	01000100	01000110010000010100010001010010
82	01010010	
73	01001001	
68	01000100	B
68	01000100	01001001010001000100010001001001
73	01001001	
77	01001101	
65	01000001	C
78	01001110	01001101010000010100111001000001
65	01000001	
84	01010100	
73	01001001	D
78	01001110	01010100010010010100111001000111
71	01000111	

Setelah dapat nilai pada masing-masing blok, maka dilanjutkan dengan langkah-langkah berikut :

1. *Whitening* awal
Whitening awal, dengan menjumlahkan B dengan subkunci S (0), dan D dengan subkunci S(1).
B = B+ S (0)
= 01001001010001000100010001001001
01011011111000001001011110101101 +
B = 00010010101001001101001111100100
D = D + S (1)
= 01010100010010010100111001000111
11110010101111111110001001001001 +
D = 10100110111101101010110000001110

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 155 - 161

2. Iterasi
   Iterasi dilakukan sebanyak 20 kali. Setiap iterasi mengikuti aturan sebagai berikut :
   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 [2] = (ROTL ((X [2] XOR t), t)) + S [2 * I + 1]
   Tem = x [0]
   x [0] = x [1]
   x [1] = x [2]
   x [2] = x [3]
   x [3] = Temp
   nilai t dan u di dapat dari blok B dan D di proses dengan fungsi f(x) = x(2x+1), kemudian dilanjutkan dengan menggeser nilai t dan u ke kiri sejauh 5 bit
   t = B* (2 * B + 1) <<<5
   = 00010010101001001101001111100100
   00100101010010011010011111001001 *
   00000000000000001000001111000000<<<5
   00000000000100000111100000**00000**

   Nilai 5 bit terakhir dari t yaitu 00000, atau dalam decimal sebesar 1, akan digunakan untuk pergeseran nilai blok C pada proses berikutnya sejauh 1 bit
   u = D * (2 * D + 1) <<<5
   u = 10100110111101101010110000001110
   01001101111011010101100000011101 *
   00000100111001000000100000001100<<<5
   10011100100000010000000110000000

   Nilai bit terakhir dari u yaitu **00000**, atau dalam decimal sebesar 24 bit, akan digunakan untuk pergeseran nilai blok A pada proses berikutnya sejauh 24 bit. Langkah selanjutnya adalah memproses blok A dan C dengan nilai yang telah dihasilkan
   A = A XOR t <<<u + S [2]
   A = 01000110010000010100010001010010
   10000000001100000010000000000001 ⊕
   11000110011100010110010001**10011** <<<10011 + S [2]

   C = ROTL C XOR u <<<t +S [3]
   C = 01001101010000010100111001000001
   00000000111011000001100000001100 ⊕
   01001101101001010100011001**01101** <<<01101 + S [3]

   Maka di dapat nilai masing-masing blok adalah :
   A = 11000110011100010110010001**10011**
   B = 00010010101001001101001111**00100**
   C = 01001101101001010100011001**01101**
   D = 10100110111101101010110000**01110**
   Langkah berikutnya adalah mempertukarkan (A, B, C, D) = (B, C, D, A) sehingga nilai hasil sebagai berikut :
   A = 00010010101001001101001111**00100**
   B = 01001101101001010100011001**01101**
   C = 10100110111101101010110000**01110**
   D = 11000110011100010110010001**10011**
3. Whitening akhir
   Dalam proses whitening akhir ini menjumlahkan nilai dari blok A dengan sub kunci S [2 * r + 2] dan blok C dengan S [2 & r + 3]
   X [0] = X [0] + S [42].
   A = 00010010101001001101001111100100
   11111010110010110011000010000000 +
   11101000011011111110001101100100
   x [2] = x [2] +S [43]
   C = 10100110111101101010110000001110
   01010111010010010000101100001110 +
   11110001101111111010010000000000
   Sehingga dihasilkan chiperteks seperti dibawah ini :
   20 229 106 245 24 86 168 149 34 178 191 189 144 112 42 211

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 155 - 161

3.2 Implementasi

Pengujian sistem merupakan tahap dimana sistem akan diuji untuk mengidentifikasi apakah sudah sesuai dengan apa yang dirancang pada tahap analisis dan perancangan sistem.pada tahapan ini merupakan uji coba dari hasil program yang dibuat, mulai dari buat pesan baru untuk dikompresi dan dikirim dan membuka sms masuk yang sudah terkompresi.

1. Buat pesan
   Berikut tampilan dari buat pesan yang akan dikompresi dan dikirim ke nomor tujuan.

Gambar 2. Tampilan pesan yang akan dikirim

2. SMS Masuk
   Berikut dari tampilan dari SMS masuk yang telah diterima pada tujuan yang akan dienkripsi dan didekompresi.

Gambar 3. Tampilan SMS masuk yang telah dienkripsi dan didekompresi.

4. KESIMPULAN

Berdasarkan hasil analisa maka penulis dapat menarik kesimpulan, dimana kesimpulan tersebut nanti dapat kiranya berguna bagi pembaca, sehingga penulisan penelitian ini berguna dan memanfaat. Kesimpulan-kesimpulan antara lain sebagai berikut :

1. Dengan adanya kompresi dan pengamanan ini maka SMS berhasil dikirim dan diamankan dengan hasil bit yang lebih kecil dari SMS sebelumnya.
2. Hasil dari kompresi dan keamanan dengan menggunakan Algoritma Elias Delta Code dan RC6 munculnya bentuk karakter atau symbol yang akan sampai kepada nomor yang ditujuh dan akan didekompresi ataupun dideskripsi oleh penerima.

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

3. Pengguna dari Aplikasi SMS yang menggunakan Algoritma Elias Delta Code dan RC6 hanya dapat digunakan dengan android...

REFERENCES

[1] D. Salomon, D., "Handbook of Data Compression," pp. 6–22, 2010.
[2] R. W. A. Permana, "Implementasi Algoritma RC6 Untuk Enkripsi SMS Pada Telepon Selular," pp. 4–23, 2013.
[3] Y. Laia, "Optimasi Rasio Kompresi Dan Kompleksitas Waktu Kompresi File Teks Menggunakan Algoritma Lempel-ZIV-Welch Dengan Fibonacci Search," vol. 1, pp. 61–66, 2016.
[4] M. K. T. Sutoyo, S.Si., Teori Pengolahan Citra Digital. Semarang, 2009.
[5] M. Romzi, Logika dan Algoritma, no. tahun 1736. 2012.
[6] B. C. Neyfa and D. Tamara, "Perancangan Aplikasi E-Canteen Berbasis Android Dengan Menggunakan Metode Object Oriented Analysis & Design (OOAD)," J. Penelit. Komun. dan Opini Publik, vol. 20, no. 1, pp. 83–92, 2016.
[7] R.Sahara, "Implementasi Keamanan SMS Dengan Algoritma RSA pada Smartphone Android, "vol IX, no 2, pp 118-122,2017.
[8] M. Ikhsan, "Sistem Keamanan SMS (Short Message Service) pada Jaringan Selular dengan Peningkatan Fungsionalitas Menggunakan Internet," 2012.
[9] A. Jogiyanto, Analisis dan Desain Sistem Informasi. Yogyakarta, 2005.
[10] J. Eclipse, "Pembuatan Aplikasi Android Sederhana dengan Eclipse," pp. 1–6, 2015.

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**