TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Implementasi Algoritma RC4+ dan Metode Spread Spectrum Untuk Mengamankan Pesan Rahasia Kedalam Citra Digital

Otriyani Zalukhu

Program Studi Teknik Informatika, Universitas Budi Darma Medan, Indonesia
Email: milyani209@gmail.com

Abstrak−Perkembangan teknologi informasi saat ini saat melesat luas sehingga sangat mudah untuk melakukan pertukaran data atau informasi. Di lain sisi, akan banyak timbul masalah keamanan data yang dilakukan oleh orang-orang yang tidak bertanggung jawab dengan cara penyadapan, perusakan, pencurian, pemalsuan data, atau dengan cara lainnya. Teknik kriptografi dan steganografi dua alat yang menawarkan keamanan data. Kriptografi menjamin kerahasiaan, keaslian dan integritas data sedangkan steganogafi bertujuan untuk menyembunyikan dan mengambil data. Algoritma RC4+ merupakan salah satu algoritma kriptografi hasil variasi dari algoritma asimetris RC4 yang hanya menggunakan satu kunci untuk melakukan proses enkripsi dan dekripsi. Metode spread spectrum merupakan metode steganogafi yang sering digunakan untuk melindungi hak cipta atas produk digital dengan cara pengacakan pesan. Penelitian ini menguraikan tentang pengamanan pesan rahasia kedalam bentuk citra digital dengan menggabungkan dua teknik pengamanan. Dimana RC4+ bertujuan untuk menjaga keaslian data dengan cara melakukan proses enkripsi dan dekripsi, sedangkan metode spread spectrum berfungsi untuk menjaga keutuhan hak cipta dari pesan citra digital tersebut. Berdasarkan hasil pengujian yang dilakukan dengan algoritma RC4+ dengan proses enkripsi dan dekripsi serta metode spread spectrum pada aplikasi matlab yang dibuat dapat berjalan dengan baik sesuai dengan perancangan awal. Kedua metode pengamanan yang digunakan menjamin keutuhan data atau informasi hingga sampai ketangan penerima pesan rahasia tersebut.

Kata Kunci: Kriptografi; Steganografi; RC4+; Spread Spectrum; Citra Digital

Abstract−The current development of information technology is soaring widely that it is very easy to exchange data or information. On the other hand, there will be many data security problems committed by irresponsible people by means of wiretapping, destroying, stealing, falsifying data, or by other means. Cryptographic techniques and steganography are two tools that offer data security. Cryptography guarantees the confidentiality, authenticity and integrity of data while steganogafi aims to hide and retrieve data. The RC4 + algorithm is a variation of the RC4 asymmetric algorithm, which uses only one key to perform the encryption and decryption process. The spread spectrum method is a strategic method that is often used to protect copyright on digital products by means of random messages. This study describes the security of secret messages in the form of digital images by combining two security techniques. Where RC4 + aims to maintain the authenticity of data by means of encryption and decryption processes, while the spread spectrum method serves to maintain the integrity of the copyright of the digital image message. Based on the results of tests carried out with the RC4 + algorithm with encryption and decryption processes as well as the spread spectrum method in the Matlab application that was made to run well in accordance with the initial design. The two security methods used ensure the integrity of the data or information until it reaches the hands of the recipient of the secret message.

Keywords: Cryptography; Steganography; RC4+; Spread Spectrum; Digital Image

1. PENDAHULUAN

Seiring dengan majunya perkembangan teknologi, banyak tersedia berbagai macam teknik untuk dapat melindungi pesan atau informasi yang dirahasiakan dari orang yang tidak berhak untuk mengakses pesan tersebut seperti pencurian atau percobaan *hacking*. Dengan berkembangnya teknologi sering kita mendengar atau melihat bahwa pesan-pesan rahasia yang ingin kita kirimkan kepada seseorang telah dilakukan pencurian atau penyadapan oleh orang yang tidak berhak atau orang yang tidak bertanggung jawab. Maka dari itu dalam pengiriman/penerimaan informasi, pengguna membutuhkan suatu jaminan keamanan yang dapat meyakinkan bahwa yang diperoleh adalah informasi yang aman dan benar. Sehingga dapat mempersulit para pihak yang tidak bertanggung jawab akan kejahatan komputer.

Pada saat ini telah banyak teknik yang dilakukan untuk menjaga keamanan data dan informasi, seperti kriptografi dan steganografi. Kriptogarafi adalah ilmu mengenai teknik enkripsi dimana data diacak menggunakan suatu kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yang tidak memiliki kunci dekripsi. Dekripsi menggunakan kunci dekripsi mendapatkan kembali data asli. Proses enkripsi dilakukan menggunakan suatu algoritma dengan beberapa parameter [1].

Steganografi adalah seni untuk menyisipkan pesan rahasia kedalam suatu media, dimana pesan rahasia yang akan disembunyikan tidak diubah bentuknya, melainkan disisipkan pada sebuah citra digital, sehingga orang lain tidak mengetahui bahwa didalam citra digital tersebut ada pesan rahasia. Metode steganografi yang digunakan adalah Metode spread sprectum. Metode ini merupakan penyembunyian pesan yang memperlakukan cover-object baik sebagai derau (noise) ataupun sebagian usaha untuk menambahkan derau semu (pseudonoise) kedalam cover-object.

RC4+ adalah algoritma simetris hasil modifikasi dari algoritma *RC4* yang diajukan oleh Maitra dan Paul pada tahun 2008. Agar sandi pada algoritma *RC4+* lebih kuat, maka ditambahkan beberapa operasi menggunakan struktur seperti *RC4*. Keamanan yang lebih baik diperoleh dengan memanfaatkan poin yang ada pada *RC4* kemudian memberikan beberapa fitur tambahan [2]. Kelebihan dari *RC4+* adalah dapat melakukan enkripsi dengan cepat dan efisien sehingga bisa mempersingkat waktu saat melakukan enkripsi [**3**]. Metode *spread spectrum* merupakan sebuah teknik transimisi dimana kode *pseudonoise, independent* dari data informasi yang digunakan sebagai gelombang modulasi untuk menyebarkan energi sinyal melalui sebuah *bandwidth* jauh lebih besar dari pada *bandwidth* sinyal

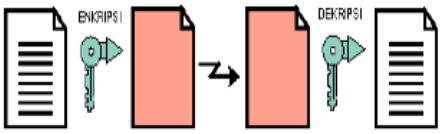
TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

informasi. Metode *spread spectrum* sangat terjamin keamanannya karna menggunakan replika kode *pseudonoise* yang telah disinkronisasikan.

2. METODOLOGI PENELITIAN

2.1 Kriptografi

Secara etimologi, kriptografi berasal dari bahasa Yunani yaitu "kriptos" dan "graphia". Kriptos dapat diartikan sebagai rahasia, sedangkan graphia dapat diartikan sebagai tulisan. Kriptografi merupakan ilmu yang digunakan untuk mempelajari tulisan rahasia dimana komunikasi dan data dapat dikodekan dan berfungsi mencegah orang yang tidak berwenang untuk memanipulasi informasi melalui sebuah teknik sehingga hanya pihak berwenang saja yang dapat mengetahui isi informasi tersebut [4]. Kriptografi adalah ilmu mengenai teknik enkripsi dimana data diacak menggunakan suatu kunci enkripsi menjadi sesuatu yang sulit dibaca oleh seseorang yang tidak memiliki kunci dekripsi. Dekripsi menggunakan kunci dekripsi mendapatkan kembali data asli. Proses enkripsi dilakukan menggunakan suatu algoritma dengan beberapa parameter [1**]**.



Gambar 1. Konsep Penyandian Kriptografi

Berdasarkan pengertian diatas dapat disimpulkan bahwa kriptografi adalah suatu ilmu dan seni pengamanan data atau pesan dengan cara menyandikan pesan teks dalam bentuk simbol yang sulit dipecahkan dan membutuhkan waktu yang lama dalam memecahkannya. Kriptografi masih merupakan sistem yang efektif dalam hal keamanan dan proteksi. Keamanan pesan data dapat terjaga dengan aman karena dibutuhkan waktu yang lama.

2.2 Algoritma RC4+

RC4+ merupakan salah satu jenis dari algoritma *RC4*. Dimana algoritma *RC4* adalah salah satu algoritma kunci simetris yang berbentuk *stream cipher* yang melakukan proses enkripsi atau dekripsi dalam satu byte dan menggunakan kunci yang sama. *Stream cipher* digunakan untuk mengenkripsi *plaintext* menjadi *ciphertext* bit per bit (1 bit setiap kali transformasi) atau byte per byte (1 karakter = 1 byte). *RC4* menggunakan variabel yang panjang kuncinya dari 1 sampai 256 byte yang digunakan untuk menginisialisasikan tabel sepanjang 256 byte [1]. Algoritma *RC4+* memiliki dua bagian utama, yaitu *Key Scheduling Algorithm (KSA)* dan *Pseudo- Random Generation Algorithm (PRGA)* [1]. Struktur algoritma *RC4+* sama seperti struktur algoritma *RC4* di mana kedua algoritma tersebut memiliki *Key Scheduling Algorithm (KSA)*. Adapun *Key Scheduling Algorithm (KSA)* dari algoritma *RC4+* [1**]** sebagai berikut:

F or i=0 to 255
    S[i] :=i
end for
j :=0
for i from 0 *to 255*
    j := (j + S [i] + key [i mod keylength] mod 256
    swap values of S[i] and S [j]
    j: = j
end for.

Key Scheduling Algorithm (KSA) dari algoritma *RC4+*, dapat dilihat bahwa *i* dan *j* merupakan variabel awal bernilai 0 dan S adalah semua permutasi 256 yang mungkin terjadi [1].

Pseudo Random Generation Algorithm (PRGA+) dari algoritma *RC4+* berbeda dengan algoritma RC4. Berikut adalah langkah-langkah dari *PRGA+* adalah sebagai berikut [1] :

Input : Key-dependent scramblesd permutasi array S [0...N-1].
Output : Pseudo-random keystream bytes z.
i = 0; j= 0;
For i = 0 to Panjang Plain -1
    i : = i+1
    a : = S[i]
    j : = (j+a) mod 256
    swap S[i] and S[j]
    b : = S[j]; S[i] : = b ; S[j]: = a
    c := S[i<<5 ⊕ j>>3] + S[j<<5 ⊕ i>>3]
    z : = (S[a+b] + S [c⊕0xAA]) ⊕ S [j+b]
endfor.

TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

i dan j adalah indeks *array* 8-bit, << pergeseran ke kiri dan >> pergeseran ke kanan, ⊕ adalah eksklusif OR. Adapun proses enkripsi dan dekripsi pada algoritma RC4+ ini [1] adalah sebagai berikut :

$Ci = Pi \oplus Ki$ (1)

Formula Dekripsi

$Pi = Ci \oplus Ki$ (2)

2.3 Steganografi

Steganografi (steganography) adalah ilmu dan seni menyembunyikan pesan didalam pesan lain sehingga keberadaan pesan yang pertama tidak diketahui. Steganografi sangat kontras dengan kriptografi. Kriptografi merahasiakan makna pesan sementara eksistensi pesan tetap ada, sedangkan steganografi menutupi keberadaan pesan. Steganografi dapat dipandang sebagai ekalnjutan dari kriptografi, dalam prakteknya pesan dienkripsi terlebih dahulu, kemudian disembunyikan didalam media lain sehingga pihak ketiga tidak menyadari keberadaan pesan tersebut [7].

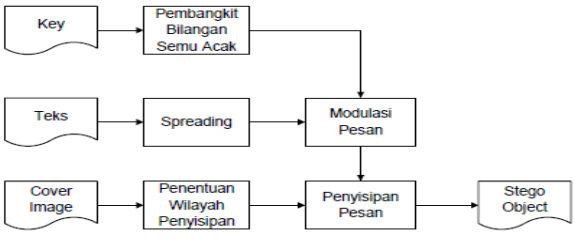
Penyembunyian data rahasia ke dalam citra digital akan mengubah kualitas citra tersebut. Kriteria yang harus diperhatikan dalam penyembunyian data adalah sebagai berikut [1] :

1. *Imperceptibility*. Keberadaan pesan rahasia tidak dapat dipersepsi oleh inderawi, misalnya jika *convertext* berupa citra, maka penyisipan pesan membuat citra *stegotext* sukar dibedakan oleh mata dengan citra *covertext* nya, jika *covertext* berupa audio (misal: MP3, wav, midi dan lain-lain) maka indera pendengaran (telinga) tidak dapat mendeteksi perubahan pada audio *stegotext*nya.
2. *Fidelity.* Mutu media penampung tidak berubah banyak akibat penyisipan. Perubahan tersebut tidak dapat dipersepsikan oleh inderawi.
3. *Recovery*. Data yang disembunyikan harus dapat diungkapkan kembali (*reval*). Karena tujuan steganografi adalah data *hiding*, maka sewaktu-waktu pesan rahasia di dalam *stegotext* harus dapat diambil kembali untuk digunakan lebih lanjut.

2.4 Metode Spread Spectrum

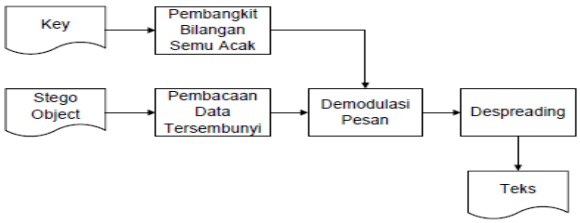
Spread spectrum merupakan bagian dari ranah *transform*. Sebuah teknik pentransmisian dengan menggunakan *pseudo-noise code*, yang independen terhadap data informasi, sebagai modulator bentuk gelombang untuk menyebarkan energi sinyal dalam sebuah jalur komunikasi (*bandwith*) yang lebih besar daripada sinyal jalur komunikasi informasi. Oleh penerima, sinyal dikumpulkan kembali menggunakan replika *pseudo-noise code* tersinkronisasi.
Berdasarkan definisi dapat dikatakan bahwa steganografi menggunakan metode *spread spectrum* memperlakukan *cover-object* baik sebagai derau (*noise*) ataupun sebagai usaha untuk menambahkan derau semu (*pseudo-noise*) ke dalam *cover-object* [8].

Di dalam metode *spread spectrum*, penyisipan pesan atau informasi terdapat kunci atau *key* yang digunakan untuk mengenkripsi pesan. *Key* tersebut kita dapatkan melalui pembangkit bilangan semu acak dengan algoritma LCG (*Linear Conruential enerator*). Sebelum pesan disisipkan kedalam cover image, maka terlebih dahulu menentukan wilayah penyisipannya. Setelah menentukan wilayah penyisipan, selanjutnya adalah proses *spreading*. Proses *spreading* dilakukan sesuai dengan bilangan pengali skalar yang ditentukan. Pada proses ini citra rahasia diambil nilai intensitas per-pixel nya, lalu diubah kedalam bilangan biner. Kemudian bilangan biner tersebut disebar sesuai bilangan pengali skalar yang telah ditentukan, maka hasil keluaran dari proses spreading ini adalah deret bilangan biner yang telah tersebar dengan panjang setiap deretnya sebesar 32 bit [8].
Skema penyisipan pesan dapat dilihat pada Gambar 2 berikut:



Gambar 2. Skema Penyisipan Pesan

Skema proses ekstraksi dapat dilihat pada gambar 3 sebagai berikut [8] :



Gambar 3. Skema Proses Ekstraksi Pesan.

TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

2.5 Citra Digital

Istilah lain dari gambar adalah citra digital dan merupakan bagian dari komponen multimedia yang sangat penting peranannya dalam menghasilkan informasi visual. Gambar atau citra digital kaya dengan informasi sehingga karakteristik inilah yang membedakannya dengan teks. Citra di bagi menjadi dua jenis yaitu citra kotinu dan citra diskrit. Citra yang dihasilkan melalui proses digitalisasi terhadap citra kontinu di sebut dengan citra diskrit atau citra digital (*digital image*) [9]. Citra digital adalah citra yang dapat diolah komputer. Beberapa format citra digital yang banyak ditemui adalah BMP, JPEG, GIF, PNG, dan lain-lain [10]. Citra digital adalah citra yang dapat diolah oleh komputer. Istilah citra digital sangat popular pada masa sekarang. Banyak peralatan elektronik, misalnya *scanner*, kamera digital, mikroskop digital, dan *fingerprint reader* (pembaca sidik jari), yang menghasilkan citra digital juga sangat popular digunakan oleh pengguna untuk mengolah foto atau untuk berbagai keperluan lain [1].

3. HASIL DAN PEMBAHASAN

Analisa adalah proses menentukan kebutuhan aplikasi, apa yang harus dilakukan pada untuk memenuhi kebutuhan klien (*user*). Dengan adanya analisis aplikasi, aplikasi yang dirancang akan lebih baik dan memudahkan pengembangan aplikasi dalam perbaikan apabila pada kemudian hari ditemukan kesalahan atau kekurangan. Berikut merupakan tahapan enkripsi dan dekripsi terhadap *plaintext* menjadi *ciphertext* pada algoritma *RC4+*. *Key* yang dimasukkan akan diacak dengan *Key Scheduling Algorithm (KSA)* sebanyak 256 kali. Lalu dilanjutkan dengan *Pseudo Random Generation Algorithm (PRGA)* sepanjang *plaintext* untuk mendapatkan hasil *keystream* yang akan di *xor* dengan *plaintext*.

3.1 Proses Enkripsi Algoritma RC4+

Inisalisasi *state* awal tersebut merupakan langkah pertama dari *key scheduling*. Berikut inisialisasi *state* awal dari *Key Scheduling Algorithm* (KSA) berupa larik 256 elemen.
Dapat dilihat pada contoh kasus dibawah ini tentang algoritma *RC4+* dengan metode enkripsi di bawah ini :

Plainteks : [O, T, R, I]
Plainteks : [79, 84, 82, 73] Dalam *ASCII*
Index : [0, 1, 2, ... 225]
Kunci = [Z, A, L]
Kunci = [90, 65, 76] Dalam *ASCII*

Jumlah karakter *plaintext* 4 (*length of plaintext)*
Jumlah karakter kunci 3 (*length of key)*

Kunci akan di-*generate* dengan *Key Scheduling Algorithm (KSA)* dan *Pseudo Random Generation Algorithm (PRGA)* adalah sebagai berikut :

a. Tahap *Key Scheduling Algorithm* RC4+

Dengan nilai *i* = 0 hingga 255, lakukan perhitungan nilai *j* yang pertama. Proses perhitungan sebagai berikut :
Nilai *i = 0*

1. *j = (j + S[i] + key[i mod keylength]) mod 256*
   j = (0 + S[0] + key[0]) mod 256
   j = (0 + 0 + 90) mod 256
   j = 90 mod 256
   j = 90

   Nilai *S[i] swap* dengan nilai *S[j]* nilai *S*[0] ditukar dengan *(swap)* dengan nilai *S[90]*, maka nilai dari *S[0] = 90* dan nilai dari *S[90] = 0*. Kemudian nilai *i = 1*, lakukan perhitungan nilai *j* yang kedua yaitu *j = 90* Proses perhitungan sebagai berikut :
2. *j = (j + S[i] + key[i mod keylength]) mod 256*
   j = (90 + S[1] + key[1 mod 3]) mod 256
   j = (90+ 1 + key[1]) mod 256
   j = (90 + 1 + 65) mod 256
   j = (156) mod 256
   j = 156

   Tukarkan nilai *S[1]* dengan *S[156]*, maka nilai dari *S[1] = 156* dan nilai dari *S[156] = 1*. Kemudian nilai *i = 2*, lakukan perhitungan nilai *j* yang kedua yaitu *j = 90*, maka ingat bahwa nilai *j* pada saat *i=1* adalah *156* maka proses perhitungan sebagai berikut :
3. *j = (j + S[i] + key[i mod keylength]) mod 256*
   j = (156 + S[2] + key[1 mod 3]) mod 256
   j = (156+ 2 + key[1]) mod 256
   j = (156 + 2 + 76) mod 256
   j = (234) mod 256
   j = 234

TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Tukarkan nilai *S[2]* dengan *S[234]*, maka nilai dari *S[2] = 234* dan nilai dari *S[234] = 2*. Ulangi langkah 3 hingga indeks *i = 255*, kemudian *array* s_1 teracak sebanyak 256 kali. Nilai-nilai pada *state* tersebut masih bersifat sementara hingga *i = 255*. Nilai *state* akhir didapatkan apabila *i* telah mencapai nilai 255.

b. Tahap *Pseudo Random Generation Algorithm RC4+*

Pada tahap awal inisialisasikan nilai *i* = 0 dan *j* = 0, Lakukan inisialisasi pada nilai *i* dan *j* sama dengan 0. Kemudian inisialisasikan *a* = *S*[*i*] dan *b* = *S*[*j*]. Selanjutnya lakukan perhitungan nilai *i*, *j*, *a*, *b*, *c* dan *z* yang baru dengan cara :

Untuk *i = 0*

i = (i + 1) mod 256 = (0 + 1) mod 256 = 1
a = S[i] = S[1] = 70
j = (j + a) mod 256 = (0 + 70) mod 256 = 70

Selanjutnya tukarkan nilai *S*[*i*] dan *S*[*j*] dengan cara sebagai berikut :

b = S[j] = S[70] = 26
S[i] = b = S[1] = 26
S[j] = a = S[70] = 70

Dari perhitungan diatas didapat S[1] = 26 dan S[70] = 70. Berdasarkan proses sebelumnya yang didapatkan nilai *i* dan *j* adalah sebagai berikut :

i : 1 dan *j :70*

Selanjutnya hitung nilai *c* dengan cara :

$c = (S[(i \ll 5) \ (j \gg 3)) \bmod 256] + S[(j \ll 5) \oplus (i \gg 3)) \bmod 256]) \bmod 256$
$c=(S[((1 \ll 5) \oplus (70 \gg 3)) \bmod 256] + S[((70 \ll 5) \oplus (1 \gg 3)) \bmod 256]) \bmod 256$
$c = (S[(32 \oplus 0) \bmod 256] + S[(2240 \oplus 0) \bmod 256]) \bmod 256$
$c = (S[(32) \bmod 256] + S[(2240) \bmod 256]) \bmod 256$
$c = (S[32] + S[192]) \bmod 256$
$c = (144 + 110) \bmod 256$
$c = 254 \bmod 256$
$c = 254$

Setelah mendapatkan nilai *c*, selanjutnya hitung nilai *z* dengan cara sebagai berikut:

i : 1 dan *j :70*

a = S[i] = S[1] = 70
j = (j + a) mod 256 = (0 + 70) mod 256 = 70
S[i] = b = S[1] = 26
S[j] = a = S[70] = 70

Nilai *c : 254*

Mencari nilai z.

$z=((S[(a+b) \bmod 256] + S[((c \oplus 170) \bmod 256)]) \oplus S[(j+b) \bmod 256]) \bmod 256$
$z=((S[(70+26) \bmod 256] + S[((254 \oplus 170) \bmod 256)]) \oplus S[(70+26) \bmod 256]) \bmod 256$
$z=((S[(96) \bmod 256] + (254 \oplus 170) \bmod 256)]) \oplus S[(96) \bmod 256]) \bmod 256$
$z=((S[(96) + S[((254 \oplus 170) \bmod 256)]) \oplus S[(96) \bmod 256]) \bmod 256$
$z = ((S[96] + S[168]) \oplus S[96]) \bmod 256$
$z = ((237 + 96) \oplus 237) \bmod 256$
$z = ((333) \oplus 237) \bmod 256$
$z = 416 \bmod 256$
$z = 160$

Nilai z ini akan menjadi nilai kunci *[0]* untuk mengekripsi plainteks.

Plainteks yang diamankan : O

Plainteks [0] O : 79		01001111
Kunci [0]	: 160	10100000 ⊕
Cipherteks [0]: 239		11101111
Cipherteks [0]: Ï		(dalam *ASCII*)

Inisialisasi pada nilai *i* = 1 dan *j* = 70. Dilakukan hal yang sama seperti pada pencarian nilai kunci O sampai dengan kunci yang terakhir kunci I.

Jadi hasil yang didapat adalah sebagai berikut :

Plainteks : [O, T, R, I]
Plainteks : [79, 84, 82, 73] (dalam *ASCII)*
Cipherteks : [239, 124, 211, 197]
Cipherteks : [Ï, |, Ó, Å] (dalam *ASCII)*

3.2 Proses Metode Spread Sprectrum

TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Misalkan *text* berisikan dari hasil cipherteks dari proses enkripsi algoritma *RC4+* [“Ï, |, Ó, Å”] (dalam *ASCII*) [239, 124, 211, 197]. Dalam biner (11101111, 01111100, 11010011, 11101001), kemudian biner pesan disebarkan dengan besaran skalar pengalinya empat, sehingga akan menghasilkan segmen baru yaitu :

1111 1111 1111 0000 1111 1111 1111 1111
0000 1111 1111 1111 1111 1111 0000 0000
1111 1111 0000 1111 0000 0000 1111 1111
1111 1111 1111 0000 1111 0000 0000 1111

Kemudian langkah selanjutnya adalah pembangkitan pseudonoise dengan bibit pembangkitan yang ditentukan berdasarkan kata kunci yaitu “otri”.

o = 01101111
t = 01110100 xox
00011011
r = 01110010xox
01101001
i = 01101001 xox
00000000
z = 01111010 xox
01111010 jadi, kuncinya 010 (desimal)

Setelah mendapatkan nilai dari kata kunci (010), nilai tersebut digunakan sebagai bibit awal pembangkitan bilangan acak. Perhitungan pembangkitan bilangan acak sesuai dengan rumus pembangkitan bilangan acak LCG adalah sebagai berikut:

$$Xn+1 = (aXn + c) \bmod$$

Keterangan :
Xn = bilangan bulat ke-n
a = bilangan pengali
m = modulus

Jika nilai a = 17, c = 7, m = 84, dan Zi□1 = 010, perhitungan bilangan acaknya adalah:

X1 = (17 *010 + 7) mod 84 = 34
X2 = (17 * 34 + 7) mod 84 = 81
X3 = (17 * 81 + 7) mod 84 = 40
X4 = (17 * 40 + 7) mod 84 = 15

Demikian seterusnya untuk X4, X5, X6...Xn. Sebagai contoh dilakukan empat kali penyebaran dan hasilnya hasilnya adalah “34 81 40 15 ” jika diubah dalam bentuk biner menjadi “00100010 01010001 00101000 00001111.” Untuk mendapatkan hasil modulasi, segmen pesan akan dimodulasi dengan pseudonoise signal menggunakan fungsi XOR (Exlusive OR).

Segmen pesan:

11111111111100001111111111111111
00001111111111111111111100000000
11111111000011110000000011111111
11111111111110000111100000000001111

Pseudonoise signal:

00100010010100010010100000001111

Maka hasil proses modulasi antara segmen pesan dengan pseudonoise signal menggunakan fungsi XOR adalah ;

11111111111100001111111111111111
00100010010100010010100000001111 xox
11011101101000011101011111110000

00001111111111111111111100000000
00100010010100010010100000001111 xox
00101101101011101101011100001111

11111111000011110000000011111111
00100010010100010010100000001111 xox
11011101010111100010100011110000

11111111111100001111000000001111
00100010010100010010100000001111 xox
11011101101000011101100000000000

TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Hasil dari proses modulasi inilah yang [1] akan disisipkan ke bit-bit gambar. Sebagai contoh, misalkan mengambil sepuluh piksel dari gambar dan mengambil tiga puluh bit pertama dari modulasi antara segmen pesan dan pseudonoise signal.

Gambar 4. Citra Gambar RGB Yang Disisipkan Teks

Red = 115 164 126 83 102 110 126 182 195 110
Green = 72 149 68 26 36 46 35 126 175 92
Blue = 88 145 144 154 73 26 61 172 185 103

Kemudian diubah menjadi biner dan disisipkan hasil proses modulasi antara segmen pesan dengan pseudonoise signal menjadi sebagai berikut.

Red	Green	Blue
01110011	01001000	01011000
10100100	10010101	10010001
01111110	01000100	10010000
01010011	00011010	10011010
01111000	00100100	01001001
01101110	00101110	00011010
01111110	00110111	00111101
10110110	01111110	10101100
11000011	10101111	10111001
01101110	01011100	01100111

Langkah tersebut berlanjut sampai modulasi antara segmen pesan dan pseudonoise signal disisipkan semua. Proses terakhir setelah proses penyisipan adalah mengembalikan header.

Gambar 5. Hasil Gambar Penyisipan Teks

Pada proses ekstraksi prosesnya adalah kebalikan dari proses *encode* (penyisipan). Pilih gambar yang akan diekstrak, gunakan kunci yang sama seperti saat encode yaitu “010”. Langkah awal adalah membaca gambar apakah gambar tersebut sudah pernah disisipi gambar atau belum. Apabila belum kemudian suatu fungsi akan mengambil *header* gambar terlebih dahulu, selanjutnya pada *body* gambar dilakukan proses penyaringan agar mendapatkan bit-bit hasil modulasi. Hasil dari proses penyaringan yang dilakukan akan mendapatkan bit-bit sebagai berikut:

11011101101000011101011111110000
00101101101011101101011100001111
11011101010111100010100011110000
11011101101000011101100000000000

Setelah semua bit-bit hasil modulasi diperoleh, kemudian dilakukan proses demodulasi dengan pseudonoise signal dari kata kunci yang sama pada proses modulasi agar memperoleh bit-bit yang berkorelasi. Hasil penyaringan yaitu :

11010011101000111011010111111010
00100011101011001011010100000101
11010011010111000100101011111010
11010011010100111011101000001010

Pseudonoise signal:

00100010010100010010100000001111

Hasil demodulasi:

1111 1111 1111 0000 1111 1111 1111 1111
0000 1111 1111 1111 1111 1111 0000 0000

TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

1111 1111 0000 1111 0000 0000 1111 1111
1111 1111 1111 0000 1111 0000 0000 1111

Proses berikutnya yaitu membagi empat hasil demodulasi, yang berguna untuk menyusutkan hasil demodulasi menjadi isi pesan yang sebenarnya. Proses penyusutan (de-spreading) segmen tersebut menjadi :

11101111011111100110100111101001

Hasil akhir " 11101111 01111100 11010011 11101001 " merupakan segmen pesan yang sama ketika disembunyikan pada proses encode. Hasil tersebut kemudian diubah ke bentuk karakter akan menjadi "239, 124, 211, 197" dalam bentuk simbol *Ascii* yaitu [Ï, |, Ó, Å].

3.3 Proses Dekripsi Algoritma *RC4+*

Proses dekripsi sama dengan proses enkripsi dengan menggunakan kunci yang sama, berikut ini adalah proses dekripsi yang bertujuan untuk mengembalikan pesan tersandi kepesan asli.
Cipherteks (Pesan Tersandi) : [239, 124, 211, 197]
Cipherteks (Pesan Tersandi) : [Ï, |, Ó, Å] (dalam *ASCII)*

Pada tahap awal $i = 0$ dan $j = 0$, Lakukan inisialisasi pada nilai i dan j sama dengan 0. Kemudian $a = S[i]$ dan $b = S[j]$. Selanjutnya lakukan perhitungan nilai i, j, a, b, c dan z yang baru dengan cara :

Untuk $i = 0$
$i = (i + 1) \bmod 256 = (0 + 1) \bmod 256 = 1$
$a = S[i] = S[1] = 70$
$j = (j + a) \bmod 256 = (0 + 70) \bmod 256 = 70$

Selanjutnya tukarkan nilai $S[i]$ dan $S[j]$ dengan cara sebagai berikut :

b	$= S[j]$	$= S[70]$	$= 26$
$S[i]$	$= b$	$= S[1]$	$= 26$
$S[j]$	$= a$	$= S[70]$	$= 70$

Dari perhitungan diatas didapat S[1] = 26 dan S[70] = 70. Berdasarkan proses sebelumnya yang didapatkan nilai i dan j adalah sebagai berikut :
$i : 1$ dan $j : 70$
Selanjutnya hitung nilai c dengan cara :

$$c = (S[(i \ll 5) \ (j \gg 3)) \bmod 256] + S[(j \ll 5) \oplus (i \gg 3)) \bmod 256]) \bmod 256$$
$$c = (S[((1 \ll 5) \oplus (70 \gg 3)) \bmod 256] + S[((70 \ll 5) \oplus (1 \gg 3)) \bmod 256]) \bmod 256$$
$$c = (S[(32 \oplus 0) \bmod 256] + S[(2240 \oplus 0) \bmod 256]) \bmod 256$$
$$c = (S[(32) \bmod 256] + S[(2240) \bmod 256]) \bmod 256$$
$$c = (S[32] + S[192]) \bmod 256$$
$$c = (144 + 110) \bmod 256$$
$$c = 254 \bmod 256$$
$$c = 254$$

Setelah mendapatkan nilai c, selanjutnya hitung nilai z dengan cara sebagai berikut:
$i : 1$ dan $j : 70$
$a = S[i] = S[1] = 70$
$j = (j + a) \bmod 256 = (0 + 70) \bmod 256 = 70$
$S[i] = b = S[1] = 26$
$S[j] = a = S[70] = 70$
Nilai $c : 254$
Mencari nilai z.

$$z = ((S[(a+b) \bmod 256] + S[((c \oplus 170) \bmod 256)]) \oplus S[(j + b) \bmod 256]) \bmod 256$$
$$z = ((S[(70+26) \bmod 256] + S[((254 \oplus 170) \bmod 256)]) \oplus S[(70+26) \bmod 256]) \bmod 256$$
$$z = ((S[(96) \bmod 256] + (254 \oplus 170) \bmod 256)]) \oplus S[(96) \bmod 256]) \bmod 256$$
$$z = ((S[(96) + S[((254 \oplus 170) \bmod 256)]) \oplus S[(96) \bmod 256]) \bmod 256$$
$$z = ((S[96] + S[168]) \oplus S[96]) \bmod 256$$
$$z = ((237 + 96) \oplus 237) \bmod 256$$
$$z = ((333) \oplus 237) \bmod 256$$
$$z = 416 \bmod 256$$
$$z = 160$$

Nilai z ini akan menjadi nilai kunci *[0]* untuk mengekripsi plainteks.
Plainteks yang diamankan : Ï (dalam *ASCII*)

Cipherteks Ï	: 239	11101111	
Kunci	: 160	10100000	⊕
Plainteks	: 79	01001111	
Plainteks	: O	(dalam *ASCII*)	

Nilai $i = 1$ dan $j = 70$. Dilakukan hal yang sama seperti pada pencarian nilai kunci O sampai dengan kunci yang terakhir kunci I.

TIN: Terapan Informatika Nusantara
Vol 1, No 12, Mei 2021, Hal 597-606
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Jadi hasil yang didapat adalah sebagai berikut :

Cipherteks : [239, 124, 211, 197]
Cipherteks : [Ï, |, Ó, Å] (dalam *ASCII)*
Plainteks : [O, T, R, I]
Plainteks : [79, 84, 82, 73] (dalam *ASCII)*

Berdasarkan hasil perhitungan keseluruhan proses enkripsi dan dekripsi pengujian secara manual dengan *iphertext* [Ï, |, Ó, Å] menjadi *Plaintext* [O, T, R, I] dengan algoritma *RC4+* menunjukkan hasil yang sesuai.

3.4 Hasil Pengujian

Hasil pengujian program merupakan hasil akhir dari sebuah program aplikasi yang telah dibangun, mulai dari hasil *input* dan proses adalah sebagai berikut :

1. Pengujian Proases Enkripsi
   Pengujian proses enkripsi merupakan proses yang pertama dilakukan untuk mengamankan pesan rahasia, apakah berhasil atau tidak. Dapat dilihat pada tabel 1 berikut:

Tabel 1. Hasil Pengujian Enkripsi

No	*Plaintext*	Kunci	*Ciphertext*	Penyisipan	Ket
1	OTRI	ZAL	Ï|ÓÅ		Sukses
2	ZALUKHU	YZ	“j|@${€		Gagal

   Berdasarkan pengujian enkripsi tabel 1 sukses karna memiliki kunci yang sama sesuai dengan proses kunci sebelumnya. Sedangkan yang gagal menggunakan kunci yang berbeda.

2. Pengujian Proses Dekripsi
   Pengujian proses dekripsi merupakan proses yang dilakukan untuk mengembalikan dari proses enkripsi yang telah dilakukan, apakah sesuai dengan hasil dari awal. Dapat dilihat pada tabel 2 berikut:

Tabel 2. Hasil Pengujian Dekripsi

No	Ekstraksi	*Ciphertext*	Kunci	*Plaintext*	Keterangan
1		Ï|ÓÅ	ZAL	Berhasil	
2		“j|@${€	YZ	Gagal	

Berdasarkan pengujian dekripsi tabel 2 sukses karna memiliki kunci yang sama sesuai dengan proses kunci sebelumnya pada enkripsi. Sedangkan yang gagal menggunakan kunci yang berbeda.

4. KESIMPULAN

Berdasarkan hasil penelitian dapat di tarik kesimpulan bahwa implementasi pengamanan pesan dengan menggunakan algoritma *RC4+* dan metode *spread spectrum* berhasil meningkatkan keamanan pesan dengan memberikan perlindungan ganda, yaitu yang pertama dengan menyandikan pesan dalam bentuk simbol-simbol yang tidak dimengerti kemudian disisipkan ke dalam sebuah citra. Berdasarkan hasil pengujian yang telah dilakukan menggunakan algoritma *RC4+* dan metode *spread spectrum* adalah semakin panjang pesan asli yang diamankan akan semakin banyak waktu pemrosesan enkripsi dan proses penyisipan pesan. Begitu juga dengan proses dekripsi semakin banyak pesan yang disandikan semakin lama pula waktu pemrosesannya demikian juga dengan proses ekstraksi pesan karna semakin panjang pesan yang akan diekstraksi maka semakin banyak waktu dibutuhkan untuk proses penyaringan pesan dan

demodulasi pesan yang terjadi pada saat proses *encoding*. Hal ini terjadi karena algoritma *RC4+* mengekripsi pesan per karakter dengan kunci yang sama baik dalam proses enkripsi maupun dekripsi.

REFERENCES

[1] Sulindawaty, H. T. Sihotang J. I. Sar, "Implementasi Penyembunyian Pesan Pada Citra Digital Dengan Menggabungkan Algoritma Hill Cipher Dan Metode Least Significant Bit (LSB)," Jurnal Manajemen dan Informatika Pelita Nusantara, vol. 1, no. p-ISSN 2088-3943 e-ISSN 2580-9741, pp. 1-8, Desember 2017.
[2] A. R. Panggabean, I. W. Sinaga, T. Zebua D. Iqbal, "Implementasi Algoritma RC4+ Untuk Mengamankan Pesan Teks Pada Aplikasi Chatting," Seminar Nasional Teknologi Komputer & Sains (SAINTEKS), no. ISBN: 978-602-52720-1-1, pp. 916 - 920, Januari 2019.
[3] Farhan Aulia Rangkuti, "Implementasi Algoritma Vernam Cipher dan Algoritma Rc4+ Cipher Dalam Keamanan File Citra," Skripsi Universitas Sumatera Utara, pp. 1-72, Juli 2018.
[4] E. H. Rachmawanto dan C. A. Sari, "Keamanan File Menggunakan Teknik Kriptografi Shift Cipher," Jurnal Techno.COM, vol. 14, pp. 329-335, November 2015.
[5] Sentot Kromodimoeljo, Teori dan Aplikasi Kriptografi. Jakarta: SPK IT Consulting, 2010.
[6] E. H. Rachmawanto, D. W. Utomo, dan R. R. Sani C. A. Sari, "Penyembunyian Data Untuk Seluruh Ekstensi File Menggunakan Kriptografi Vernam Cipher dan Bit Shiffting," Journal of Applied Intelligent System, vol. 1, pp. 179-190, Oktober 2016.
[7] H. K. Putra dan S. A. Sudiro, "Triple Transposisi dan Spread Spectrum sebagai Metode untuk Pengembangan Algoritme Steganogra," Jurnal Ilmiah KOMPUTASI, vol. 17, no. p-ISSN 1412-9434/e-ISSN 2549-7227, pp. 161-168, Juni 2018.
[8] A. Noercholis dan Y. Nugraha, "Pengamanan Pesan Teks Menggunakan Teknik Steganografi Spread Spectrumberbasis Android," Jurnal Ilmiah dan Teknik, vol. 10, no. p-ISSN: 1978-5232 e-ISSN: 2527-337X, pp. 24-29, Mei 2016.
[9] T. Zebua dan E. Ndruru, "Pengamanan Citra Digital Berdasarkan Modifikasi Algoritma RC4," Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK), vol. 4, no. p-ISSN: 2355-7699 e-ISSN: 2528-6579, pp. 275-282, Desember 2017.
[10] F. A. Setyaningsih, dan M. Dipenogoro M. Rick, "Analisis Kompresi Steganography Pada Citra Digital Dengan Menggunakan Metode Least Significant Bit Berbasis Mobile Android," Jurnal Coding, Rekayasa Sistem Komputer, vol. 06, no. ISSN 2338-493X, pp. 75-86, 2018.