IMPLEMENTASI ALGORITMA UNARY CODING UNTUK MENGKOMPRESI TEKS TERENKRIPSI EL GAMAL

Fitri Angelina Siahaan

Program Studi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia
Jalan Sisingamangaraja No. 338 Medan, Indonesia

Abstrak

Kecepatan dalam pengiriman informasi merupakan hal yang sangat dibutuhkan di dalam perkembangan teknologi sekarang ini. Informasi adalah pesan atau kumpulan pesan atau suatu data yang telah diolah sehingga menghasilkan sesuatu yang dibutuhkan oleh pihak yang membutuhkan. Informasi dapat berbentuk dalam berbagai format data, antara lain teks, video ataupun gambar. Masalah kecepatan pengiriman informasi seringkali sedikit terhambat dikarenakan adanya ukuran file yang besar dari informasi tersebut. Selain itu, faktor keamanan juga sangat penting, karena banyak pihak-pihak yang tidak bertanggungjawab yang akan mencoba untuk mencuri informasi tersebut dan dapat merugikan pihak yang telah dicuri informasinya apabila informasi tersebut disalahgunakan. Proses yang dilakukan di dalam penelitian ini adalah dengan menganalisa proses enkripsi dan dekripsi algoritma El Gamal dan kompresi algoritma Unary coding kemudian merancang aplikasi keamanan dan kompresi menggunakan alat bantu perancangan seperti Unified Modelling Language (UML) dan Microsoft Visual Basic 2008 untuk mempermudah dalam pembuatan aplikasi. Setelah selesai perancangan, kemudian file akan dienkripsi dan di kompresi menggunakan aplikasi yang sudah dirancang tersebut. Kemudian, Hasil dari proses tersebut adalah file teks yang tidak dapat dimengerti dan memiliki ukuran yang lebih kecil dari aslinya.

Kata Kunci: kriptografi, kompresi, elgamal, unarycoding

Abstract

The speed at which information is transmitted is indispensable in current technological developments. Information is a message or a collection of messages or a data that has been processed so as to produce something needed by the party in need. Information can take the form of various data formats, including text, video or images. The problem of the speed of information delivery is often slightly inhibited due to the large file size of the information. In addition, the security factor is also very important, as many irresponsible parties will try to steal the information and may harm the stolen information when it is misused. The process undertaken in this study is to analyze the process of encryption and decryption algorithm El Gamal and compression algorithm Unary coding then designing security and compression applications using design tools such as Unified Modeling Language (UML) and Microsoft Visual Basic 2008 to simplify the making of applications. After completion of the design, then the file will be encrypted and compressed using applications that have been designed. Then, the result of the process is a text file that can not be understood and has a size smaller than the original.

Keywords: cryptography, compression, elgamal, unarycoding

1. PENDAHULUAN

File teks berupa sederetan teks, angka dan sebagainya yang dimuat dalam suatu aplikasi *file* teks yang berupa informasi penting. Umumnya bersifat rahasia sebagai contoh data, no.pin, no. identitas, informasi biodata. Untuk itu agar tidak mudah dibaca atau diambil orang lain untuk kepentingan yang menyulitkan pemiliknya maka perlu diamankan. Salah satu teknik penyimpanan yang dilakukan adalah kriptografi. Kemudian file teks jika diperlukan maka akan dikirim ke orang yang membutuhkan, agar pada proses pengiriman terkesan cepat maka ada baiknya file tersebut diubah ukurannya menjadi lebih kecil karena proses pengiriman file yang lebih kecil akan mempercepat pengiriman untuk itulah diperlukan proses kompresi.

Berdasarkan penelitian Namira Utami Tanjung yang berjudul “Implementasi Algoritma Kriptografi *Kanpsack* dan Algoritma Kompresi Data *Run Length Encoding*”: Perancangan sebuah aplikasi yang dapat menerapkan sistem pengamanan *file* teks dan kemampuan (kompresi) data *file* teks yang telah dienkripsi dapat mengamankan pesan dengan baik, serta dapat melakukan pengecilan ukuran suatu data agar data yang dikirim atau diterima selama komunikasi data berlangsung tidak berukuran besar yang berpotensi tidak terkirim atau diterima sepenuhnya selama proses tersebut.

Salah satu algoritma teknik kompresi yang dapat digunakan adalah algoritma *Unary Coding*, di dalam jurnal Sandeep Kaur yang berjudul “*Entropi Coding and Different Coding Techniques*”: *Unary coding* merupakan *encoding entropi* yang mewakili sejumlah bilangan alami, n , dengan n yang diikuti oleh 0 (jika bilangan alami dipahami sebagai bilangan *integer* non-negatif). *Unary coding* biasanya digunakan di dalam algoritma *Golomb coding*[12].

Salah satu algoritma yang dapat digunakan untuk memenuhi aspek keamanan adalah algoritma *El Gamal* yaitu salah satu algoritma kriptografi kunci publik. Pada algoritma *El Gamal* ada tiga proses yang terjadi yaitu proses pembentukan kunci dengan menggunakan bilangan prima dan bilangan random g dan x , kemudian membagi pesan menjadi blok-blok plainteks untuk dienkripsi dan proses yang terakhir adalah proses dekripsi. Penggunaan algoritma kriptografi *EL Gamal* sangat baik dan sulit dipecahkan karena memerlukan *logaritma*

diskrit dalam proses pemecahannya. Sedangkan penggunaan algoritma *unary coding* merupakan dasar dari algoritma kompresi yang lain seperti *golomb coding*.

Proses yang dilakukan di dalam penelitian ini adalah dengan menganalisa proses enkripsi dan dekripsi algoritma *El Gamal* dan kompresi algoritma *Unary coding* kemudian merancang aplikasi keamanan dan kompresi menggunakan alat bantu perancangan seperti *Unified Modelling Language (UML)* dan *Microsoft Visual Basic 2008* untuk mempermudah dalam pembuatan aplikasi. Setelah selesai perancangan, kemudian *file* akan dienkripsi dan di kompresi menggunakan aplikasi yang sudah dirancang tersebut. Kemudian, Hasil dari proses tersebut adalah *file* teks yang tidak dapat dimengerti dan memiliki ukuran yang lebih kecil dari aslinya.

2. TEORITIS

2.1 File Teks

Secara visual, *file* merupakan dokumen-dokumen yang dikemas dalam sebuah arsip. *File* sering kali identik dengan kertas-kertas kerja yang masing-masing memiliki fungsi dan makna tersendiri. *File* dalam dunia komputerisasi tidaklah hanya sebatas pada dokumen berupa teks saja, tetapi bisa juga gambar, baik sketsa, desain, grafik, dan lain sebagainya. Keragaman jenis *file* tersebut berpengaruh pada beragamnya pula *format file*. *Format file* merupakan suatu cara khusus untuk melakukan pengkodean informasi yang disimpan dalam *file* komputer[3].

Teks adalah bentuk data multimedia yang paling mudah disimpan dan dikendalikan adalah *text* (teks). Teks dapat membentuk kata, surat atau narasi dalam multimedia yang menyajikan bahasa. Kebutuhan teks bergantung pada penggunaan aplikasi multimedia.

2.2 Unary Coding

Unary Coding dikenal juga dengan thermometer code adalah sebuah encoding entropy yang mempergunakan angka dimulai n dengan n berikutnya dimulai oleh 0 (jika angka awal dimengerti sebagai bilangan non negative integer) atau dengan $n-1$ yang diikuti oleh 0 (jika angka yang pertama dimengerti sebagai bilangan positif). Sebagai contoh 5 di dalam bilangan biner adalah 11110 atau 11110. Sering menggunakan n atau $n-1$, 0 diikuti dengan 1. 1 dan 0 bergantian tanpa kehilangan nilai biasanya. Unary coding adalah keduanya sebuah prefix kode dan sinkronisasi kode sendiri.

Unary coding memiliki fungsi encoding dengan mengambil 1 bit dari nilai karakter kemudian menulis bit b untuk keluarannya dan dalam proses decoding fungsi nilai bit yang diambil dikembalikan menjadi nilai 0 atau 1 tergantung nilai awal dari sebuah file yang dikompresi[12].

Langkah-langkah algoritma unary coding adalah sebagai berikut[12]:

1. Mengurutkan frekuensi kemunculan dari tiap karakter dari frekuensi tertinggi hingga terendah.
2. Mendefinisikan bilangan non negatif sebagai $n-1$ bit yang diikuti oleh sebuah bit 0.
3. n yang terakhir diperoleh dengan hanya menggunakan $n-1$ bit tanpa diikuti oleh sebuah bit 0.

2.3 Algoritma El-Gamal

Algoritma *El Gamal* merupakan algoritma dalam kriptografi yang termasuk dalam kategori algoritma asimetris. Keamanan algoritma *El Gamal* terletak pada kesulitan penghitungan logaritma diskrit pada bilangan modulo prima yang besar sehingga upaya untuk menyelesaikan masalah logaritma ini menjadi sangat sukar. Algoritma *El Gamal* mempunyai kunci publik berupa tiga pasang bilangan dan kunci rahasia berupa satu bilangan. Algoritma ini mempunyai kerugian pada ciphertekstanya yang mempunyai panjang dua kali lipat dari plaintekstanya. Akan tetapi, algoritma ini mempunyai kelebihan pada enkripsi. Untuk plainteks yang sama, algoritma ini memberikan cipherteks yang berbeda (dengan kepastian yang dekat) setiap kali plainteks di enkripsi.

Algoritma *El Gamal* terdiri dari tiga proses, yaitu proses pembentukan kunci, proses enkripsi dan proses dekripsi. Algoritma ini merupakan *cipher* blok, yaitu melakukan proses enkripsi pada blok-blok plainteks dan menghasilkan blok-blok cipherteks yang kemudian dilakukan proses dekripsi dan hasilnya digabungkan[8].

a. Proses Pembentukan Kunci

Proses pertama adalah pembentukan kunci yang terdiri dari kunci rahasia dan kunci publik. Pada proses ini dibutuhkan sebuah bilangan prima (p) yang digunakan untuk membentuk grup Z_p^* , elemen primitif α dan sembarang $a \in \{0, 1, \dots, p-2\}$. Kunci publik algoritma *El Gamal* terdiri atas pasangan 3 bilangan, yaitu (p, α, β) di mana:

$$\beta = \alpha^a \text{ mod } p$$

Sedangkan kunci rahasianya adalah bilangan a tersebut. Karena pada algoritma *El Gamal* menggunakan bilangan bulat dalam proses perhitungannya, maka pesan harus dikonversi ke dalam suatu bilangan bulat. Untuk mengubah pesan menjadi bilangan bulat, digunakan kode ASCII. Kode ASCII merupakan representasi *numeric* dari karakter-karakter yang digunakan pada komputer, serta mempunyai nilai minimal 0 dan maksimal 255. Oleh

karena itu, berdasarkan sistem kriptografi *El Gamal* di atas, maka harus digunakan bilangan prima yang lebih besar dari 255[8].

b. Proses Enkripsi

Pada proses ini pesan dienkripsi menggunakan kunci publik (p, a, b) dan sembarang bilangan acak rahasia $k \in \{0, 1, \dots, p-1\}$, $i=1, 2, \dots, n$. Untuk nilai ASCII bilangan 0 digunakan untuk menandai akhir dari suatu teks. Proses enkripsi pada algoritma *El Gamal* dilakukan dengan menghitung:

$$\gamma = a^k \text{ mod } p$$

dan

$$\delta = \beta^k \cdot m \text{ mod } p$$

dengan rahasia $k \in \{0, 1, \dots, p-2\}$ acak. Diperoleh cipherteks (γ, δ)

Bilangan acak k ditentukan oleh pihak pengirim dan harus dirahasiakan, jadi hanya pengirim saja yang mengetahuinya, tetapi nilai k hanya digunakan saat melakukan enkripsi saja dan tidak perlu disimpan[8].

Langkah dalam proses enkripsi algoritma *El Gamal* adalah sebagai berikut[8]:

1. Pesan dipotong-potong ke dalam bentuk blok-blok pesan dengan setiap blok adalah satu karakter pesan.
2. Konversikan masing-masing karakter ke dalam kode ASCII, maka diperoleh plainteks sebanyak n bilangan, yaitu $m_1, m_2, \dots, m_n$.
3. Untuk i dari 1 sampai n kerjakan:
 - a. Pilih sembarang bilangan acak rahasia.
 $k \in \{0, 1, \dots, p-2\}$
 - b. Hitung $\gamma = a^k \text{ mod } p$
 - c. Hitung $\delta = \beta^k \cdot m \text{ mod } p$
4. Diperoleh cipherteks yaitu (γ, δ)

c. Proses Dekripsi

Dekripsi dari cipherteks ke plainteks menggunakan kunci rahasia a yang disimpan kerahasiaannya oleh penerima pesan. Diberikan (p, a, β) sebagai kunci publik dan a sebagai kunci rahasia pada algoritma *El Gamal*. Jika diberikan cipherteks (r, t) , maka: $M = t (r^a)^{-1} \text{ mod } p$

Dengan M adalah plainteks. Di mana nilai:

$$(r^a)^{-1} = r^{-a} = r^{p-1-a} \text{ mod } p.$$

Langkah-langkah proses dekripsi algoritma *El Gamal* adalah sebagai berikut[8]:

1. Ambil sebuah blok cipherteks dari pesan yang telah dienkripsikan pengirim.
 2. Dengan menggunakan a yang dirahasiakan oleh penerima, hitung nilai plainteks dengan menggunakan persamaan $M = t (r^a)^{-1} \text{ mod } p$ dan persamaan $(r^a)^{-1} = r^{-a} = r^{p-1-a} \text{ mod } p$.
- Karena Z_p^* merupakan grup siklik yang mempunyai order $p-1$ dan $a \in \{0, 1, \dots, p-2\}$, maka $(r^a)^{-1} = r^{-a} = r^{p-1-a} \text{ mod } p$.

3. ANALISA DAN PEMBAHASAN

Tahap analisa merupakan tahap yang sangat berpengaruh dan menentukan terhadap tahap selanjutnya. Analisa terhadap sistem merupakan tahap yang sangat penting untuk mengetahui proses yang terjadi di dalam aplikasi yang akan dirancang, dalam hal ini menggunakan algoritma *Unary Coding* untuk mengompress file dan algoritma *El Gamal* untuk menyandikan file.

Pengompressan file berguna untuk mengurangi ruang penyimpanan serta mempercepat proses pengiriman suatu file. Penyandian data berguna untuk meningkatkan keamanan dari suatu file agar tidak dapat dimanipulasi dan dimanfaatkan oleh orang-orang yang tidak bertanggung jawab. Proses yang terjadi di dalam sistem yang akan dibuat memiliki 2 proses utama, yaitu file yang di-input-kan akan disandikan dengan algoritma *El Gamal*, kemudian file tersebut dapat juga dikompress menggunakan algoritma *unary coding*.

3.1 Menerapkan Algoritma El Gamal dalam Penyandian File Teks

Algoritma *El Gamal* memiliki tiga proses utama, yaitu proses pembentukan kunci, proses enkripsi dan proses dekripsi. Algoritma ini merupakan *cipher block* yaitu melakukan proses enkripsi pada blok-blok *plaintext* dan menghasilkan blok-blok *ciphertext* yang kemudian hasilnya akan digabungkan kembali sehingga menjadi *ciphertext* hasil.

Langkah-langkah dalam proses enkripsi algoritma *El Gamal* dapat dilihat dari contoh berikut:

Plaintext : FITRI_ANGELINA_S

Bilangan Prima (p) : 257
 Bilangan random (g) : 19
 Bilangan random x : 13

1. Pembentukan kunci

Setelah mengetahui bilangan prima (p) dan memilih dua buah bilangan acak $g=19$ yang memenuhi syarat $g < p$ dan $x = 11$ yang memenuhi syarat $1 \leq x \leq p - 2$, kemudian hitung y dengan rumus:

$$\begin{aligned} y &= g^x \bmod p \\ &= 19^{13} \bmod 257 \\ &= 42052983462257059 \bmod 257 \\ &= 53 \end{aligned}$$

2. Potong *plaintext* menjadi blok-blok M dan ubah ke dalam kode nilai desimal

M_0	=	F	=	70	M_8	=	G	=	71
M_1	=	I	=	73	M_9	=	E	=	69
M_2	=	T	=	84	M_{10}	=	L	=	76
M_3	=	R	=	82	M_{11}	=	I	=	73
M_4	=	I	=	73	M_{12}	=	N	=	78
M_5	=	_	=	95	M_{13}	=	A	=	65
M_6	=	A	=	65	M_{14}	=	_	=	95
M_7	=	N	=	78	M_{15}	=	S	=	83

3. Kemudian menentukan bilangan acak dan dimasukkan ke dalam blok-blok nilai m secara berurutan, sehingga menjadi:

k_0	=	70	k_8	=	75
k_1	=	45	k_9	=	117
k_2	=	78	k_{10}	=	67
k_3	=	70	k_{11}	=	90
k_4	=	23	k_{12}	=	66
k_5	=	27	k_{13}	=	56
k_6	=	45	k_{14}	=	27
k_7	=	70	k_{15}	=	13

4. Hitung a dengan rumus:

$$a = g^k \bmod p$$

Nilai M_0 :

$$\begin{aligned} a_0 &= 19^{70} \bmod 257 \\ &= 3.2565073798119621181704064348276e+89 \bmod 257 \\ &= 143 \\ a_1 &= 19^{45} \bmod 257 \\ &= 3.498743002442937227729601361123e+57 \bmod 257 \\ &= 206 \\ a_2 &= 19^{78} \bmod 257 \\ &= 5.5307098378518189359850989266487e+99 \bmod 257 \\ &= 79 \\ a_3 &= 19^{70} \bmod 257 \\ &= 3.2565073798119621181704064348276e+89 \bmod 257 \\ &= 143 \\ a_4 &= 19^{23} \bmod 257 \\ &= 257829627945307727248226067259 \bmod 257 \\ &= 148 \\ a_5 &= 19^{27} \bmod 257 \\ &= 3.360061494346044832271606931126e+34 \bmod 257 \\ &= 172 \\ a_6 &= 19^{45} \bmod 257 \\ &= 3.498743002442937227729601361123e+57 \bmod 257 \\ &= 206 \\ a_7 &= 19^{70} \bmod 257 \\ &= 3.2565073798119621181704064348276e+89 \bmod 257 \\ &= 143 \end{aligned}$$

$$\begin{aligned}
 a_8 &= 19^{75} \bmod 257 \\
 &= 8.0634346666450195888396252028702e+95 \bmod 257 \\
 &= 150 \\
 a_9 &= 19^{117} \bmod 257 \\
 &= 4.1131193491150401664715065252947e+149 \bmod 257 \\
 &= 192 \\
 a_{10} &= 19^{67} \bmod 257 \\
 &= 4.7477874031374283688152885768007e+85 \bmod 257 \\
 &= 252 \\
 a_{11} &= 19^{90} \bmod 257 \\
 &= 1.2241202597143419055884639608666e+115 \bmod 257 \\
 &= 31 \\
 a_{12} &= 19^{66} \bmod 257 \\
 &= 2.4988354753354886151659413562109e+84 \bmod 257 \\
 &= 135 \\
 a_{13} &= 19^{56} \bmod 257 \\
 &= 4.0756947817290982884731865054842e+71 \bmod 257 \\
 &= 189 \\
 a_{14} &= 19^{27} \bmod 257 \\
 &= 3.360061494346044832271606931126e+34 \bmod 257 \\
 &= 172 \\
 a_{15} &= 19^{13} \bmod 257 \\
 &= 42052983462257059 \bmod 257 \\
 &= 53
 \end{aligned}$$

5. Hitung b dengan rumus:

$$\begin{aligned}
 b_i &= y^k \cdot m \bmod p \\
 b_0 &= 53^{70} \cdot 70 \bmod 257 \\
 &= 3.5027481475890519034372816258074e+122 \bmod 257 \\
 &= 185 \\
 b_1 &= 53^{45} \cdot 73 \bmod 257 \\
 &= 2.8558605163288824712991841963984e+79 \bmod 257 \\
 &= 130 \\
 b_2 &= 53^{78} \cdot 84 \bmod 257 \\
 &= 2.616960183094351195707540822184e+136 \bmod 257 \\
 &= 169 \\
 b_3 &= 53^{70} \cdot 82 \bmod 257 \\
 &= 4.1032192586043179440265299045173e+122 \bmod 257 \\
 &= 180 \\
 b_4 &= 53^{23} \cdot 73 \bmod 257 \\
 &= 3.3240523969511139884799706744428e+41 \bmod 257 \\
 &= 110 \\
 b_5 &= 53^{27} \cdot 95 \bmod 257 \\
 &= 3.413281324258885166053240563409e+48 \bmod 257 \\
 &= 237 \\
 b_6 &= 53^{45} \cdot 65 \bmod 257 \\
 &= 2.5428895008407857621157119556972e+79 \bmod 257 \\
 &= 158 \\
 b_7 &= 53^{70} \cdot 78 \bmod 257 \\
 &= 3.903062221599229263830113811614e+122 \bmod 257 \\
 &= 96 \\
 b_8 &= 53^{75} \cdot 71 \bmod 257 \\
 &= 1.4857596811277808981914072752856e+131 \bmod 257 \\
 &= 84 \\
 b_9 &= 53^{117} \cdot 69 \bmod 257 \\
 &= 3.7942490497158718524608533084112e+203 \bmod 257 \\
 &= 52 \\
 b_{10} &= 53^{67} \cdot 76 \bmod 257 \\
 &= 2.5544467601420534560287390027757e+117 \bmod 257
 \end{aligned}$$

$$\begin{aligned}
 &= 223 \\
 b_{11} &= 53^{90} \cdot 73 \bmod 257 \\
 &= 1.1172519573597631759566918382107e+157 \bmod 257 \\
 &= 59 \\
 b_{12} &= 53^{66} \cdot 78 \bmod 257 \\
 &= 4.9465453647239366824786902238457e+115 \bmod 257 \\
 &= 105 \\
 b_{13} &= 53^{56} \cdot 65 \bmod 257 \\
 &= 2.3570134147716561041954646068309e+98 \bmod 257 \\
 &= 77 \\
 b_{14} &= 53^{27} \cdot 95 \bmod 257 \\
 &= 3.413281324258885166053240563409e+48 \bmod 257 \\
 &= 237 \\
 b_{15} &= 53^{13} \cdot 83 \bmod 257 \\
 &= 2161047919825338354265759 \bmod 257 \\
 &= 134
 \end{aligned}$$

6. Setelah mendapatkan nilai a dan b , hasil perhitungan tersebut disusun dengan pola sebagai berikut:

$a_0, b_0, a_1, b_1, a_2, b_2, a_3, b_3, a_4, b_4, a_5, b_5, a_6, b_6, a_7, b_7, a_8, b_8, a_9, b_9, a_{10}, b_{10}, a_{11}, b_{11}, a_{12}, b_{12}, a_{13}, b_{13}$.

Sehingga membentuk *ciphertext*:

143, 185, 206, 130, 79, 169, 143, 180, 148, 110, 172, 237, 206, 158, 143, 96, 150, 84, 192, 52, 252, 223, 31, 59, 135, 105, 189, 77, 172, 237, 53, 134.

Jika diubah menjadi karakter adalah:

$[143]'\hat{I}, O\textcircled{[143]}'^n \neg \hat{I}\hat{Z}[143]^-T \quad \hat{A}4\hat{u}\hat{\beta}[31];\ddagger i\frac{1}{2} \quad M-\hat{i}5\ddagger$

3.2 Menerapkan Algoritma *Unary Coding* pada Kompresi *File Teks*

Unary coding direpresentasikan dalam sebuah *string* dari n bit 1 diikuti dengan satu bit 0 yang mengakhiri didefinisikan sebagai $n-1$ bit 1 diikuti satu bit 0. Atau sebaliknya sebagai alternative dapat juga secara ekuivalen dimulai dari n bit 0 diikuti dengan bit 1 yang mengakhiri didefinisikan sebagai $n-1$ bit 0 diikuti dengan satu bit 1. Pada algoritma *unary coding* tidak terdapat pembagian frekuensi simbol-simbol yang ada pada sebuah *string*.

Contoh:

Sebuah *file* teks yang sudah dienkripsi menggunakan algoritma *el gamal* yang memiliki isi sebagai berikut:

$[143]'\hat{I}, O\textcircled{[143]}'^n \neg \hat{I}\hat{Z}[143]^-T \hat{A}4\hat{u}\hat{\beta}[31];\ddagger i\frac{1}{2}M-\hat{i}5\ddagger$, Berikut ini adalah

daftar frekuensi kemunculan simbol pada pesan yang telah diurutkan secara menurun berdasarkan frekuensi kemunculan.

Tabel 1. Tabel Frekuensi *Unary Coding*

Simbol	Frekuensi	n
$[143]$	3	1
$\hat{I}$	1	2
$\hat{I}$	2	3
$,$	1	4
O	1	5
$\textcircled{[143]}$	1	6
$\neg$	2	7
$\hat{Z}$	1	8
N	1	9
$\neg$	2	10
$\hat{I}$	3	11
$\hat{Z}$	1	12
$-$	1	13
T	1	14
$\hat{A}$	1	15
4	1	16
$\hat{U}$	1	17
β	1	18
$[31]$	1	19
$;$	1	20
$\ddagger$	1	21

$\frac{1}{2}$	1	22
M	1	23
5	1	24
†	1	25

N merupakan urutan dari frekuensi tertinggi hingga frekuensi terendah. Apabila terdapat beberapa simbol dengan frekuensi yang sama, maka simbol-simbol tersebut diurutkan berdasarkan kemunculannya pada *string* yang akan dikompresi. Pada *unary coding integer non-negatif* didefinisikan sebagai $n-1$ bit 1 yang diikuti oleh sebuah bit 0. n yang terakhir diperoleh dengan melakukan sebuah modifikasi, yaitu dengan hanya menggunakan $n-1$ bit. 1 tanpa diikuti oleh sebuah bit 0. Pada contoh di atas dapat diperoleh kode *unary coding* sebagai berikut:

Tabel 2. Tabel Perhitungan *Unary Coding*

Simbol	Frekuensi	n	Kode	Total bit
[143]	3	1	0	3
!	1	2	10	2
Î	2	3	110	6
,	1	4	1110	4
O	1	5	11110	5
©	1	6	111110	6
'	2	7	1111110	14
”	1	8	11111110	8
n	1	9	111111110	9
¬	2	10	1111111110	18
í	3	11	11111111110	3
ž	1	12	111111111110	2
—	1	13	1111111111110	3
T	1	14	11111111111110	4
À	1	15	111111111111110	5
4	1	16	1111111111111110	6
ü	1	17	11111111111111110	7
ß	1	18	111111111111111110	8
[31]	1	19	1111111111111111110	9
;	1	20	11111111111111111110	9
‡	1	21	111111111111111111110	1
$\frac{1}{2}$	1	22	1111111111111111111110	2
M	1	23	11111111111111111111110	3
5	1	24	111111111111111111111110	4
†	1	25	1111111111111111111111110	5
Total	32			152
Frekuensi				

Tabel di atas menunjukkan kode dari setiap simbol yang ada pada pesan. Berdasarkan tabel tersebut, total *byte* yang dibutuhkan untuk mewakili pesan di atas adalah 19 *byte*. Sedangkan ukuran *file* awal sebelum dikompresi adalah 32 *byte*, sehingga rasio kompresinya adalah

$$\text{Rasio Kompresi} = \frac{19}{32} \times 100\% = 59,38\%$$

Dari perhitungan di atas, dapat disimpulkan bahwa dengan menggunakan algoritma *unary coding* karakter di atas dapat di kompresi sebanyak 59,38%

3.3 Proses Dekompresi *Unary Coding*

Proses dekompresi *unary coding* diawali dengan meng-input *file unary coding* yang akan dikompresi, kemudian kode-kode untuk setiap simbol dari *file unary coding* akan dibaca. Kode-kode tersebut akan dicocokkan *bit per bit* dengan kode pada simbol-simbol dari *file* asli. Berdasarkan tabel 3.2 nilai yang sudah menjadi kode akan dibaca oleh sistem, kemudian menggantinya dengan nilai awal nya sebagai berikut:

Kode 10 akan diganti dengan nilai simbol ‘!’,

Kode 110 akan digantikan dengan nilai simbol ‘Î’,

Kode 1110 akan digantikan dengan nilai simbol ‘,’,

Begitu seterusnya hingga menghasilkan *file* asli yaitu *file* yang sudah terenkripsi sebagai berikut:

[143]1,00[143]''' n-11[143]-T A4uB[31];i1/2 M-i5†

3.4 Proses Dekripsi Algoritma El Gamal

Proses dekripsi merupakan proses pengembalian *ciphertext* menjadi *plaintext* sebagai berikut:

Ciphertext : [143]1,00[143]''' n-11[143]-T A4uB[31];i1/2 M-i5†

Langkah-langkah dalam proses dekripsi *ElGamal* adalah sebagai berikut:

1. Konversi *ciphertext* menjadi bilangan decimal

Ciphertext : 143, 185, 206, 130, 70, 169, 143, 180, 148, 110, 172, 237, 206,
 158, 143, 96, 150, 84, 192, 52, 252, 223, 31, 59, 135, 105, 189,
 77, 172, 237, 53, 134.

2. Hitung nilai m_i sebagai nilai *plaintext* dengan rumus sebagai berikut:

$$\begin{aligned}
 m_i &= b_i \cdot a_i^{(p-1-x)} \bmod p \\
 m_0 &= 185.143^{(257-1-13)} \bmod 257 \\
 &= 1.0323544458444758894945263352696e+526 \bmod 257 \\
 &= 70 \\
 m_1 &= 130.206^{(257-1-13)} \bmod 257 \\
 &= 2.4192341367953433493987687510653e+564 \bmod 257 \\
 &= 73 \\
 m_2 &= 169.79^{(257-1-13)} \bmod 257 \\
 &= 2.2452767963143157688866301497689e+463 \bmod 257 \\
 &= 84 \\
 m_3 &= 180.143^{(257-1-13)} \bmod 257 \\
 &= 1.0044529743351657303189985964786e+526 \bmod 257 \\
 &= 82 \\
 m_4 &= 110.148^{(257-1-13)} \bmod 257 \\
 &= 2.6000968559504188940275199618874e+529 \bmod 257 \\
 &= 73 \\
 m_5 &= 237.172^{(257-1-13)} \bmod 257 \\
 &= 4.056588401007675867029455010912e+545 \bmod 257 \\
 &= 95 \\
 m_6 &= 158.206^{(257-1-13)} \bmod 257 \\
 &= 2.9402999508743403785000420205255e+564 \bmod 257 \\
 &= 65 \\
 m_7 &= 96.143^{(257-1-13)} \bmod 257 \\
 &= 5.3570825297875505617013258478857e+525 \bmod 257 \\
 &= 78 \\
 m_8 &= 84.150^{(257-1-13)} \bmod 257 \\
 &= 5.1814968276126386210563473403667e+530 \bmod 257 \\
 &= 71 \\
 m_9 &= 52.192^{(257-1-13)} \bmod 257 \\
 &= 3.615779337125797723369519119831e+556 \bmod 257 \\
 &= 69 \\
 m_{10} &= 223.252^{(257-1-13)} \bmod 257 \\
 &= 7.7381344794135136795563006176598e+585 \bmod 257 \\
 &= 76 \\
 m_{11} &= 59.31^{(257-1-13)} \bmod 257 \\
 &= 1.4850586760470012653101422079857e+364 \bmod 257 \\
 &= 73 \\
 m_{12} &= 105.135^{(257-1-13)} \bmod 257 \\
 &= 4.9237392243021522949588905401732e+519 \bmod 257 \\
 &= 78 \\
 m_{13} &= 77.189^{(257-1-13)} \bmod 257 \\
 &= 1.166028429435743699179944389629e+555 \bmod 257 \\
 &= 65
 \end{aligned}$$

$$\begin{aligned} m_{14} &= 237.172^{(257-1-13)} \bmod 257 \\ &= 4.056588401007675867029455010912e+545 \bmod 257 \\ &= 95 \\ m_{15} &= 134.53^{(257-1-13)} \bmod 257 \\ &= 1.3370298749349190947887169053936e+421 \bmod 257 \\ &= 83 \end{aligned}$$

3. Kemudian nilai tersebut digabung kembali sehingga menghasilkan nilai sebagai berikut:
70, 73, 84, 82, 73, 95, 65, 78, 71, 69, 76, 73, 78, 65, 95, 83
4. Konversi nilai tersebut menjadi karakter sehingga menghasilkan kembali *plaintext* sebagai berikut:
FITRI_ANGELINA_S

4. KESIMPULAN

Berdasarkan pembahasan dan evaluasi dari bab-bab sebelumnya, maka dapat diambil kesimpulan-kesimpulan. Adapun kesimpulan-kesimpulan tersebut adalah sebagai berikut:

1. Penerapan algoritma *El Gamal* dalam penyandian *file* teks dilakukan dengan menggunakan kunci publik yang dibangkitkan terlebih dahulu kemudian membagi *file* teks menjadi blok-blok *cipher* untuk dienkripsi dan menggabungkan kembali blok tersebut menjadi *ciphertext*, begitu juga sebaliknya.
2. *Ciphertext* yang didapat dari proses enkripsi *ElGamal* dikompresi menggunakan algoritma *Unary Coding* dengan mengganti nilai karakter dari *file* yang akan dikompres ke dalam tabel *unary* yang berbasis satu, sehingga menghasilkan tabel kode yang akan digunakan untuk menggantikan karakter awal *file*.
3. Perancangan aplikasi dapat dilakukan dengan menggunakan alat bantu *Unified Modeling Language* dan *tools* pemrograman *Microsoft Visual Studio 2008* yang mampu melakukan proses enkripsi dan kompresi, kemudian dapat melakukan proses dekompresi dan dekripsi sehingga *file* kembali ke ukuran dan nilai aslinya.

REFERENCES

- [1] A.Nugroho, *Rekayasa Perangkat Lunak Berorientasi Objek dengan Metode USDP*, Yogyakarta: ANDI. 2010
- [2] E.Setyaningsih, *Kriptografi & Implementasinya Menggunakan Matlab*, Yogyakarta: Penerbit Andi.2015
- [3] J.Enterprise, *Rahasia Manajemen File*, Jakarta: PT.Elex Media Komputindo.2010
- [4] L.Sitorus, M.Kom, *Algoritma dan Pemrograman*, Yogyakarta: Penerbit Andi. 2015
- [5] Wardana, *Membuat Aplikasi Berbasis Pendekatan Sistem dengan Visual Basic .Net 2008*, Jakarta : PT. Elex Media Komputindo.2008
- [6] W.Gata, G.Gata, *Sukses Membangun Aplikasi Penjualan Dengan Java*, Jakarta: PT.Elex Media Komputindo.2013
- [7] A.Aneta, "Implementasi Kebijakan Program Penanggulangan Kemiskinan Perkotaan (P2KP) di Kota Gorontalo", *Jurnal Administrasi Publik*, vol.I, pp. 54-65. 2010
- [8] D. T. Massandy, "Algoritma ElGamal dalam Pengamanan Pesan Rahasia", *Jurnal Institut Teknologi Bandung*, pp. 1-5.2010
- [9] H.Christian, "Studi tentang Pelaksanaan Rencana Kerja Pembangunan Desa Tahun 2013 di Desa Loa Janan Ulu Kecamatan Loa Janan Kabupaten Kutai Kartanegara", *eJournal Pemerintahan Integratif*, vol.III, pp. 190-210. 2015
- [10] Hidayat, W.Zarman, et.al, "Implementasi Algoritma Kompresi LZW Pada Database Server", *Jurnal Komputa*, vol.II, pp. 7-14. 2013
- [11] P.Sulistyorini, "Pemodelan Visual dengan Menggunakan UML dan Rational Rose", *Jurnal Teknologi Informasi DINAMIK*, Vol.XIV, pp.23-29, 2009
- [12] S.Kaur, "Entropy Coding and Different Coding Technique", *Journal of Network Communication and Emerging Technologies*, vol.VI, pp. 4-7, 2016
- [13] Z.Amin, "Desain dan Implementasi Tunneling IPSEC Berbasis UNIX dengan ESP (Encapsulating Security Payload)", *STMIK PalComTech*, vol.II, pp109-119. 2012