ISSN : 2598-294X
E-ISSN : 2598-2958

ANALISIS KOMPLEKSITAS WAKTU ALGORITMA KRIPTOGRAFI ELGAMAL DAN DATA ENCRYPTION STANDARD

Herman Kabetta
Program Studi Teknik Informatika STMIK Widya Utama
hermanka.beta@gmail.com

Abstract— *ElGamal as an asymmetric key cryptography system and Data Encryption Standard (DES) as a symmetric key cryptography system, both of algorithms will be compared using the time complexity analysis and computer simulation. The result of time analysis shows a different complexity for both algorithms, there is quadratic complexity for ElGamal Algorithm and Linear Complexity for DES algorithm. Input that is used by ElGamal algorithm is the private key, while for DES algorithm is the plaintext's size. Based on result of simulation using a computer program, it shows a significant timing differences, ElGamal's time execution is longer than DES. This is caused by differences of arithmetic operations that is used by each algorithms.*

Keywords—complexity analysis, cryptography, elgamal, data encryption standard.

1. PENDAHULUAN

Kriptografi adalah seni dan ilmu untuk merancang atau menghasilkan pesan rahasia dari pesan asli untuk berkomunikasi secara aman antara pengirim dan penerima [1]. Terdapat dua jenis algoritma kriptografi dilihat dari kunci yang digunakan untuk enkripsi dan dekripsi, yakni, algoritma kunci privat atau algoritma simetris dan algoritma kunci publik atau algoritma asimetris [2]. Algoritma kunci privat merupakan algoritma konvensional karena menggunakan kunci enkripsi dan dekripsi yang sama, sedangkan algoritma kunci publik menggunakan kunci enkripsi dan dekripsi yang berbeda. Jika keduanya dibandingkan dari segi keamanan tentu algoritma kunci publik lebih aman daripada algoritma kunci privat [3], hal ini dikarenakan kunci untuk dekripsi tidak perlu ditransmisikan melalui media apapun sehingga tetap aman tersimpan. Namun demikian, proses enkripsi dan dekripsi pada algoritma kunci privat dianggap lebih cepat daripada algoritma kunci publik, sehingga algoritma kunci publik jarang sekali diterapkan pada kasus data besar [4]. Oleh karena itu, pada penelitian ini akan dibandingkan performa kedua jenis algoritma menggunakan analisis kompleksitas serta simulasi komputer.

Algoritma yang akan dibandingkan adalah algoritma *Data Encryption Standard* (DES) dan Algoritma ElGamal. *Data Encryption Standard* (DES) merupakan salah satu jenis algoritma kunci privat yang pertama direkomendasikan oleh NIST (*National Institute of Standards and Technology)* sebagai standar pengolahan informasi Federal AS pada tahun 1997 [5]. Sedangkan Algoritma ElGamal merupakan salah satu algoritma kriptografi kunci publik yang sudah familiar digunakan, khususnya pada program pertukaran data yang sangat terkenal yaitu PGP (*Pretty Good Privacy*), dan penerapannya pada tanda-tangan digital (*Digital Signature)* [6]. Kekuatan dari algoritma ElGamal terletak pada sulitnya menghitung logaritma diskrit [7].

2. METODOLOGI

Pada penelitian ini akan digunakan metode analisis kompleksitas waktu algoritma berdasarkan waktu (kompleksitas waktu) serta simulasi komputer untuk membandingkan kedua algoritma, *Data Encryption Standard* (DES) dan ElGamal. Kompleksitas waktu, $T(n)$, diukur dari jumlah tahapan komputasi yang dibutuhkan untuk menjalankan algoritma sebagai fungsi dari ukuran masukan n [8].

Data yang akan digunakan untuk analisis kompleksitas adalah *pseudocode* kedua algoritma. Kompleksitas waktu dihitung untuk setiap proses enkripsi dan dekripsi.

Pada tahap simulasi, alat uji yang digunakan untuk eksperimen adalah sebuah program komputer yang telah dibangun menggunakan Lazarus. Program komputer merupakan implementasi dari pseudocode yang telah dibuat sebelumnya, dirancang untuk dapat meng-enkripsi dan men-dekripsi teks asli yang kemudian dapat mencatat waktu proses untuk kedua algoritma. Eksperimen dilakukan dengan menguji beberapa variasi plainteks dengan panjang teks tertentu. Data hasil eksperimen simulasi komputer kemudian dianalisis menggunakan metode statistika deskriptif.

3. PEMBAHASAN

3.1 Analisis Komplesitas Waktu Algoritma ElGamal

Algoritma ElGamal merupakan algoritma kriptografi asimetris yang menggunakan dua jenis kunci, yaitu kunci publik dan kunci privat. Algoritma ElGamal mempunyai kunci publik berupa tiga pasang bilangan dan kunci rahasia berupa satu bilangan.

ISSN : 2598-294X
E-ISSN : 2598-2958

Properti yang digunakan pada algoritma ElGamal antara lain [9]:

1. Bilangan prima, p (tidak rahasia)
2. Bilangan acak, g ($g < p$) (tidak rahasia)
3. Bilangan acak, x ($x < p$)(rahasia, kunci privat)
4. $y = g^x \bmod p$ (tidak rahasia, kunci publik)
5. m (plainteks) (rahasia)
6. a dan b (cipherteks) (tidak rahasia)

Proses pertama dalam algoritma Elgamal adalah pembangkitan kunci. Pada proses ini dibutuhkan sebuah bilangan prima p, elemen g dan sembarang $x \in \{0,1,..., p-2\}$.

Langkah-langkah :

1. Pilih sembarang bilangan prima $p>255$ (p dapat dipublikasikan)
2. Pilih dua buah bilangan acak, g dan x, dengan syarat $g < p$ dan $0 \le x \le p-2$
3. Hitung $y = g^x \bmod p$.

Kunci publik algoritma ElGamal berupa pasangan 3 bilangan, yaitu (y,g,p). Sedangkan kunci privatnya adalah bilangan x tersebut.

Setelah proses pembangkitan kunci selesai kemudian pesan dienkripsi menggunakan kunci public (y,g,p) dan sembarang bilangan acak rahasia $k \in \{0,1,..., p-2\}$. Misalkan m adalah pesan yang akan dikirim.

Langkah-langkah Enkripsi Algoritma ElGamal :

1. Susun plainteks menjadi blok-blok m_1, m_2, …, m_n, 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, …, m_n,
3. Untuk i dari 1 sampai n kerjakan :
   3.1. Pilih sembarang bilangan acak rahasia k $i \in \{0,1,..., p-2\}$.
   3.2. Hitung $a_i = g^{ki} \bmod p$
   3.3. Hitung $b_i = y^{ki} m_i \bmod p$
4. Diperoleh cipherteks yaitu (a_i,b_i) , i =1, 2,..., n. jadi ukuran Cipherteks dua kali ukuran plainteksnya.

Dari algoritma diatas, kemudian dibuat *pseudocode* untuk memudahkan proses analisis. Dalam *pseudocode* algoritma ElGamal ini terdapat tiga buah perulangan, perulangan yang pertama adalah operasi pembangkitan kunci, dan dua yang lainnya adalah proses enkripsi. Pada *pseudocode* ini plainteks dan cipherteks diterjemahkan ke dalam bentuk bilangan integer.

Berdasarkan *pseudocode* di gambar 1, kemudian dihitung banyaknya langkah-langkah pada operasi-operasi yang khas.

Banyak langkah = C1 + x + C2 + n. (C3 + k + C4)
= x + nk + C

Dari banyak langkah yang telah dihitung, maka diperoleh kompleksitas waktu untuk algoritma enkripsi ElGamal adalah,

$$T(n) = x + n\text{k} + \text{C}$$

```
Procedure ElGamalEnkripsi(
input g,p,m[n],x : integer,
output a,b :integer)

{  Prosedur Enkripsi pada  Algoritma ElGamal
   Masukan : g,p,x, plainteks m[1]..m[n]
   Keluaran: cipherteks a[1]b[1]..a[n]b[n]
}

Deklarasi
   k:integer
Algoritma
   temp_y←1                          } C1
   For i←1 to x do                   }
      temp_y←temp_y*g                } x
   Endfor                            }
   y←(temp_y) mod p                  } C2
   for i←1 to n do                                    }
      k←random(p-2)                  }               }
      temp_a←1                       } C3            }
      temp_b←1                       }               }
      for j←1 to k do                }               }
            temp_a←temp_a*g          } k             } n
            temp_b←temp_b*y          }               }
      endfor                         }               }
      a[i]←(temp_a) mod p            }               }
      b[i]←(temp_b*m[i])mod p        } C4            }
endfor
```

Gambar 1 Pseudocode enkripsi Elgamal

Sehingga kompleksitas waktu untuk kasus terburuk algoritma enkripsi ElGamal dapat dinyatakan sebagai berikut,

$$T_{max}(n) = x + n\text{k} + \text{C}$$

Pada penggunaan yang sebenarnya, pemakaian x menggunakan seratus digit bilangan sebagai kunci privatnya, yang mengakibatkan lamanya proses enkripsi banyak tersita pada proses pembangkitan kunci, atau dapat dikatakan bahwa lama proses enkripsi sebanding dengan nilai x. Sedangkan n yang merupakan proses perulangan untuk setiap karakter plainteks dapat akan berpengaruh jika jumlah teks yang dienkripsi sangat panjang. Jika diasumsikan pemilihan nilai random k menggunakan bilangan yang sangat besar, bahkan mendekati nilai x, dan n menggunakan plainteks yang sangat panjang maka nilai k dan n akan sebanding dengan x. Oleh karena itu, nilai k dan n tidak bisa diabaikan karena turut berpengaruh terhadap lama proses enkripsi. Sehingga kompleksitas waktu untuk kasus terburuk menjadi,

$$T_{max}(n) = x^2 + x + \text{C}$$

ISSN : 2598-294X
E-ISSN : 2598-2958

Dari kompleksitas waktu tersebut, dapat diketahui kompleksitas waktu asimtotik dari proses enkripsi algoritma ElGamal adalah

$$T(n) = O(x^2)$$

Proses dekripsi pada algoritma ElGamal menggunakan kunci publik p dan kunci rahasia x. Dapat ditunjukkan bahwa plainteks m dapat diperoleh dari cipherteks menggunakan kunci rahasia x.
Langkah-langkah :

1. Cipherteks (a_i,b_i) , i =1, 2,..., n, kunci publik p dan kunci rahasia x.
2. Untuk i dari 1 sampai n kerjakan :
   2.1. Hitung $a_i^{p-1-x} \bmod p$
   2.2. Hitung $m_i = b_i/a_i^x \bmod p = b_i(a_i^x)^{-1} \bmod p$
3. Diperoleh plainteks m_1, m_2, …, m_n.
4. Konversikan masing-masing bilangan m_1, m_2, …, m_n ke dalam karakter sesuai dengan kode ASCII-nya, kemudian hasilnya digabungkan kembali.

Berbeda dari proses Enkripsinya, pada proses dekripsi, algoritma hanya menggunakan dua buah perulangan dikarenakan pada proses ini tidak ada langkah pembangkitan kunci.

```
Procedure ElGamalDekripsi(
input a,b,p,x : integer,
output m :integer)

Deklarasi
   r:integer

Algoritma
   r←p-1-x                          } C1
   For i←1 to n do                  } C2
      Temp←1
      For j←1 to r do               } r
            Temp←temp*a[i]
      endfor
      a[i]←a[i] mod p               } C3
      m[i]←b[i]*a[i]) mod p
   endfor
```

n

Gambar 2 Pseudocode dekripsi Elgamal

Berdasarkan *pseudocode* pada gambar 2, kemudian dihitung banyaknya langkah proses dekripsi.
Banyak langkah = C1 + n . (C2 + r + C3)
= nr + C

Dari banyak langkah yang telah dihitung, maka diperoleh kompleksitas waktu untuk algoritma dekripsi ElGamal adalah :

$$T(n) = n\text{r} + \text{C}$$

Pada penggunaan yang sebenarnya, pemilihan kunci privat x menggunakan bilangan yang sangat besar, sehingga mempengaruhi nilai r. Hal ini menjadikan variabel yang berpengaruh terhadap kompleksitas waktu algoritma dekripsi ElGamal adalah x, dan proses dekripsi menggunakan cipherteks yang sangat panjang hingga mendekati x, sehingga kompleksitas waktu untuk algoritma dekripsi ElGamal adalah:

$$T(n) = x^2 + \text{C}$$

Persamaan diatas adalah kompleksitas waktu untuk kasus terbaik dan terburuk, sehingga kompleksitas waktu asimtotik algoritma dekripsi ElGamal dapat dinyatakan sebagai berikut.

$$T(n) = O(x^2)$$

Dengan demikian, kompleksitas waktu dari algoritma enkripsi dan dekripsi ElGamal adalah kuadratik.

3.2 Analisis Komplesitas Waktu Algoritma *Data Encryption Standard* (DES)

Algoritma DES memadukan teknik blocking, permutasi, ekspansi, dan substitusi, serta operasi XOR yang semuanya dilakukan dalam 16 kali perulangan [10]. Sebagai sistem sandi modern yang berbasiskan peralatan elektronik, DES beroperasi dalam bentuk bit yang berupa angka binari 0 dan 1, yang berkelompok dengan masing-masing kelompok terdiri dari 8 bit membentuk bilangan heksadesimal atau bilangan berbasis 16. DES beroperasi pada ukuran blok 64 bit, mengenkripsikan 64 bit plainteks menjadi 64 bit cipherteks dengan menggunakan 56 bit kunci internal (*internal key*). Kunci internal dibangkitkan dari kunci eksternal (*external key*) yang panjangnya 64 bit.

Ada tiga langkah umum pada proses enkripsi DES, yaitu Inisial Permutasi (proses pengacakan urutan bit-bit pada plainteks), 16 putaran *enciphering*, dan invers Inisial Permutasi. Skema global dari algoritma DES adalah sebagai berikut:

a. Blok plainteks dipermutasi dengan permutasi awal (IP, *Initial Permutation*).
b. Hasil permutasi awal kemudian di-*enciphering* sebanyak 16 kali (16 putaran). Setiap putaran menggunakan kunci internal yang berbeda dengan perhitungan L_iR_i dengan $1 \le i \le 16$.
c. Hasil *enciphering* kemudian dipermutasi dengan matriks permutasi balikan (*invers initial permutation* atau IP^{-1}) menjadi blok cipherteks.

Pada proses dekripsi, kunci yang digunakan sama persis dengan kunci yang digunakan dalam proses enkripsi, hanya saja penggunaan kunci internalnya yang berbeda urutannya. Dalam proses dekripsi, urutan kunci internal yang digunakan merupakan kebalikan dari urutan kunci internal yang digunakan pada proses enkripsi.

Inti dari Algoritma DES adalah pengacakan bit, pengacakan dilakukan menggunakan kaidah susunan matriks yang telah ditentukan. Pada *pseudocode* DES, proses pembangkitan kunci internal dilakukan sebelum

ISSN : 2598-294X
E-ISSN : 2598-2958

proses enkripsi maupun dekripsi, oleh karena itu terdapat dua buah perulangan, perulangan yang pertama adalah proses pembangkitan kunci internal, sedangkan perulangan yang kedua merupakan proses enkripsi atau dekripsi dari algoritma DES. Input maupun output proses dikondisikan berupa bilangan biner.

```
Procedure DESenkripsi(
input kunci, plainteks:Byte,
output cipherteks:Byte)

Deklarasi
   Ip, ipInv, kunci, plainteks of Byte
   cipherteks:array[0..63] of Byte
   Exp, pc2,k,:array[0..47] of Byte
   pBox:array[0..31] of Byte
   sBox:array[0..7] of array[0..63] of Byte
   pc1:array[0..55] of Byte
   n: integer

Algoritma
   n←16                                  } C1
   c[0]d[0]←pc1(kunci)                   }
   for i←1 to n do                       }
      c[i]←LeftShift[i](c[i-1])          } n
      d[i]←LeftShift[i](d[i-1])          }
      k[i]←pc2(c[i]d[i])                 }
   endfor
   L[0]R[0]←ip(plainteks)                } C2
   for i←1 to n do                       }
      A←exp(R[i-1])XOR k[i]              }
      B←sBox(A)                          } n
      R[i]←L[i-1] XOR pBox(B)            }
      L[i]←R[i-1]                        }
   Endfor                                }
Cipherteks←ipInv(R[16]L[16])             } C3
```

Gambar 3 Pseudocode Enkripsi DES

Berdasarkan *pseudocode* pada gambar 3, kemudian dihitung banyaknya langkah proses enkripsi algoritma DES.

Banyak langkah = C1 + n + C2 + n + C3
= 2n + C1 + C2 + C3
= 2n + C

Pada proses enkripsi algoritma DES yang dilakukan hanyalah pengacakan bit dan penggunaan operasi logika XOR. Hal ini mengakibatkan lama proses enkripsi hanya dipengaruhi oleh nilai n sebagai jumlah plainteks masukan, atau dapat diasumsikan bahwa kompleksitas kasus terbaik dan terburuknya adalah sama, yaitu

$$T(n) = 2n + C$$

Sehingga kompleksitas waktu asimtotik algoritma enkripsi DES dapat dinyatakan sebagai berikut.

$$T(n) = 2n + C = O(n)$$

Proses dekripsi pada algoritma DES sama dengan proses enkripsi-nya, hanya berbeda pada penggunaan kunci yang dibalik, pergeseran bit kunci eksternal yang dilakukan secara *right shift* dan masukan berupa cipherteks.

Dari *pseudocode* di gambar 4, kemudian dihitung banyaknya langkah proses dekripsi algoritma DES.

Banyak langkah = C1 + n + C2 + n + C3
= 2n + C1 + C2 + C3
= 2n + C

```
Procedure DESdekripsi(
input kunci, cipherteks:Byte,
output plainteks:Byte)

Deklarasi
   Ip, ipInv, kunci, plainteks of Byte
   cipherteks:array[0..63] of Byte
   Exp, pc2,k,:array[0..47] of Byte
   pBox:array[0..31] of Byte
   sBox:array[0..7] of array[0..63] of Byte
   pc1:array[0..55] of Byte
   n: integer

Algoritma
   n←16                                  } C
   c[0]d[0]←pc1(kunci)                   }
   for i←1 to n do                       }
      c[i]←RightShift[i](c[i-1])         } n
      d[i]←RightShift[i](d[i-1])         }
      k[i]←pc2(c[i]d[i])                 }
   endfor
   L[0]R[0]←ip(cipherteks)               } C2
   for i←1 to n do                       }
      A←(R[i-1])XOR k[n+1-i]             }
      B←Box(A)                           } n
      R[i]←L[i-1] XOR pBox(B)            }
      L[i]←R[i-1]                        }
   Endfor                                }
plainteks←ipInv(R[16]L[16])              } C3
```

Gambar 4 Pseudocode Dekripsi DES

Sama halnya dengan proses enkripsi, proses dekripsi algoritma DES hanya dipengaruhi oleh nilai n sebagai jumlah cipherteks masukan, atau dapat diasumsikan bahwa kompleksitas kasus terbaik dan terburuknya adalah sama, yaitu

$$T(n) = 2n + C$$

Sehingga kompleksitas waktu asimtotik algoritma dekripsi DES dapat dinyatakan sebagai berikut.

$$T(n) = 2n + C = O(n)$$

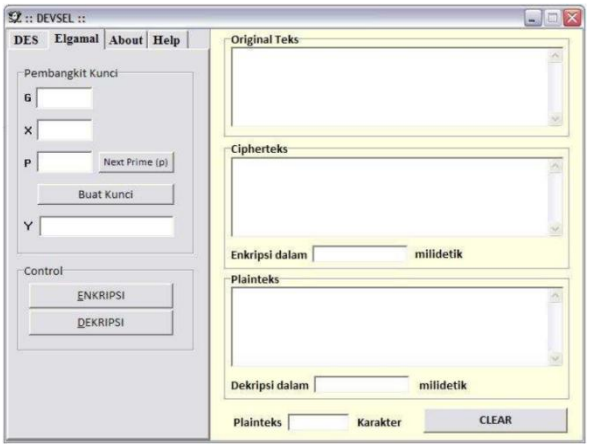
Dengan demikian, kompleksitas waktu dari algoritma enkripsi maupun dekripsi DES dapat dikatakan linier.

3.3 Simulasi Komputer

Simulasi komputer dilakukan untuk membandingkan performa kedua algoritma, baik dalam proses enkripsi maupun dekripsi. Pencatatan waktu akan dilakukan oleh program ketika tombol enkripsi/dekripsi ditekan. Dalam uji coba ini kedua algoritma mendapatkan masukan yang sama dalam proses enkripsi/dekripsi teks sampel, yaitu pengujian dilakukan sebanyak tiga kali untuk masing-masing

ISSN : 2598-294X
E-ISSN : 2598-2958

algoritma, dengan jumlah karakter pada teks sampel berturut-turut sebanyak 1000, 2000, dan 3000 karakter.

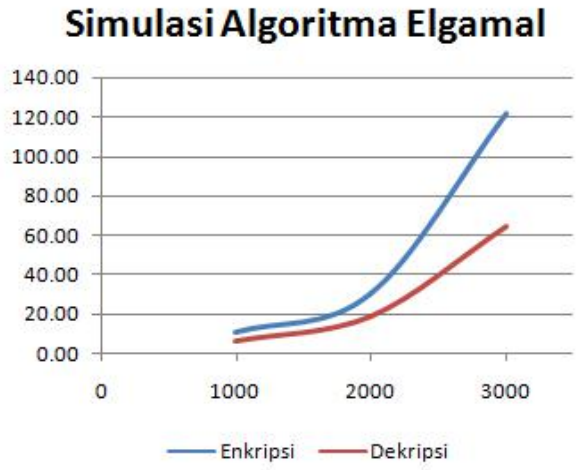


Gambar 5 Antarmuka Progam

Pengujian pertama dilakukan untuk algoritma ElGamal. Menggunakan 4 digit integer kunci privat, berikut adalah hasil eksekusinya :

Tabel 1 Hasil simulasi algoritma ElGamal

Perlakuan	Ukuran sampel teks	Waktu Enkripsi	Waktu Dekripsi
1	1000	10,8	5,9
2	2000	30,2	19
3	3000	122	64,2



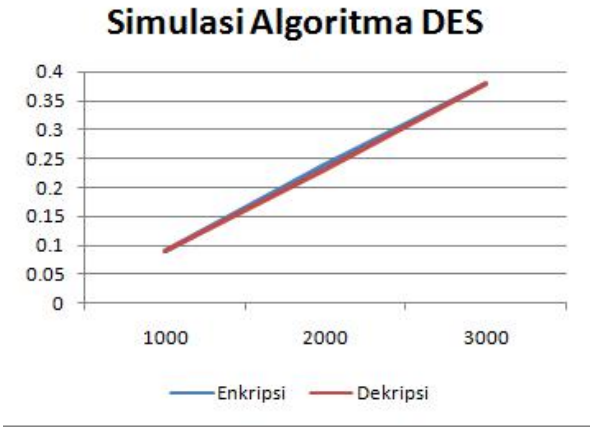
Gambar 6 Grafik Simulasi Algoritma ElGamal

Terlihat pada Gambar 6 hasil simulasi untuk algoritma ElGamal menunjukkan

Bagian selanjutnya, algoritma yang akan diuji-cobakan adalah DES. Dengan menggunakan 64 bit kunci privat. Berikut adalah hasil dari eksekusinya :

Tabel 2 Hasil simulasi algoritma DES

Perlakuan	Ukuran sampel teks	Waktu Enkripsi	Waktu Dekripsi
1	1000	0.09	0.09
2	2000	0.24	0.23
3	3000	0.38	0.38



Gambar 7 Grafik Simulasi Algoritma DES

Terlihat dari kedua tabel, menunjukkan peningkatan waktu proses yang mendekati linier sesuai dengan kompleksitasnya. Jika kedua eksekusi algoritma dibandingkan, baik pada proses enkripsi maupun dekripsi, algoritma DES membutuhkan waktu yang relatif lebih singkat untuk menyelesaikan proses enkripsi/dekripsi, hal ini dapat disebabkan karena pada algoritma DES, hanya menggunakan operator logika XOR dan pengacakan bit, sedangkan pada algoritma ElGamal menggunakan operator aritmatika perpangkatan, perkalian dan modulo. Nilai kunci privat *x* pada algoritma ElGamal pun turut mempengaruhi lamanya proses enkripsi/dekripsi, penggunaan *x* yang sangat besar akan menambah lama waktu proses algoritma, sedangkan pada algoritma DES lama waktu proses algoritma hanya dipengaruhi oleh penambahan masukan *n*.

4. KESIMPULAN

Kompleksitas waktu untuk algoritma ElGamal adalah kuadratik, sementara algoritma DES linier. Hal ini disebabkan pada algoritma ElGamal terdapat perulangan bersarang yang membuat proses menjadi lebih lama. Sedangkan pada algoritma DES proses yang terjadi hanya pengacakan bit plainteks yang membuat proses tidak terlalu berat.

Kompleksitas waktu pada algoritma ElGamal lebih dipengaruhi oleh panjang kunci privatnya, selain panjang plainteks. Sehingga notasi kompleksitas waktu untuk algoritma Elgamal adalah $O(x^2)$, dengan x adalah kunci privat. Sedangkan pada algoritma DES, kompleksitas waktu hanya bergantung pada panjang plainteksnya sehingga notasi kompleksitas waktu untuk algoritma DES adalah $O(n)$ dengan n adalah panjang plainteks.

ISSN : 2598-294X
E-ISSN : 2598-2958

DAFTAR PUSTAKA

[1] Mathur, A. 2012. A Research paper : An ASCII Value Based Data Encryption Algorithm and Its Comparison with Other Symmetric Data Encryption Algorithms. *International Journal on Computer Science and Engineering (IJCSE)*, 4(9), 1650–1657.

[2] Kabetta, H., & Dwiandiyanta, B. 2012. Information Hiding in CSS: A Secure Scheme Text-Steganography using Public Key Cryptosystem. *arXiv preprint arXiv:1201.1968*. Retrieved from http://arxiv.org/abs/1201.1968

[3] Denning, D. E., & Sacco, G. M. 1981. Timestamps in Key Distribution Protocols. *Communications of the ACM*, 24(8), 533–536. doi:10.1145/358722.358740

[4] Jeeva, A. L., Palanisamy, D. V, & Kanagaram, K. 2012. Comparative Analysis of Performance Efficiency and Security Measures of Some Encryption Algorithms. *International Journal of Engineering Research and Applications (IJERA) ISSN*, 2(3), 2248–9622.

[5] Singh, P., & Singh, K. 2013. Image Encryption and Decryption Using Blowfish Algorithm In MATLAB. *International Journal of Scientific & Engineering Research*, 4(7), 150–154.

[6] Hanno, B. 2015. A Look At the PGP Ecosystem through the Key Server Data. *IACR Cryptology ePrint Archive*, 262.

[7] Bernhard, D., Fischlin, M., & Warinschi, B. 2016. On the Hardness of proving CCA-Security of Signed ElGamal. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9614*, 47–69. doi:10.1007/978-3-662-49384-7_3

[8] Bahri, R. S., & Maliki, I. 2012. Feature Extraction Pada Optical Character Recognition. *Jurnal Komputer dan Informatika (KOMPUTA)*, 1(I).

[9] Smart, N. P., Paterson, K., & Cramer, R. 2015. *Cryptography Made Simple*. doi:10.1007/978-3-319-21936-3

[10] Bajaj, S., & Khajuri, K. 2016. Enhancement of Security in Visual Cryptography using DES. *International Journal of Engineering Science and Computing*, 6(7).