STEGO KRIPTO BERBASIS *NOISE* DENGAN MENGGUNAKAN KURVA ELIPTIK

NOISE BASED STEGANOGRAPHY CRYPTOGRAPHY USING ELIPTIC CURVE

M. Luthfan Alwafi[1], Dr. Marisa W Paryasto, S.T., M.T.[2], Dr. Ida Wahidah, S.T., M.T.[3]

[1,3]Prodi S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Universitas Telkom

[2]Prodi S1 Sistem Komputer, Fakultas Teknik Elektro, Universitas Telkom

[1]luthfanivan@gmail.com [2]marisa.paryasto@gmail.com [3]wahidah@telkomuniversity.ac.id

ABSTRAK

Teknik *hacking* yang semakin berkembang menjadikan sebuah pesan tidak lagi terjaga kerahasiaannya. Selain itu, pencurian data yang bersifat pribadi juga menjadi salah satu masalah yang harus diatasi. Teknik kriptografi dan steganografi menjadi salah satu solusi untuk mengatasi masalah tersebut.

Tugas akhir ini membahas tentang teknik kriptografi yang digunakan untuk meningkatkan keamanan suatu pesan. Teknik yang digunakan adalah kriptografi-steganografi yang berbasis *noise*. Teknik ini merupakan gabungan dari teknik kriptografi dan steganografi, dimana sebuah *noise* akan dienkrip dan disisipkan di anatara deretan pesan yang dikirim. Algoritma yang digunakan adalah algoritma El-Gamal kurva eliptik (*Eliptic Curve Crytography*) yang kemudian dimodifikasi (*Modified El-Gamal*). Algoritma ini juga yang akan digunakan pada proses dekripsi di penerima, dimana penerima akan mengidentifikasi pesan yang bernilai acak dan besar (*noise*), yang merupakan titik di luar kurva dari persamaan kurva yang telah ditentukan sebagai persamaan proses enkripsi dan dekripsi.

Dengan teknik tersebut, sebuah *noise* akan terlihat seperti pesan terenkripsi biasa. Sehingga, hanya penerima yang berhak yang mengetahuinya. Maka dari itu, semakin banyak *noise* yang disisipkan, maka waktu dekripsi penerima berhak akan semakin cepat dan kompleksitasnya juga menurun. Sedangkan penerima tidak berhak memerlukan waktu yang lebih lama saat mendekripsi pesan dan juga kompleksitasnya meningkat. Dengan demikian, tugas akhir ini menjawab isu tersebut, dimana tingkat keamanan sebuah pesan akan meningkat dan terjaga kerahasiannya.

Kata kunci : Kriptografi-Steganografi, Eliptic Curve Cryptography, Modified El Gamal, Noise

ABSTRACT

The growing hacking technique makes a message no longer kept secret. In addition, personal data theft is a problem that must be resolved. Cryptography and steganography techniques are one of the solutions to resolve that problem.

This final project discusses cryptographic techniques used to improve the security of a message. The technique used is noise-based cryptography-steganography. This technique is a combination of cryptography and steganography techniques, where a noise will be encrypted and inserted between the lines of messages sent. The alogorithm used is Eliptic Curve Cryptography (ECC) algorithm which is the modified (Modified El-Gamal). This algorithm will also be used in decryption process in the receiver, where the receiver will identify the message of random and big valuable (noise), which is the point beyond the curve of the curve equation that has been defined as the equation of the encryption and decryption process. With this technique, a noise will look like a normal encrypted message. So, only authorized user knows it. Therefore, the more noise is inserted, then the authorized user's decryption time will become fast and the complexity is decreases. While the unauthorized user need more time when decrypting messages and also increasing the complexity. Thus, this final task answer the issues, where the security level of a message will increased and a message will be kept confidential.

Keyword : Cryptography-Steganography, Eliptic Curve Cryptography, Modified El-Gamal, Noise

1. Pendahuluan

Salah satu teknik kriptografi yang bisa digunakan untuk meningkatkan keamanan suatu data adalah teknik kriptografi-steganografi yang berbasis noise. Teknik steganografi yang digunakan adalah menyisipkan noise pada deretan pesan. Sebuah noise akan dienkripsi dan kemudian disisipkan ke dalam deretan pesan yang sudah dienkripsi. Proses enkripsi dilakukan satu per satu. Setelah semua pesan termasuk noise telah dienkripsi menggunakan kunci publik (public key), semua pesan tersebut dikumpulkan menjadi sebuah deretan pesan yang kemudian dikirimkan dan hanya bisa didekripsi oleh penerima yang memiliki kunci rahasia (private key). Dengan demikian, orang – orang yang berniat mencuri, menghapus, memanipulasi, atau bahkan yang hanya sekedar ingin membaca tidak akan pernah tau bahwa ada noise di deretan pesan tersebut. Sehingga, hacker akan melihat noise tersebut sebagai pesan terenkripsi biasa.

Pada penelitian ini dibuat sebuah program yang dapat mengilustrasikan sebuah pengiriman dan penerimaan pesan maupun *noise*. Keluaran dari program ini adalah waktu eksekusi saat enkripsi dan dekripsi. Program juga akan menampilkan apakah pesan yang dikirim akan kembali seperti saat dikirim ataukah ada perubahan. Dari keluaran program ini akan dianalisis waktu eksekusinya, kompelksitasnya, dan juga akurasinya.

2. Tinjauan Pustaka

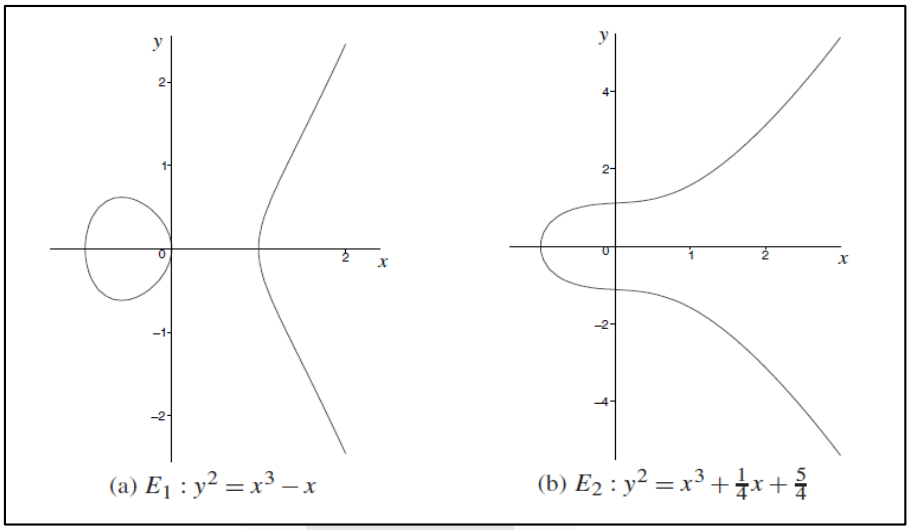
2.1. Kriptografi dan Steganografi

Kriptografi adalah ilmu yang mempelajari bagaimana menyandikan pesan. Kriptografi bersandarkan pada teknik matematika untuk berurusan dengan keamanan informasi seperti kerahasiaan, keutuhan data, dan otentikasi entitas. Sedangkan steganografi adalah ilmu yang menyembunyikan teks pada media lain yang telah ada sedemikian sehingga teks yang tersembunyi menyatu dengan media itu. Media tempat persembunyian bisa berupa teks, gambar, audio, ataupun video [4].

2.2. Kriptografi El-Gamal Kurva Eliptik (ECC)

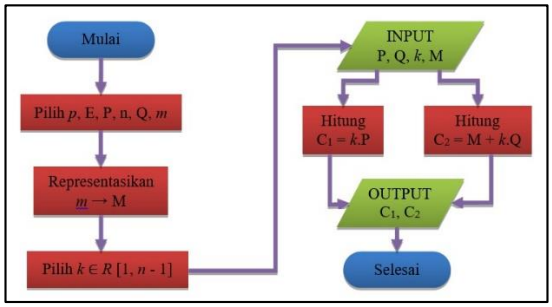
Kriptografi El-Gamal merupakan kriptografi yang bersifat asimetris atau dapat juga disebut sebagai kriptografi pertukaran kunci. Kurva eliptik adalah sekumpulan solusi yang memenuhi persamaan kubik yang melibatkan dua variable [2] [5]. Sebuah kurva eliptik memenuhi persamaan :

$$y^2 = x^3 + ax + b \tag{2.1}$$

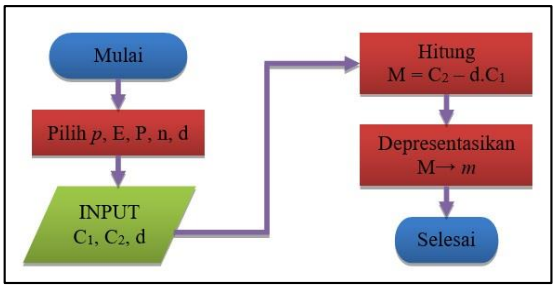


Gambar 2.1 Sebuah kurva eliptik pada bilangan real [2] [5].

Sebuah *plaintext* akan diubah menjadi bilangan integer yang besar, kemudia dimasukan ke kurva dan dikalikan dengan sebuah bilangan prima besar yang acak untuk mendapatkan *ciphertext*. Proses dekripsi pesan dilakukan dengan cara mengalikan *ciphertext* dengan kunci rahasia (*privat key*) untuk mendapatkan *plaintext*.



Gambar 2.2 Proses enkripsi El-Gamal.

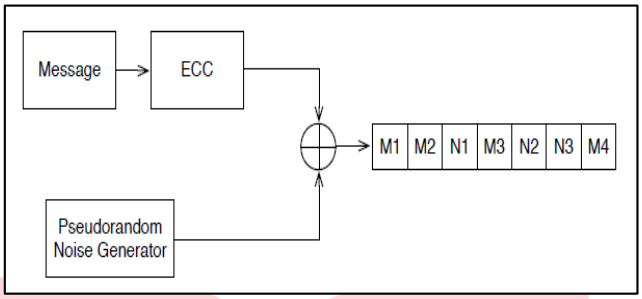


Gambar 2.3 Proses dekripsi El-Gamal

3. Perancangan Sistem

3.1. Pembangkitan *Noise* dan Penyisipan *Noise*

Noise dihasilkan oleh *pseudorandom noise* generator, dibangkitkan dari deretan angka acak k1, k2, … dengan $n_i = f(k_i)$, dimana f adalah fungsi El-Gamal termodifikasi. Penyisipan *noise* ditunjukan oleh gambar 2.4.



Gambar 2.4 Penyisipan *noise* pada pesan terenkripsi

Deretan *noise* N1, N2, dan N3 disisipkan secara acak pada pesan terenkripsi M1, M2, M3, dan M4. Format panjang *noise* dan pesan terenkripsi haruslah sama. Jika pesan terenkrip memiliki panjang 299-bit, maka panjang *noise* juga haruslah 299-bit [4].

3.2. *Pollard's Rho Attack* Sebagai Kompleksitas Kriptografi

Pollard's Rho digunakan untuk menemukan pasangan berbeda (c', c'') dan (d', d'') integer modulo n, sehingga :

$$c'P + d'Q = c''P + dQ''$$

Kemudian

$$(c' - c'')P = (d'' - d')Q = (d'' - d')lP$$

dan juga

$$(c' - c'') \equiv (d'' - d')l \ (mod\ n)$$

Dimana $l = log_p\ Q$ dapat diperoleh dengan menghitung :

$$l = (c' - c'')(d'' - d')^{-1}\ mod\ n \qquad (3.1)$$

3.3. Menghitung Kompleksitas Algoritma Menggunakan *Big O Notaion*

Analisis menggunakan *Big* O dilakukan untuk melihat performansi sebuah algoritma, dimana waktu eksekusi sebuah algoritma (*time complexity*) akan terus meningkat seiring dengan bertambahnya jumlah data/bertambahnya operasi yang ada pada sebuah algoritma. Namun, jika beberapa operasi dilakukan dalam satu langkah (*space complexity*), maka waktu eksekusi sebuah algoritma bisa menjadi lebih cepat. Maka dari itu, *time complexity* berbanding terbalik dengan *space complexity*.

Tabel 3.1 Kompleksitas Asimtotik

Kompleksitas Asimtotik	Kategori
O(1)	**Konstan**
O(log n)	**Logaritmik**
O(n)	**Linear**
O(n log n)	**n log n**
$O(n^2)$	**Kuadratik**
$O(n^3)$	**Kubik**
$O(2^n)$	**Eksponensial**
O(n!)	**Faktorial**

3.4. Kompleksitas Total

Menghitung kompleksitas total memerlukan waktu eksekusi rata – rata dan orde kurva. Waktu eksekusi rata – rata didapat dari hasil pengujian, sedangkan orde kurva didapatkan dari perhitungan cepat menggunakan PARI/GP. Kompleksitas Enkripsi dapat dihitung dengan persamaan :

$$O(e)_{total} = (pesan + noise) \times orde\ kurva \times \bar{t}(e)_{total} \qquad (3.2)$$

Kompleksitas Dekripsi untuk penerima berhak dihitung dengan persamaan :

$$O(d)_{total} = pesan \times orde\ kurva \times \bar{t}(d)_{pesan} \qquad (3.3)$$

Kompleksitas Dekripsi untuk penerima tidak berhak (*hacker*) dihitung dengan persamaan :

$$O(d)_{total} = (pesan + noise) \times orde\ kurva \times \bar{t}(d)_{total} \quad (3.4)$$

4. Skenario Pengujian, Hasil Perbandingan, dan Kompleksitas

4.1. Skenario Pengujian

Proses penyisipan *noise* dilakukan dengan meletakan *noise* pada deretan pesan yanga akan dikirim dan letaknya sesuai keinginan pengirim. Selain itu, penerima yang berhak sudah mengetahui letak *noise* di antara pesan yang dikirm.

Pengujian dilakukan dengan beberapa skenario, di antaranya :

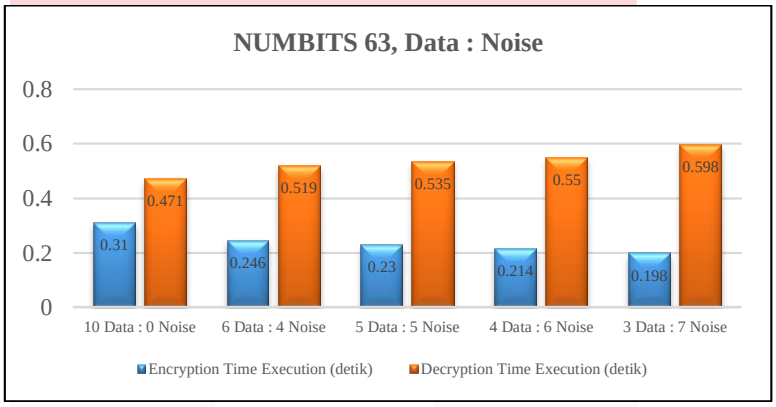
- Menggunakan NUMBITS 63, 93, dan 155
- Setiap NUMBITS dilakukan pengujian sebanyak lima kali dengan komposisi 10 pesan : 0 *noise*, 6 pesan : 4 *noise*, 5 pesan : 5 *noise*, 4 pesan : 6 *noise*, dan 3 pesan : 7 *noise*.
- *Noise* merupakan pesan juga, namun tidak melewati proses "poly_embed"

4.2. Perbandingan Data : *Noise*

Perbandingan data : *noise* dilakukan pada NUMBITS 63, 93, dan 155

a) NUMBITS 63

Hasil perbandingan dari pengujian dengan NUMBITS 63 ditunjukan oleh gambar 4.1

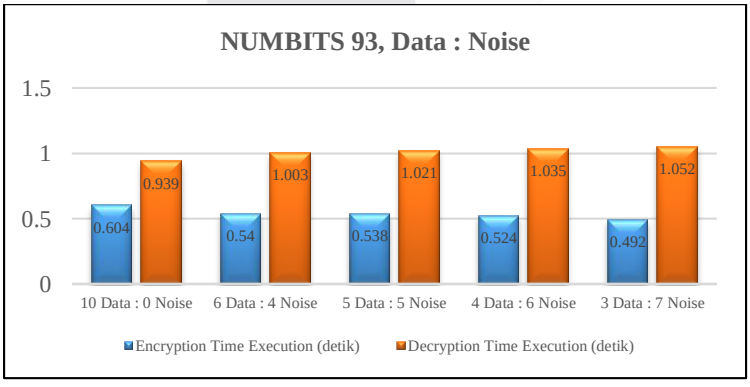


Gambar 4.1 Perbandingan data : *noise* dengan NUMBITS 63.

Pengujian menunjukan ketika *noise* ditambahkan, maka waktu enkripsi menjadi semakin cepat dan waktu dekripsi menjadi semakin lambat. Hal ini dikarenakan *noise* tidak melalui proses "poly_embed", sehingga menyebabkan proses enkripsinya menjadi semakin cepat ketika *noise*nya semakin banyak. Selain itu, *noise* yang tidak melalui proses "poly_embed" menyebabkan *noise* menjadi bilangan acak, sehingga memperlambat proses pencarian titik dalam kurvanya yang menyebabkan dekripsinya menjadi lambat.

b) NUMBITS 93

Hasil perbandingan dari pengujian dengan NUMBITS 93 ditunjukan oleh gambar 4.2

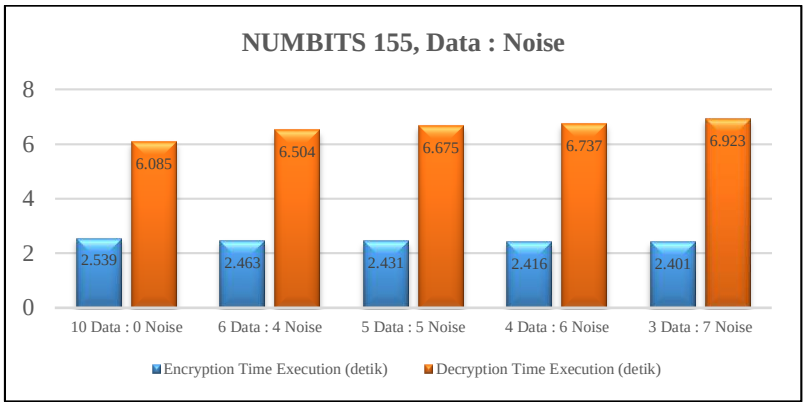


Gambar 4.2 Perbandingan data : *noise* dengan NUMBITS 93.

Pengujian menunjukan penambahan NUMBITS mengakibatkan waktu enkripsi dan dekripsi menjadi lebih lambat. Hal ini dikarenakan penambahan NUMBITS berarti jumlah titik dalam kurva semakin banyak, sehingga menyebabkan proses enkripsi dan dekripsi menjadi lebih lambat.

c) NUMBITS 155

Hasil perbandingan dari pengujian dengan NUMBITS 155 ditunjukan oleh gambar 4.3

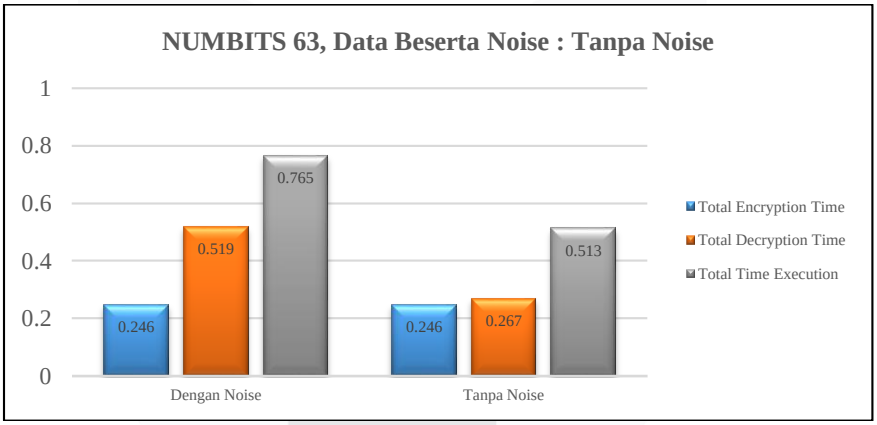


Gambar 4.3 Perbandingan data : *noise* dengan NUMBITS 155.

Pengujian menunjukan NUMBITS 155 adalah pengujian dengan waktu enkripsi dan dekripsi paling lambat. Meskipun demikian, dengan waktu dekripsi yang paling lambat dibandingkan menggunakan NUMBITS 63 dan 93, menggunakan NUMBITS 93 akan meningkatkan tingkat keamanan data yang dikirimkan. Hal ini dibuktikan dengan semakin besarnya NUMBITS berarti semakin banyak titik dalam kurva, sehingga menyebabkan penerima tidak berhak (*hacker*) akan semakin sulit dalam menemukan titik sesungguhnya (pesan sesungguhnya).

4.3. Perbandingan Data Beserta *Noise* : Tanpa *Noise*

Penerima berhak mengetahui letak *noise*, sehingga tidak perlu mendekripsi semua data yang dikirim. Maka dari itu, dilakukan pengujian untuk melihat perbandingan antara jika penerima mendekripsi semua pesan dengan jika penerima hanya mendekripsi pesannya saja (tidak mendekripsi *noise*). Hasil perbandingan ditunjukan oleh gambar 4.4



Gambar 4.4 Waktu enkripsi, dekripsi, dan total beserta *noise* : tanpa *noise*.

Pengujian menunjukan saat penerima tidak mendekripsi *noise*, waktu dekripsi menjadi jauh lebih cepat. Hal ini dikarenkan ketika penerima tidak mendekripsi *noise* berarti jumlah data yang penerima dekripsi lebih sedikit, sehingga waktu dekripsinya menjadi lebih cepat.

4.4. Kompleksitas

a) Kompleksitas kriptografi

Kompleksitas Kriptografi dihitung dari algoritma *Pollard's Rho* dengan menggunakan *Big* O *notation*. *Pollard's Rho attack* dipilih karena serangan ini adalah serang yang paling ideal untuk algoritma El-gamal.

1. Pilih akar dari L (misal, $L = 16$ atau $L = 32$). **1**
2. Pilih fungsi pembagian $H : (P) \rightarrow \{1, 2, \ldots, L\}$. **1**
3. Untuk j dari 1 sampai L, lakukan : **n**
   - Pilih $a_j, b_j \in R\ [0, n-1]$ **1**
   - Hitung $R_j = a_j P + b_j Q$. **1**
4. Pilih $c', d' \in R\ [0, n-1]$ dan hitung $X' = c'P + d'Q$ **1**
5. Atur $X'' \leftarrow X', c'' \leftarrow c', d'' \leftarrow d'$. **1**
6. Ulangi :
   - Hitung $j = H(X')$. **1**

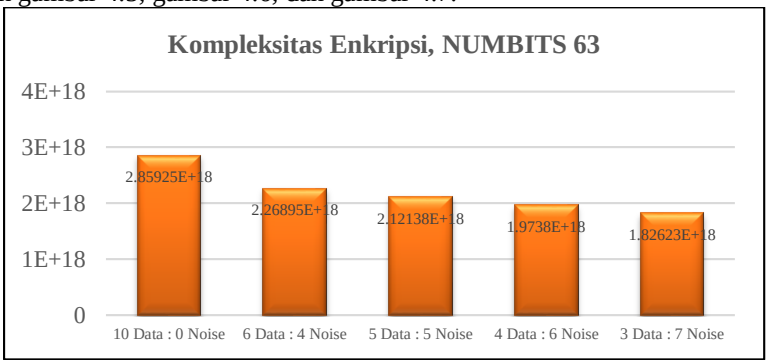
Atur $X' \leftarrow X' + R_j, c' \leftarrow c' + a_j \bmod n,\ d' \leftarrow d' + b_j \bmod n.$ **1**

- Untuk i dari 1 sampai 2, lakukan : **2**
  Hitung $j = H(X'')$. **1**
  Atur $X'' \leftarrow X'' + R_j, c'' \leftarrow c'' + a_j \bmod n,\ d'' \leftarrow d'' + b_j \bmod n.$ **1**
  Sampai $X' = X''$. **n**

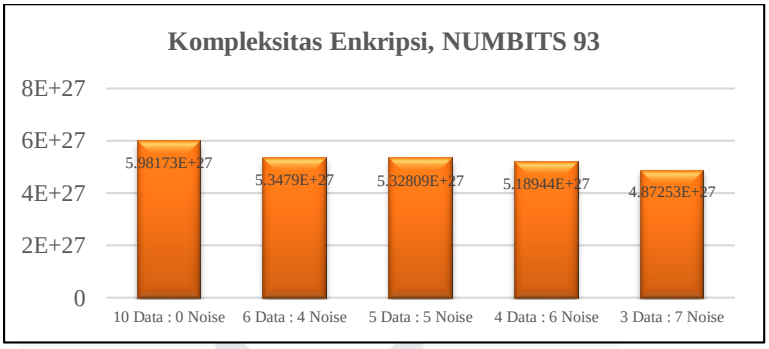
7. Jika $d' = d''$, maka ("gagal");
   Jika tidak, maka hitung $l = (c' - c'')(d'' - d')^{-1} \bmod n$ dan dapatkan (l). **1**

Time Complexity :
O(1) + O(1) + n*{1 + 1} + O(1) + O(1) + n*{1 + 1 + 2*[1 + 1]} + O(1) = 8O(n) = O(n)
Space Complexity :
O(1) + O(1) + n*{1*1} + O(1) + O(1) + n*{1*1 + 2*[1*1]} + O(1) = 3O(n) = O(n)

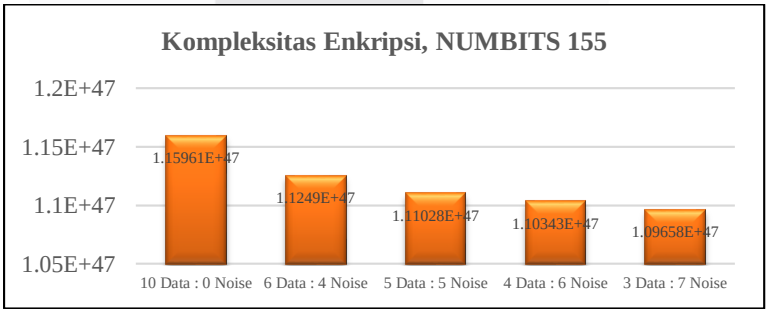
b) Kompleksitas total enkripsi
Kompleksitas total enkripsi dihitung dengan menggunakan persamaan 3.2. Hasil dari perhitungan ditunjukan oleh gambar 4.5, gambar 4.6, dan gambar 4.7.



Gambar 4.5 Kompleksitas enkripsi NUMBITS 63.



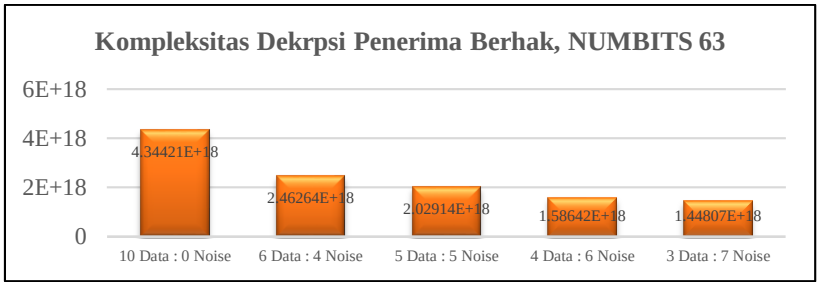
Gambar 4.6 Kompleksitas enkripsi NUMBITS 93.



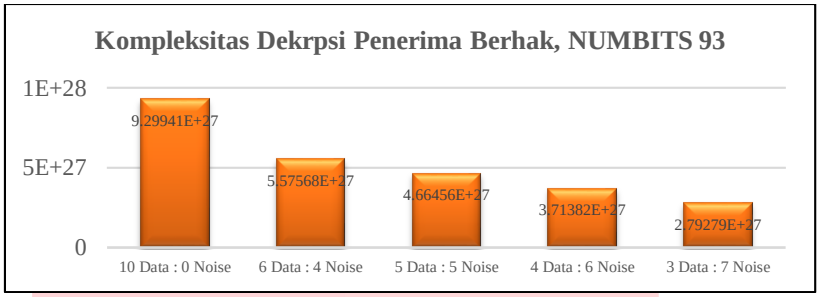
Gambar 4.7 Kompleksitas enkripsi NUMBITS 155.

Pada saat enkripsi, penambahan jumlah *noise* akan mengurangi kompleksitas, hal ini dikarenakan saat enkripsi *noise* tidak melalui proses "poly_embed", sehingga dapat mengurangi kompleksitas. Selain itu, penambahan NUMBITS akan menambahkan kompleksitas, hal ini dipengaruhi oleh banyaknya jumlah titik dalam kurva, semakin banyak titik maka akan semakin kompleks.

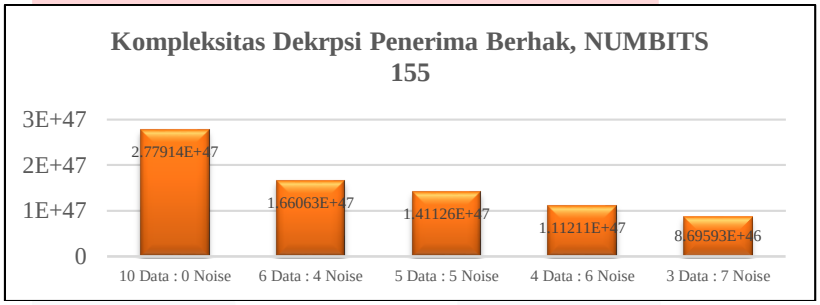
c) Kompleksitas total dekripsi penerima berhak
Kompleksitas total dekripsi untuk penerima yang berhak dihitung dengan menggunakan persamaan 3.3. Hasil dari perhitungan ditunjukan oleh gambar 4.8, gambar 4.9, dan gambar 4.10.



Gambar 4.8 Kompleksitas dekripsi penerima berhak NUMBITS 63.



Gambar 4.9 Kompleksitas dekripsi penerima berhak NUMBITS 93.

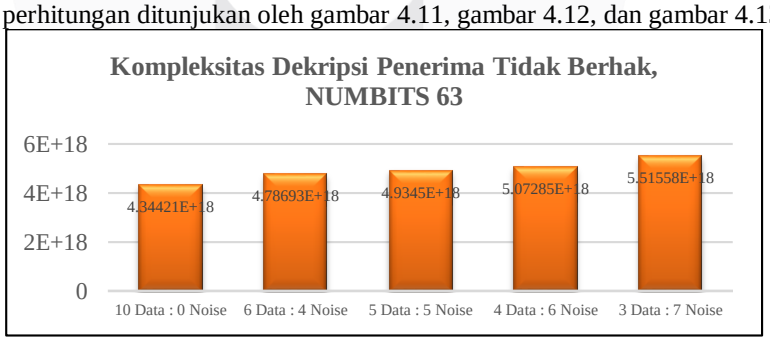


Gambar 4.10 Kompleksitas dekripsi penerima berhak NUMBITS 155.

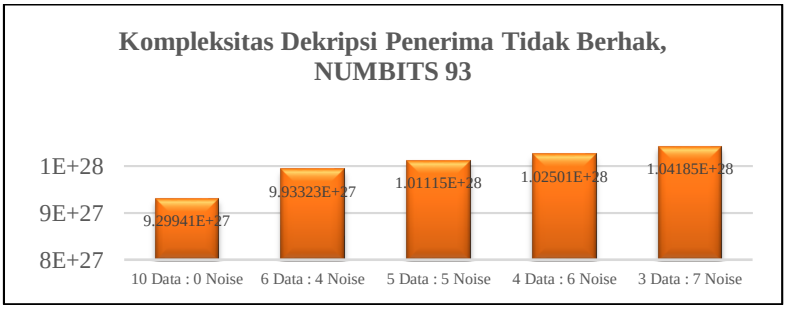
Pada saat dekripsi, penerima yang berhak tidak perlu mendekripsi *noise*, sehingga menyebabkan kompleksitas akan semakin menurun jika *noise* yang disisipkan pada pesan semakin banyak. Selain itu, penambahan NUMBITS akan menambahkan kompleksitas, hal ini dipengaruhi oleh banyaknya jumlah titik dalam kurva, semakin banyak titik maka akan semakin kompleks.

d) Kompleksitas dekripsi penerima tidak berhak

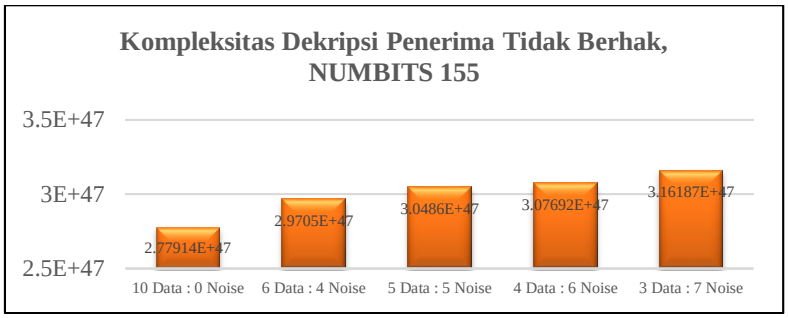
Kompleksitas total dekripsi untuk penerima yang berhak dihitung dengan menggunakan persamaan 3.4. Hasil dari perhitungan ditunjukan oleh gambar 4.11, gambar 4.12, dan gambar 4.13.



Gambar 4.11 Kompleksitas dekripsi penerima tidak berhak NUMBITS 63.



Gambar 4.12 Kompleksitas dekripsi penerima tidak berhak NUMBITS 93.



Gambar 4.13 Kompleksitas dekripsi penerima tidak berhak NUMBITS 155.

Pada saat dekripsi, penerima yang tidak berhak diharuskan mendekripsi semua pesan (termasuk *noise*). *Noise* yang menjadi bilangan acak akan mempersulit penerima yang tidak berhak, sehingga menyebabkan kompleksitas akan meningkat saat *noise*nya semakin banyak. Selain itu, penambahan NUMBITS akan menambahkan kompleksitas, hal ini dipengaruhi oleh banyaknya jumlah titik dalam kurva, semakin banyak titik maka akan semakin kompleks.

5. Kesimpulan

Berdasarkan percobaan dan analisis yang telah dilakukan, maka pada tugas akhir ini dapat ditarik beberapa kesimpulan, yaitu :

a. Metode stego-kripto berbasis *noise* terbukti dapat meningkatkan tingkat keamanan suatu data, dikarenakan penyisipan *noise* mengakibatkan penerima tidak berhak (*hacker*) diharuskan mendekripsi semua data termasuk *noise*, sedangkan penerima yang berhak tidak perlu mendekripsi *noise*, tapi cukup mendekripsi data yang berisi pesannya saja. Selain itu, *noise* yang berupa pesan yang tidak dimasukan ke kurva sehingga bernilai acak membuat waktu proses dekripsi menjadi semakin lama.
b. Untuk mengukur tingkat keamanan metode ini, dilakukan beberapa percobaan dengan jumlah data yang sama, namun komposisinya berbeda. Dengan jumlah data yang sama, jumlah pesan yang lebih sedikit dan jumlah *noise* yang lebih banyak, terbukti dapat meningkatkan tingkat keamanan data tersebut, dapat dilihat dari grafik yang menunjukan waktu dekripsi yang semakin melambat ketika jumlah *noise* ditambahkan. Waktu dekripsi yang semakin melambat menandakan bahwa mendekripsi *noise* lebih sulit dibandingkan mendekripsi pesan, dikarenakan *noise* merupakan pesan yang bernilai acak. Selain itu, jika *hacker* telah berhasil mendekripsi semua pesan, *hacker* akan membutuhkan waktu tambahan untuk menerjemahkan pesan tersebut, dikarenakan ada *noise* di antara pesan yang sesungguhnya.
c. Dari ketiga NUMBITS yang digunakan pada percobaan, proses enkripsi tercepat adalah saat menggunakan NUMBITS 63 sedangkan proses enkripsi yang paling lambat adalah saat menggunakan NUMBITS 155. Namun, NUMBITS 155 terbukti dapat lebih meningkatkan tingkat keamanan, karena dengan NUMBITS 155 titik di dalam kurva akan semakin banyak, yang mengakibatkan *hacker* akan semakin sulit untuk mendapatkan titik yang merupakan sebuah pesan.
d. Hubungan waktu eksekusi berbanding lurus dengan kompleksitas. Ketika enkripsi, semakin banyak *noise*, maka waktu enkripsi akan semakin cepat dan juga kompleksitas semakin menurun. Ketika penerima berhak mendekripsi, semakin banyak *noise*, maka waktu dekripsi akan semakin cepat dan juga kompleksitasnya menurun. Ketika penerima tidak berhak mendekripsi, semakin banyak *noise*, maka waktu dekripsi semakin lambat dan juga kompleksitas semakin meningkat.

Daftar Pustaka :

[1] Ash, A., & Gross, R. (2012). *Eliptic Tales: Curves, Counting, and Number Theory.* Oxford: Princeton University Press.
[2] Hankerson, D., Menezes, A., & Vanstone, S. (2004). *Guide to Eliptic Curve Cryptography.* New York: Springer.
[3] Menez, A. J., Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of Applied Cryptography.* Cambridge, Massachusetts: Webster Professor of Electrical Engineering and Computer Science.
[4] Rahardjo, B., Alamsyah, I. M., & Paryasto, M. W. (2014). EPJ Web of Conferences. *Noise-based Stego-ECC, 68*, 1-3.
[5] Sadikin, R. (2012). *Kriptografi untuk Keamanan Jaringan.* Yogyakarta: ANDI.
[6] Schneier, B. (1996). *Applied Cryptography, Second Edition: Protocols, Algorithm, and Source Code in C (cloth).* John Wiley & Sons, Inc.