Aplikasi Enkripsi Citra dan Teks Menggunakan Algoritma *Diffie-Hellman* dan *ElGamal*

Luthfiatun Nisa[1], Tutuk Indriyani[2], Maretha Ruswiansari[3]
Program Studi Teknik Informatika, Fakultas Teknik Informasi, Institut Teknologi Adhi Tama Surabaya
Email: nisanisa389@gmail.com

Abstract. *Data security becomes the most important thing in information storage. One of them is a security in exchanging messages. Therefore, an encryption method is needed. In encryption, there are symmetric and asymmetric encryptions. Asymmetric encryption is considered more secure because it uses two different key types than symmetric encryption which uses only one type of key. ElGamal is one of the asymmetric encryption algorithms in which advantage lies on the calculation of discrete logarithm that are difficult to solve. Meanwhile, Diffie-Hellman is one of the key exchage process algorithms that generates secret key during communication process to keep the key secrecy. In this research, a combination of Diffie-Hellman and ElGamal algorithms is used to secure text and image messages. the combination of these two algorithms consist of 4 processes, namely the process of key exchange, key generation, encryption, and decryption. The research results showed that 10 text files of 10 Kb to 100 Kb obtained the average time of encryption 119.9 ms, decryption 248.3 ms, and throughput 600.9 Kbps. Meanwhile, for image files with the size 100 x 100 pixels up to 1000 x 1000 pixels, the average time of encryption 2623.4 ms, decryption 4349.5 ms, Mean Square Error (MSE) value 213.95 with the decreasing percentage 27.67%, and value Peak Signal to Noise Ratio (PSNR) 173.27 dB with the increasing percentage 1.94%. in addition, the results of avalanche effect testing obtained that the shift bits percentage in the text file was 85.18% and that in the image file was 84.46%.*

Keywords: *Image, Diffie-Hellman, ElGamal. Encryption, Text.*

Abstrak. *Keamanan data menjadi suatu hal yang paling penting dalam penyimpanan informasi. Salah satunya adalah keamanan saat melakukan pertukaran pesan, baik berupa teks maupun citra. Untuk itulah metode enkripsi diperlukan. Dalam enkripsi, terdapat enkripsi simetris dan asimetris. Enkripsi asimetris dianggap lebih aman karena menggunakan dua jenis kunci yang berbeda dibandingkan enkripsi simetris yang hanya menggunakan satu jenis kunci. ElGamal merupakan salah satu algoritma enkripsi asimetris, kelebihannya terletak pada perhitungan logaritma diskrit yang sulit untuk dipecahkan. Sementara itu, Diffie-Hellman merupakan salah satu algoritma dalam proses pertukaran kunci yang menghasilkan secret key pada saat proses komunikasi untuk menjaga kerahasiaan kunci. Dalam penelitian ini, diterapkan kombinasi antara algoritma Diffie-Hellman dan ElGamal untuk mengamankan pesan teks dan citra. Kombinasi kedua algoritma ini terdiri dari 4 proses, yaitu proses pertukaran kunci, pembangkitan kunci, enkripsi, dan dekripsi. Dari hasil penelitian, untuk 10 file teks dengan ukuran 10 Kb hingga 100 Kb diperoleh rata-rata waktu enkripsi sebesar 119.9 ms, dekripsi sebesar 248.3 ms, serta throughput sebesar 600,96 Kbps. Sementara itu untuk file citra dengan ukuran 100×100 piksel hingga 1000×1000 piksel, diperoleh rata-rata waktu enkripsi sebesar 2623.4 ms, dekripsi sebesar 4349.5 ms, nilai Mean Square Error (MSE) sebesar 213.95 dengan presentase penurunan sebesar 27.67% , dan nilai Peak Signal to Noise Ratio (PSNR) sebesar 173.27 dB dengan presentase kenaikan sebesar 1.94%. Selain itu, dari hasil pengujian avalanche effect, diperoleh presentase pergeseran bit pada file teks sebesar 85.18% dan pada file citra sebesar 84.46%.*

Kata Kunci: *Citra, Diffie-Hellman, ElGama, Enkripsi, Teks.*

1. Pendahuluan

Keamanan data menjadi suatu hal yang paling penting dalam penyimpanan informasi. Salah satunya adalah keamanan saat melakukan pertukaran pesan. Apalagi jika pesan tersebut berisi informasi yang sangat penting dan bersifat rahasia yang menyangkut sebuah organisasi atau perusahaan. Kebutuhan akan keamanan data inilah yang mendorong terciptanya metode untuk mengamankan suatu data, metode inilah yang disebut dengan kriptografi. Kriptografi terdiri berbagai metode yang digunakan untuk mengamankan suatu data, salah satunya adalah enkripsi, yaitu mengubah informasi asli (*plaintext*) menjadi sandi-sandi atau kode yang tidak dimengerti (*ciphertext*) dan tidak dapat di dekripsi oleh pihak yang tidak memiliki wewenang atas informasi tersebut.

Terdapat dua jenis enkripsi berdasarkan jumlah kuncinya, yaitu enkripsi simetris yang menggunakan satu jenis kunci dan enkripsi asimetris (*public key encryption*) yang menggunakan dua jenis kunci (*public key* dan *private key*). Dalam proses enkripsi, kunci merupakan elemen penting, untuk itu kerahasiaan kunci sangat diperlukan. Salah satu metode yang digunakan untuk menjaga kerahasiaan kunci adalah dengan pertukaran kunci. Metode ini memungkinkan kedua pihak saling bertukar *secret key* yang hanya diketahui oleh pihak tersebut dan kemudian dapat digunakan untuk enkripsi pesan berikutnya. *Diffie-Hellman* merupakan salah satu algoritma dalam pertukaran kunci, algoritma ini terbatas pada proses pertukaran kunci saja, sehingga membutuhkan algoritma lain untuk proses enkripsi dan dekripsi. Efektifitas algoritma ini adalah sulitnya menghitung logaritma diskrit. Permasalahannya adalah bagaimana cara mendistribusikan *secret key* tersebut secara aman. Apabila pendistribusian *secret key* salah, maka informasi dapat mudah diakses oleh pihak lain dalam pengirimannya. Salah satu solusi dalam mendistribusikan secret key ini adalah dengan menggunakan enkripsi asimetris pada proses enkripsi dan dekripsinya. Beberapa algoritma enkripsi dan dekripsi asimetris diantaranya adalah *RSA*, *ElGamal*, *Rabin*, dan *ECC*.

Performansi algoritma dianggap *ElGamal* lebih baik dibandingkan dengan algoritma *RSA* pada skalabilitas, *power consumption*, serta implementasi pada *hardware* dan *software* [1]. Oleh karena itu, dalam penelitian ini, diterapkan algoritma *Diffie-Hellman* dengan *ElGamal*. Penelitian ini menggunakan algoritma Diffie-*Hellman* dalam proses pertukaran kunci, sementara proses pembangkitan kunci, enkripsi, dan dekripsi menggunakan algoritma *ElGamal*.

2. Metode

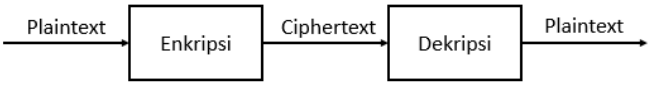
Pada penelitian ini, aplikasi yang dibangun adalah aplikasi enkripsi yang dapat mengirim pesan terenkripsi dan mendekripsi pesan yang diterima. Faktor yang mendasari dibentuknya aplikasi ini adalah keamanan data. Keamanan data telah menjadi aspek yang sangat penting dari suatu sistem informasi [2]. Untuk keperluan tersebut, maka diperlukan sebuah teknik kriptografi dengan menggunakan algoritma enkripsi dan dekripsi data. Algoritma enkripsi yang digunakan dalam penelitian ini adalah algoritma enkripsi Diffie-Hellman dan ElGamal.

2.1 Enkripsi dan Dekripsi

Enkripsi (*encryption*) merupakan suatu proses di mana sebuah pesan asli (*plaintext*) ditranformasikan atau diubah menjadi bentuk pesan lain yang tidak dapat dibaca (*ciphertext*) menggunakan suatu fungsi matematis kunci tertentu yang disebut *key*.

Sementara Dekripsi (*descryption*) merupakan proses kebalikan dari enkripsi, dimana *ciphertext* dirubah kembali ke *plaintext* dengan menggunakan fungsi matematis dan *key* [3].

Dalam Notasi Matematis, proses enkripsi dan dekripsi dapat dijabarkan menjadi E(p) = c, dimana fungsi enkripsi E memetakan *plaintext* p ke enkripsi *ciphertext* c. Sedangkan pada proses kebalikannya D(c) = p, dimana fungsi dekripsi D memetakan *ciphertext* c ke *plaintext* p. Gambar 1 menunjukkan proses enkripsi dan dekripsi.



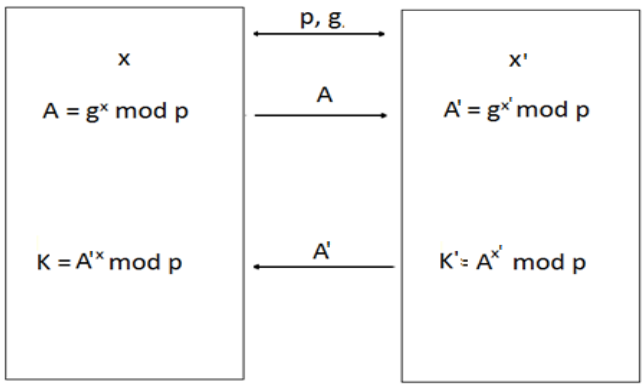
Gambar 1. Proses Enkripsi dan Dekripsi

2.2 Algoritma Enkripsi

Algoritma enkripsi dapat diartikan sebagai suatu fungsi matematis yang digunakan untuk melakukan enkripsi dan dekripsi [4]. Berdasarkan jumlah kuncinya, algoritma enkripsi dibagi menjadi dua macam yaitu algoritma simetris (*symmetric algorithms*) dan algoritma asimetris (*asymmetric algorithms*). Keamanan algoritma simetris tergantung pada kunci, membocorkan kunci berarti bahwa orang lain dapat mengenkripsi dan mendekripsi pesan. Agar komunikasi tetap aman, kunci harus tetap dirahasiakan. Sifat kunci yang seperti ini membuat pengirim harus selalu memastikan bahwa jalur yang digunakan dalam pendistribusian kunci adalah jalur yang aman atau memastikan bahwa seseorang yang ditunjuk membawa kunci untuk dipertukarkan adalah orang yang dapat dipercaya. Masalahnya akan menjadi rumit apabila komunikasi dilakukan secara bersama-sama oleh sebanyak n pengguna, maka risiko kebocoran kunci akan semakin besar. Sementara itu, algoritma asimetris memiliki kunci publik (*public*) dan kunci privat (*private*), dimana kunci publik bersifat umum, artinya kunci ini tidak dirahasiakan sehingga dapat dilihat oleh siapa saja. Sedangkan kunci privat adalah kunci yang dirahasiakan dan hanya orang-orang tertentu saja yang boleh mengetahuinya. Keuntungan utama dari algoritma ini adalah memberikan jaminan keamanan kepada siapa saja yang melakukan pertukaran informasi meskipun di antara mereka tidak ada kesepakatan mengenai keamanan pesan terlebih dahulu maupun saling tidak mengenal satu sama lainnya [4].

2.3 *Diffie-Hellman*

Algoritma kunci publik diterbitkan pertama kali dalam makalah Diffie dan Hellman yang didefinisikan sebagai kriptografi kunci publik dan biasanya disebut sebagai *Diffie-Hellman Key Exchange* (pertukaran kunci) atau Protokol *Diffie-Hellman*. Tujuan dari algoritma ini adalah untuk memungkinkan dua pengguna saling bertukar kunci secara aman, kemudian dapat digunakan untuk enkripsi dan dekripsi pesan berikutnya. Algoritma ini memungkinkan dua *user* saling bertukar *secret key* secara aman, kemudian dapat digunakan untuk enkripsi pesan berikutnya algoritma ini terbatas untuk pertukaran *key*. Efektifitas dari algoritma ini tergantung pada tingkat kesulitan dalam komputasi logaritma diskrit. Contoh Masalah logaritma diskrit: Jika *p* adalah bilangan prima dan *g* dan *y* adalah sembarang bilangan bulat, carilah *x* sedemikian sehingga $g^x = y \ (mod\ p)$. Secara umum algoritma *Diffie-Hellman* dapat digambarkan pada Gambar 2.



Gambar 2. Skema Algoritma Diffie-Hellman

2.4 *ElGamal*

Dibuat oleh Taher Elgamal pada tahun 1985. Pertama kali dikemukakan di dalam makalah berjudul "*A public key cryptosystem and a signature scheme based on discrete logarithms*". Algoritma ini merupakan bagian dari enkripsi asimetris dan merupakan *cipher* blok, yaitu melakukan proses enkripsi pada blok-blok plainteks dan menghasilkan blok-blok cipherteks yang kemudian dilakukan proses dekripsi dan hasilnya digabungkan [2].

Algoritma *ElGamal* terdiri dari tiga proses, yaitu proses pembentukan kunci, proses enkripsi dan proses dekripsi. Keamanan algoritma *ElGamal* terletak pada kesulitan penghitungan logaritma diskrit pada bilangan modulo prima yang besar sehingga upaya untuk menyelesaikan masalah logaritma ini menjadi sangat sukar. Algoritma ini mempunyai kelebihan pada enkripsi, yaitu untuk plainteks yang sama akan menghasilkan cipherteks yang berbeda. Algoritma ini juga akan menghasilkan jumlah cipherteks dua kali lipat dibandingkan plainteksnya.

2.5 *Throughput*

Throughput didefinisikan sebagai banyaknya data yang diterima dalam selang waktu pengiriman tertentu. Dalam enkripsi, throughput dapat diartikan sebagai perbandingan ukuran *plaintext* dan waktu enkripsi yang dibutuhkan [5]. *Throughput* dapat dirumuskan sebagai berikut:

$$Throughput = \frac{ukuran\ plaintext}{waktu\ enkripsi} \quad (1)$$

2.6 *Mean Square Error* (*MSE*)

MSE adalah salah satu metode untuk menghitung nilai error kuadrat rata-rata antara citra asli dan citra hasil dekripsi. Semakin kecil nilai *MSE*, maka dapat diartikan bahwa citra hasil dekripsi semakin baik [5].

$$MSE = \frac{1}{N \times M} \sum_{n=1}^{N} \sum_{m=1}^{M} [|f(i,j) - f_0(i,j)|^2] \quad (2)$$

M dan N merupakan dimensi citra, *f* dan f_0 merupakan fungsi intensitas dari citra asli dan citra hasil dekripsi, serta (i, j) menunjukkan posisi piksel.

2.7 *Peak Signal to Noise Ratio* (*PSNR*)

PSNR digunakan untuk mengetahui perbandingan kualitas citra asli dan citra setelah didekripsi. Untuk menentukan nilai *PSNR*, maka harus mengetahui nilai MSE terlebih dahulu. Kualitas citra dikatakan baik apabila nilai *PSNR* diatas 40 *dB* [6].

$$PSNR = 20 \times \log \frac{255^2}{\sqrt{MSE}} \quad (3)$$

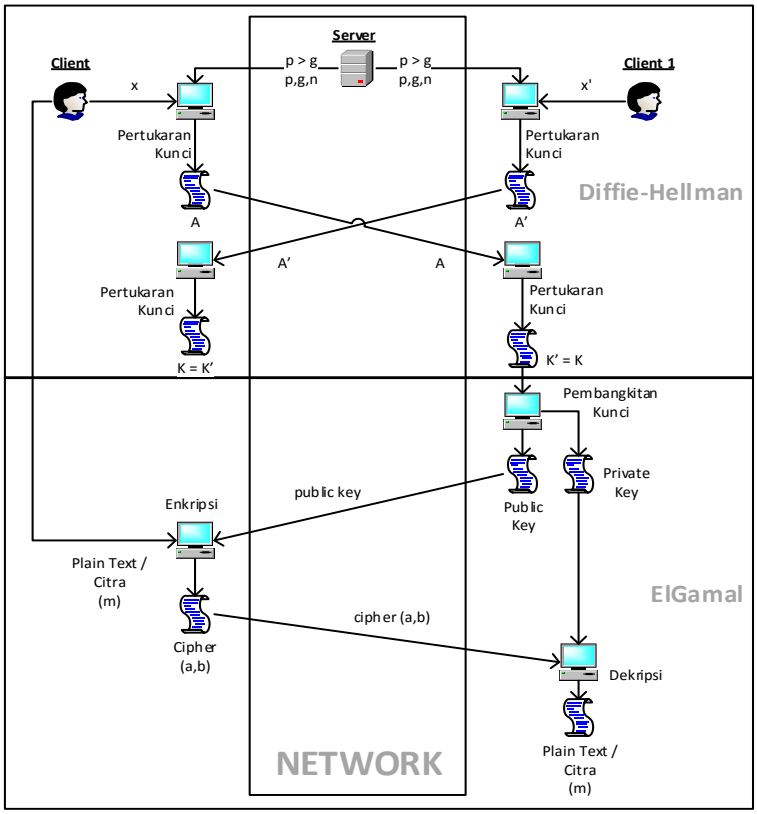
2.8 *Avalanche Effect*

Avalanche effect digunakan untuk mengetahui seberapa besar perubahan bit yang terjadi antara input dan output sebuah algoritma enkripsi [7] Suatu *avalanche effect* dikatakan baik apabila perubahan bit *yang d*ihasilkan berkisar antara 45 – 60 % atau lebih.

$$avalanche\ effect = \frac{jumlah\ perubahan\ bit}{jumlah\ seluruh\ bit\ cipherteks} \times 100\% \quad (4)$$

2.9 Gambaran Umum Sistem

Pada penelitian ini, adapun sistem yang dibangun merupakan aplikasi enkripsi pada pesan teks, *file rich text*, dan citra menggunakan hybrid algoritma *Diffie-Hellman* dan *ElGamal*. Proses dimulai saat pengirim dan penerima pesan saling melakukan pertukaran kunci, kemudian penerima akan membangkitkan kunci dan menghasilkan *public key* dan *private key*. *Public key* ini akan dikirimkan pada pengirim untuk mengenkripsi pesan, sementara *private key* akan digunakan penerima untuk mendekripsi pesan. Gambaran umum sistem pada penelitian ini dapat ditunjukkan pada Gambar 3.



Gambar 3. Desain Sistem

Pada Gambar 3, adapun sistem yang dibangun merupakan aplikasi enkripsi pada pesan teks, *file rich text*, dan citra menggunakan *hybrid* algoritma *Diffie-Hellman* dan *ElGamal*. Proses dimulai saat pengirim dan penerima pesan saling melakukan pertukaran kunci, kemudian penerima akan membangkitkan kunci dan menghasilkan *public key* dan *private key*. *Public key* ini akan dikirimkan pada pengirim untuk mengenkripsi pesan, sementara *private key* akan digunakan penerima untuk mendekripsi pesan.

2.10 Skenario Pengujian

Dalam penelitian ini, parameter yang ingin diuji adalah waktu komputasi, nilai *throughput*, nilai *mean square error* (MSE), nilai *peak signal to noise ratio* (PSNR), dan nilai *Avalance Effect*. Pengujian waktu komputasi dilakukan dengan menghitung waktu yang digunakan dalam proses enkripsi dan proses dekripsi. Pengujian throughput dilakukan dengan menghitung ukuran pesan sebelum dienkripsi dan membandingkannya dengan waktu enkripsi yang dibutuhkan. Pengujian nilai MSE dilakukan dengan cara menghitung nilai error kuadrat rata-rata antara citra asli dan citra hasil dekripsi. Semakin kecil nilai MSE, maka nilai error dianggap semakin kecil.

Pengujian PSNR digunakan untuk mengetahui perbandingan kualitas citra asli dan citra setelah didekripsi. Untuk menentukan PSNR, maka harus mengetahui nilai MSE terlebih dahulu. Semakin tinggi nilai PSNR, maka citra asli dan citra hasil dekripsi dianggap semakin mirip. Sementara itu, pengujian *avalanche effect* dilakukan untuk menghitung jumlah perubahan bit pada penerapan algoritma enkripsi. Pengujian ini dilakukan dengan cara membandingkan bit-bit *file input* (plainteks) dengan *bit-bit file output* (cipherteks).

3. Hasil dan Pembahasan

3.1 Perangkat Uji Coba

Untuk melakukan uji coba aplikasi ini menggunakan laptop dengan spesifikasi perangkat keras (*hardware*) dan perangkat lunak (*software*) sebagai berikut : (1) *Processor* Intel® Core™ i3-2330M CPU @ 2.20 GHz. (2) Memori RAM 2 GB. (3) *Hardisk* 500 GB. (4) Sistem Operasi Windows 8.1 64-bit.

3.2 Program Aplikasi

Aplikasi ini dibuat dengan bahasa pemrograman *Java* dengan *source code editor* NetBeans IDE 8.1 dan jdk16.

3.3 Hasil Pengujian Waktu Enkripsi dan Dekripsi

Tabel 1. Hasil Pengujian Waktu Enkripsi dan Dekripsi Teks

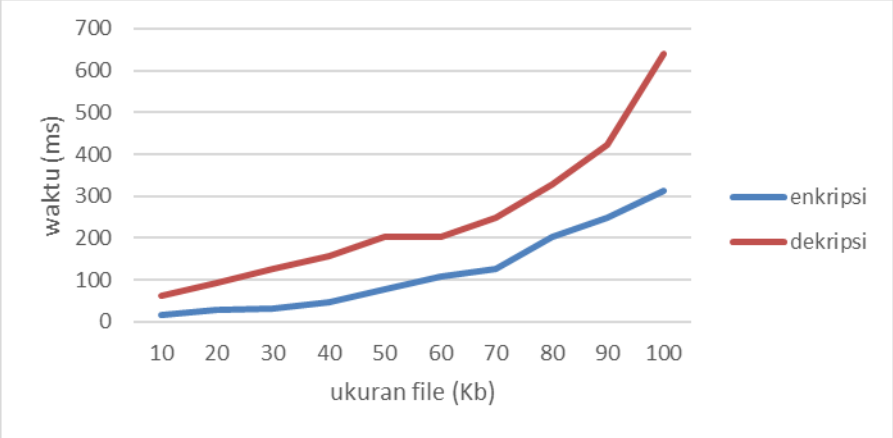
Ukuran (Kb)	***Enkripsi***	***Dekripsi***
10	16	62
20	27	94
30	31	125
40	47	156
50	78	203
60	109	203
70	125	250
80	203	328
90	250	422
100	313	640
Rata-rata	**119.9**	**248.3**

Dari Tabel 1, diperoleh rata-rata waktu enkripsi untuk file dengan ukuran 10 Kb hingga 100 Kb adalah 119.9 *ms* dan rata-rata waktu dekripsi adalah 248.3 *ms*. Dapat diketahui bahwa rata-rata waktu dekripsi lebih lama dibandingkan waktu enkripsi hal ini dikarenakan proses dekripsi membutuhkan nilai n' sebagai *public key* yang diperoleh dari proses pembangkitan kunci. Sehingga waktu komputasinya sedikit lebih lama dibandingkan waktu enkripsi yang hanya menggunakan *private key*.

Gambar 4 menunjukkan bentuk visualisasi waktu enkripsi dan dekripsi pada *file* dokumen. Grafik tersebut menunjukkan bahwa waktu komputasi berbanding lurus dengan ukuran *file*. Semakin besar ukuran *file* maka waktu enkripsi dan dekripsi akan semakin lama.

Dari Tabel 2, diperoleh rata-rata waktu enkripsi untuk file citra dengan ukuran 100x100 piksel hingga 1000x1000 piksel adalah 2623.4 *ms* dan rata-rata waktu dekripsi adalah 4349.5 *ms*.

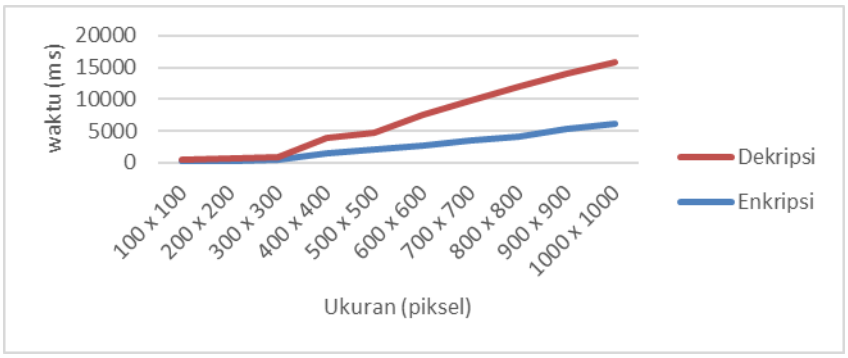
Gambar 5 menunjukkan bentuk visualisasi waktu enkripsi dan dekripsi pada *file* citra. Grafik tersebut menunjukkan bahwa waktu komputasi berbanding lurus dengan ukuran piksel. Semakin besar ukuran piksel maka waktu enkripsi dan dekripsi akan semakin lama.



Gambar 4. Grafik Pengujian Waktu Enkripsi dan Dekripsi *File* Dokumen

Tabel 2. Hasil Pengujian Waktu Enkripsi dan Dekripsi Citra

Ukuran (Piksel)	***Enkripsi***	***Dekripsi***
100 x 100	203	265
200 x 200	266	281
300 x 300	344	422
400 x 400	1516	2422
500 x 500	2031	2703
600 x 600	2719	4861
700 x 700	3562	6143
800 x 800	4125	7894
900 x 900	5359	8704
1000 x 1000	6109	9800
Rata-rata	**2623.4**	**4349.5**



Gambar 5. Grafik Pengujian Waktu Enkripsi dan Dekripsi *File* Citra

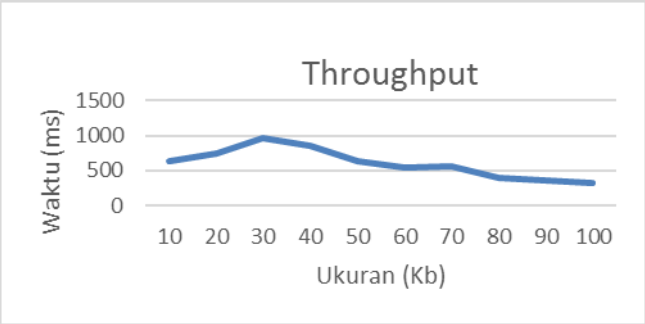
3.4 Hasil Pengujian *Throughput*

Pengujian *throughput* dilakukan dengan menghitung ukuran pesan sebelum dienkripsi dan membandingkannya dengan waktu enkripsi yang dibutuhkan dan dinyatakan dalam satuan *Kilobyte per second* (*Kbps*).

Tabel 3. Hasil Pengujian *Throughput*

Ukuran (Kb)	***Enkripsi***	***Dekripsi***
10	16	625
20	27	740.5319149
30	31	967.7419355
40	47	851.0638298
50	78	641.025641
60	109	550.4587156
70	125	560
80	203	394.08867
90	250	360
100	313	319.4888179
Rata-rata	**119.9**	**600.960835**

Berdasarkan Tabel 3, diperoleh rata-rata *throughput* untuk *file* dokumen berukuran 10 *Kb* hingga 100 *Kb* adalah 600.96 *Kbps*.

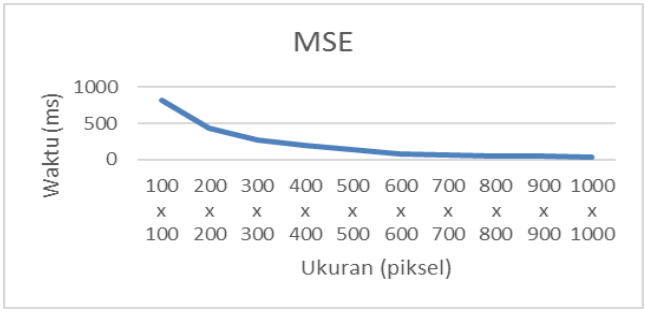


Gambar 6. Grafik Pengujian *Throughput*

Gambar 6 menunjukkan bentuk visualisasi *throughput*. Grafik tersebut menunjukkan *file* dokumen berukuran 30 *Kb* memiliki nilai *throughput* paling tinggi. Hal ini dikarenakan perhitungan *throughput* menggunakan perbandingan antara ukuran *file* dan waktu enkripsi, sehingga semakin kecil selisih antara ukuran *file* dan waktu enkripsi, maka nilai *throughput* akan semakin besar.

3.5 Hasil Pengujian *Mean Square Error (MSE)*

Pengujian nilai MSE dilakukan dengan cara menghitung nilai *error* kuadrat rata-rata antara citra asli dan citra hasil dekripsi. Berdasarkan Tabel 4, diperoleh diperoleh nilai rata-rata seluruh *MSE* adalah 213.9529047. Nilai ini diperoleh dari nilai rata-rata tiap intensitas piksel (R, G dan B) berdasarkan ukuran piksel yang berbeda dengan rata-rata penurunan sebesar 27.67%.



Gambar 7. Grafik Pengujian *MSE* pada Citra

Gambar 7 menunjukkan bentuk visualisasi rata-rata nilai *MSE* pada citra dengan berbagai ukuran. Grafik tersebut menunjukkan bahwa semakin besar dimensi citra maka nilai *MSE* akan semakin kecil.

Tabel 4. Hasil Pengujian *MSE*

Ukuran (Piksel)	*MSE*	Presentase penurunan
100 x 100	625	0.0000000
200 x 200	740.5319149	46.8784116
300 x 300	967.7419355	36.6853569
400 x 400	851.0638298	30.2892849
500 x 500	641.025641	25.5062709
600 x 600	550.4587156	42.7353137
700 x 700	560	18.5116830
800 x 800	394.08867	17.5156165
900 x 900	360	15.7873841
1000 x 1000	319.4888179	15.1927020
Rata-rata	**600.960835**	**27.6780026**

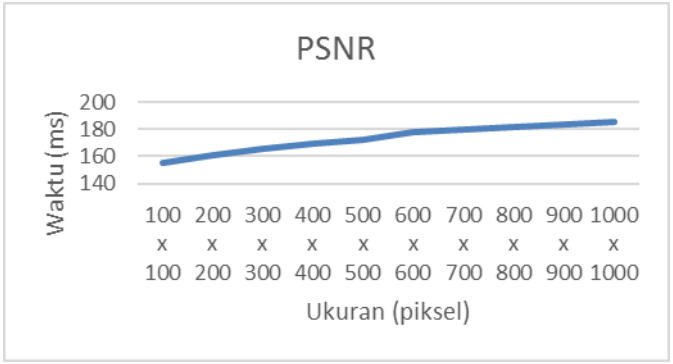
3.6 Hasil Pengujian *Peak Signal to Noise Ratio (PSNR)*

Pengujian nilai *PSNR* dilakukan untuk mengetahui perbandingan kualitas citra asli dan citra setelah didekripsi. Nilai *PSNR* ini diperoleh setelah mendapatkan nilai *MSE*.

Tabel 5. Hasil Pengujian PSNR

Ukuran (Piksel)	*PSNR*	Presentase Kenaikan
100 x 100	155.1563704	0,0000000
200 x 200	161.3642179	3.847102924
300 x 300	165.8226918	2.688699467
400 x 400	169.3955365	2.109172909
500 x 500	172.3687630	1.724921851
600 x 600	177.9009810	3.109717514
700 x 700	179.9566493	1.142313052
800 x 800	181.8947848	1.06552564
900 x 900	183.6229178	0.941131426
1000 x 1000	185.2847099	0.896885735
Rata-rata	**173.2767622**	**1.9472745**

Berdasarkan Tabel 5, diperoleh diperoleh nilai rata- *PSNR* adalah 173.2767622. Nilai ini diperoleh dari nilai rata-rata tiap intensitas piksel (R. G dan B) berdasarkan ukuran piksel yang berbeda dengan rata-rata kenaikan sebesar 1.94%.



Gambar 8. Grafik Pengujian *PSNR* pada Citra

Gambar 8 menunjukkan bentuk visualisasi rata-rata nilai *PSNR* pada citra dengan berbagai ukuran. Grafik tersebut menunjukkan bahwa semakin besar dimensi citra maka nilai PSNR akan semakin besar.

3.7 Hasil Pengujian *Avalanche Effect*

Pengujian *avalanche effect* diperoleh dari perbandingan bit antara *plaintext* dengan *ciphertext*, yang kemudian akan dibagi dengan jumlah seluruh bit pada *ciphertext*. Pada *file* teks diperoleh nilai *avalanche effect* sebesar 85.18%. Sementara itu, untuk *file* citra, diperoleh nilai *avalanche effect* sebesar 84.46%.

4. Kesimpulan

Setelah melakukan beberapa uji coba terhadap sistem aplikasi ini, didapatkan beberapa kesimpulan sebagai berikut: (1) Algoritma *Diffie-Hellman* dan *ElGamal* dapat dikombinasikan untuk enkripsi dan dekripsi data dengan melalui 4 proses, yaitu proses pertukaran kunci, pembangkitan kunci, enkripsi, dan dekripsi. (2) Dari hasil pengujian, diperoleh rata-rata waktu enkripsi *file* teks sebesar 119.9 *ms*, dekripsi 248.3 *ms*, dan *throughput* 600.96 *Kbps*. (3) Rata-rata waktu enkripsi *file* citra sebesar 2623.4 *ms* dan waktu dekripsi 4349.5 *ms*. (4) Dari hasil pengujian pada beberapa dimensi citra yang berbeda, rata-rata *MSE* pada citra sebesar 213.9 dan *PSNR* sebesar 173.27 *dB*. Nilai ini menunjukkan kualitas citra cukup baik (diatas 40 dB). (5) Semakin besar dimensi citra, maka nilai *MSE* tiap intensitas piksel akan semakin kecil, dengan rata-rata penurunan sebesar 27.67%. (6) Semakin besar dimensi citra, maka nilai PSNR akan semakin besar, dengan rata-rata kenaikan sebesar 1.94%. (7) Kombinasi algoritma *Diffie-Hellman* dan *ElGamal* menghasilkan nilai *avalanche effect* sebesar 85.18% untuk *file* teks dan 84.46% untuk *file* citra. Nilai ini menunjukkan kombinasi algoritma cukup baik dengan hasil pergeseran bit diatas 60%.

Referensi

[1] Shetty, Annapoorna., Shetty, Shravya., Krithika. 2014. A Review on Asymmetric Cryptography RSA and ElGamal. India: IJIRCCE.

[2] ElGamal, Taher. 1985. A Public Key Cryptosystem and A Signature Scheme based on Discrete Logarithm. Palo Alto: Springer.

[3] Widarma, Adi. 2016. Kombinasi Algoritma AES, RC4 dan ElGamal dalam Skema Hybrid untuk Keamanan Data. Medan: Universitas Negeri Medan.

[4] Massandy, Danang Tri. 2009. Algoritma ElGamal dalam Pengamanan Pesan Rahasia. Bandung: Sekolah Teknik Elektro dan Informatika.

[5] Kumar, Manish., Iqbal, Akhlaq., Kumar, Pranjal. 2016. A New RGB Image Encryption based on DNA Encoding and Elliptic Curve Diffie-Hellman Cryptography. India: Elsevier.

[6] Cheddad, Abbas., Condell, Joan., Curran, Kevin., Kevitt, Paul Mc. 2009. Digital Image Steganography: Survey and Analysis of Current Methods. UK: Elsevier.

[7] Sulaksono, Danang Haryo. 2016. Multiple Encrytion Menggunakan Metode Vigenere Chiper dan Blowfish. Surabaya: Institut Teknologi Adhi Tama Surabaya.