PENGAMANAN SQLITE DATABASE MENGGUNAKAN KRIPTOGRAFI ELGAMAL

Deny Adhar

Teknik Informatika, STMIK Potensi Utama Medan
Jln. Kol. Yos. Sudarso Km. 6,5 No. 3A Medan
adhar_7@yahoo.com

Abstrak

SQLite database merupakan sebuah sistem manajemen basis data relasional yang bersifat ACID-compliant dan memiliki ukuran pustaka kode yang relatif kecil, ditulis dalam bahasa C. Namun ada satu hal yang masih menjadi kekurangan di dalam database SQLite yaitu masalah keamanan data karena database tersebut masih bersifat plain, dalam arti tidak terlindungi oleh enkripsi. Database SQLite tidak memiliki mekanisme untuk memproteksi data seperti password atau enkripsi. Siapapun yang memiliki akses ke fisik file database, maka yang bersangkutan akan bisa membukanya dan melihat isi data di dalamnya. Untuk melindungi SQLite database tersebut dibutuhkan teknik kriptografi. Algoritma Elgamal merupakan salah satu dari algoritma yang digunakan untuk menyelesaikan permasalahan pada bidang kriptografi. Algoritma Elgamal dipilih dalam teknik kriptografi ini karena algoritma Elgamal dalam mengamankan pesan rahasia membutuhkan pembentukan kunci dengan menggunakan bilangan prima dan pemecahan masalahnya menggunakan logaritma diskrit yang cukup menyulitkan. Kunci yang dimiliki algoritma elgamal ada dua jenis yaitu kunci public dan kunci private. Kunci publik untuk umum dan kunci private untuk diri sendiri. Dengan algoritma elgamal diharapkan akan terciptanya sebuah sistem yang optimal untuk mengamankan SQLite database.

Kata Kunci : *Cryptographic, Encryption, Decryption, Algorithm Elgamal, SQLite database*

1. PENDAHULUAN

Hampir semua aplikasi - aplikasi sistem informasi menggunakan *database* untuk menyimpan informasi. Suatu sistem *database* yang memiliki informasi yang penting, sangat memerlukan suatu *system* keamanan untuk melindungi *database* dari orang yang tidak mempunyai wewenang otoritas dari data tersebut. Keamanan *database* adalah suatu cara untuk melindungi *database* dari ancaman, baik dalam bentuk kesengajaan atau pun bukan. Ancaman adalah segala situasi atau kejadian baik secara sengaja maupun tidak yang bersifat merugikan dan mempengaruhi *system* serta secara konsekuensi terhadap perusahaan/organisasi yang memiliki *system database*. Hal-hal yang berkaitan dengan pengamanan data-data penting tersebut haruslah benar-benar diperhatikan agar data yang akan tersimpan dalam komputer kita tetap aman dari orang-orang yang tidak bertanggung jawab.
Salah satu aspek keamanan pada *database* adalah melakukan autentifikasi pada *user* yang berhak mengolah data pada *database*, umumnya *database* yang digunakan pada aplikasi seperti *mySQL* maupun *SQL* sudah memiliki menu enkripsi pada struktur databasenya. Namun di dalam *SQLite database* tersebut masih bersifat *plain*, dalam arti tidak terlindungi oleh enkripsi. *SQLite database* tidak memiliki mekanisme untuk memproteksi data atau enkripsi. Siapapun yang memiliki akses ke fisik *file database*, maka yang bersangkutan akan bisa membukanya dan melihat isi data di dalamnya. Sehingga dibutuhkan suatu algoritma kriptografi yang dapat mengenkripsi data *administrator* pada *SQLite database*. [1], [2, 1].

Kriptografi adalah suatu ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi, *integritas* suatu data, serta autentifikasi data. Algoritma kriptografi yang digunakan untuk menyelesaikan masalah diatas adalah dengan menggunakan kriptografi *Elgamal*

2. LANDASAN TEORI

2.1 Ancaman Keamanan

Begitu banyak terjadi pertukaran informasi setiap detiknya di internet. juga banyak terjadi pencurian atas informasi oleh pihak - pihak yang tidak bertanggung jawab. Ancaman keamanan yang terjadi terhadap informasi adalah :

1. *Interruption*
   Interruption merupakan suatu bentuk ancaman terhapap *availability*, di mana suatu data dirusak sehingga tidak dapat digunakan lagi. Tindakan perusakan yang dilakukan dapat berupa perusakan fisik maupun non fisik. Perusakan fisik umumnya berupa perusakan *harddisk* dan media penyimpanan

lainnya serta pemotongan kabel jaringan. Sedangkan perusakan *non* fisik berupa penghapusan suatu *file-file* tertentu dari sistem komputer.

2. *Interception*
   Interception merupakan suatu bentuk ancaman terhadap *secrecy*, di mana pihak yang tidak berhak berhasil mendapat hak akses untuk membaca suatu data/ informasi dari suatu sistem komputer. Tindakan yang biasa dilakukan biasanya melalui penyadapan data yang ditransmisikan lewat jalur *public*/ umum. Tindakan seperti ini biasa dikenal dengan istilah *wiretapping* dalam *wired networking* (jaringan yang menggunakan kabel sebagai media transmisi data).
3. *Modification*
   Modification merupakan suatu bentuk ancaman terhadap *integrity*, di mana pihak yang tidak berhak berhasil mendapat hak akses untuk mengubah suatu data/ informasi dari suatu sistem komputer. Biasanya data/ informasi yang diubah adalah *record* dari suatu tabel pada *file database*.
4. *Fabrication*
   Fabrication juga merupakan suatu bentuk ancaman terhadap *integrity*. Tindakan yang biasa dilakukan adalah dengan meniru dan memasukkan suatu objek ke dalam sistem komputer. Objek yang dimasukkan bisa berupa suatu *file* maupun suatu *record* yang disisipkan pada suatu *program* aplikasi. [2], [2, 8-9].

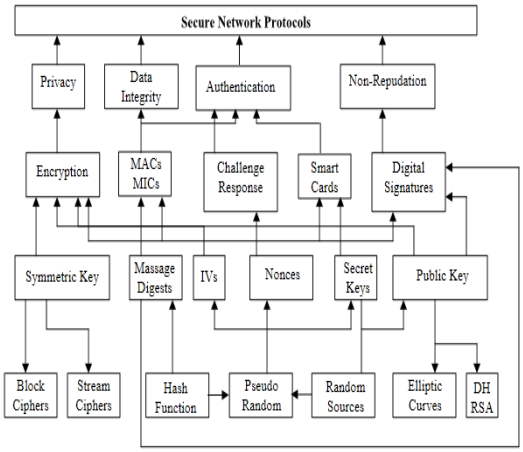
Tabel 1. Ancaman Terhadap Keamanan

System	Avability	Secrecy	Integrity
Hardware	Dicuri atau dirusak		
Software	Program dihapus	software dicopy	Program dimodifikasi
Data	File dihapus atau dirusak	Dicuri, disadap	File dimodifikasi
Line komunikasi	Kabel diputus	Informasi disadap	Informasi dimodifikasi

2.2. Algoritma Kriptografi Modern

Enkripsi modern berbeda dengan enkripsi konvensional. Enkripsi modern sudah menggunakan komputer untuk pengoperasiannya, berfungsi untuk mengamankan data baik di transfer melalui jaringan komputer maupun yang bukan.

Hal ini sangat berguna untuk melindungi privacy, data integrity, authentication dan non-repudiation. dibawah ini akan digambarkan bagaimana enkripsi modern saling mendukung satu dengan yang lain. [2], [2, 45].



Gambar 1. Skema Kriptografi Modern

2.3. Algoritma Elgamal

Algoritma Elgamal diciptakan oleh Taher Elgamal pada tahun 1984. Algoritma ini pada mulanya digunakan untuk kepentingan digital signature, namun kemudian dimodifikasi sehingga algoritma Elgamal bisa digunakan untuk enkripsi dan dekripsi. Elgamal digunakan di dalam perangkat lunak security yang dikembangkan oleh GNU, program PGP dan pada sistem security lainnya. Keamanan algoritma ini terletak pada sulitnya menghitung logaritma diskrit.

Alogaritma ini disebut alogaritma diskrit karena nilainya berhingga dan bergantung pada bilangan prima yang digunakan. Karena bilangan prima yang digunakan adalah bilangan prima, maka sangat sulit bahkan tidak mungkin menurunkan kunci private dari kunci public yang diketahui walaupun serangan dilakukan dengan menggunakan sumberdaya komputer yang sangat besar. [3], [2, 1]

Algoritma Elgamal memerlukan sepasang kunci yang dibangkitkan dengan memilih bilangan prima *p* dan dua buah bilangan acak (*random*) *g* dan *x*, dengan syarat bahwa nilai *g* dan *x* lebih kecil dari *p* yang memenuhi persamaan.

$y = gx \bmod p$ ……(1)

Dari persamaan tersebut nilai *y*, *g* dan *p* merupakan pasangan kunci *public* sedangkan *x*, *p* merupakan pasangan kunci pribadi. Besaran-besaran yang digunakan dalam algoritma kriptografi Elgamal adalah:

1. Bilangan prima *p* bersifat tidak rahasia.
2. Bilangan acak *g* ($g < p$) bersifat tidak rahasia
3. Bilangan acak *x* ($x < p$) bersifat rahasia.
4. Bilangan *y* bersifat tidak rahasia.
5. *m* (*plaintext*) bersifat rahasia merupakan pesan asli yang digunakan untuk data
6. Sumber dalam proses enkripsi dan merupakan data hasil pada proses dekripsi.
7. *a* dan *b* (*ciphertext*) bersifat tidak rahasia

[4], [2, 1-2]

Proses pertama adalah pembentukan kunci yang terdiri dari kunci rahasia dan kunci *public*.

Pada proses ini dibutuhkan sebuah bilangan prima *p* yang digunakan untuk membentuk grup *Z * p* elemen *primitif* dan sebarang *a* 0,1,..., *p* 2.
Kunci publik *algoritma Elgamal* berupa pasangan 3 bilangan, yaitu (*p*, ,) , dengan :

$$y = g^x \bmod p \qquad (1)$$

Karena pada algoritma *Elgamal* menggunakan bilangan bulat dalam proses perhitungannya, maka pesan harus dikonversi ke dalam suatu bilangan bulat. Untuk mengubah pesan menjadi bilangan bulat, digunakan kode ASCII (*American Standard for Information Interchange*). Kode ASCII merupakan representasi numerik dari karakter-karakter yang digunakan pada komputer, serta mempunyai nilai minimal 0 dan maksimal 255. Oleh karena itu, berdasarkan sistem kriptografi *Elgamal* di atas maka harus digunakan bilangan prima yang lebih besar dari 255. Kode ASCII berkorespondensi 1-1 dengan karakter pesan. Pihak yang membuat kunci public dan kunci rahasia adalah penerima, sedangkan pihak pengirim hanya mengetahui kunci public yang diberikan oleh penerima, dan kunci public tersebut digunakan untuk mengenkripsi pesan. Jadi, kentungan menggunakan algoritma kriptografi *Elgamal* adalah tidak ada permasalahan pada distribusi kunci apabila jumlah pengirim sangat banyak serta tidak ada kepastian keamanan jalur yang digunakan. [5], [2, 3-4]

2.4. Enkripsi Dan Dekripsi Menggunakan Algoritma Elgamal

Besaran yang digunakan didalam algoritma *Elgamal*

1. Bilangan prima, *p* (tidak rahasia)
2. Bilangan acak, *g* (*g* < *p*) (tidak rahasia)
3. Bilangan acak, *x* (*x* < *p*) (rahasia)
4. M (*plaintext*) (rahasia)
5. a dan b (*ciphertext*) (tidak rahasia)

Prosedur Membuat Pasangan Kunci

1. Pilih sembarang bilangan prima *p*.
2. Pilih dua buah bilangan acak, *g* dan *x*, dengan syarat $g < p$ dan $1 \le x \le p - 2$.
3. Hitung $y = g^x \bmod p$.

Kunci publik adalah *y*, kunci rahasia adalah *x*. Nilai *g* dan *p* tidak dirahasiakan dan dapat diumumkan kepada anggota kelompok, *plaintext* disusun menjadi blok-blok m1, m2, …, sedemikian sehingga setiap blok merepresentasikan nilai di dalam rentang 0 sampai $p - 1$.

Pilih bilangan acak *k*, yang dalam hal ini $k \le p - 1$, sedemikian sehingga *k* relatif prima dengan $p - 1$.

Setiap blok m dienkripsi dengan rumus :

$a = g^k \bmod p$
$b = y^k .m \bmod p$

Pasangan a dan b adalah *ciphertext* untuk blok pesan m. Jadi, ukuran ciphertext dua kali ukuran *plaintext*-nya. Untuk mendekripsi a dan b digunakan kunci rahasia, x, dan *plaintext* m diperoleh kembali dengan persamaan

$n = a^{p-1-x} \bmod p$

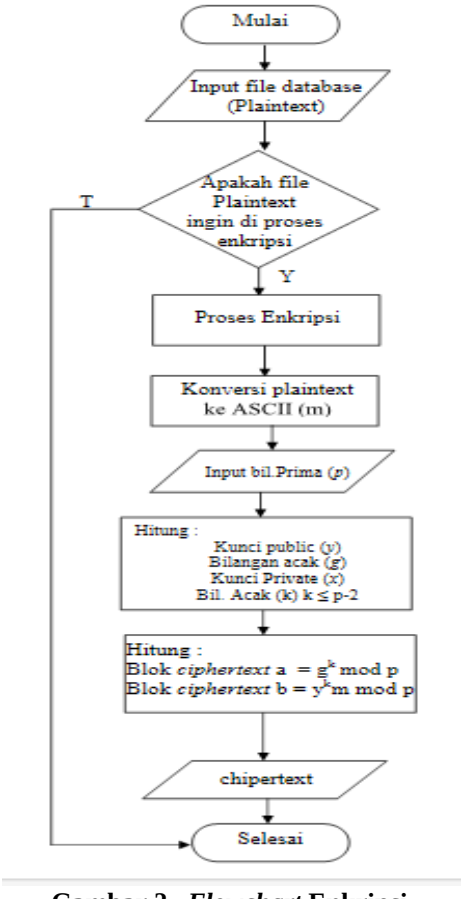
dan

$m = b \times n \bmod p$

yang berarti bahwa *plaintext* dapat ditemukan kembali dari pasangan *ciphertext* a dan b. [6], [2, 5]

3. Proses Enkripsi

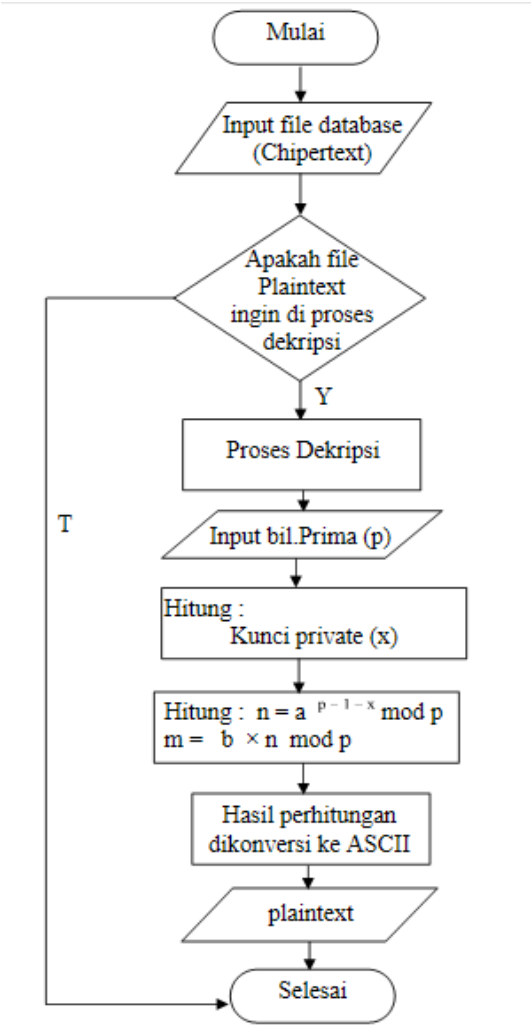
Enkripsi merupakan suatu langkah untuk mengolah data awal (*plaintext*) menjadi sebuah data acak (*ciphertext*) yang tidak dapat diterjemahkan secara langsung. Proses kerja enkripsi kriptografi *Elgamal* dapat digambarkan seperti *flowchart* pada gambar 2.



Gambar 2. *Flowchart* Enkripsi

Proses Dekripsi

Proses *dekripsi* merupakan sistem untuk mengolah data acak (*ciphertext*) menjadi data awal (*plaintext*). Dalam proses dekripsi ini terdapat proses dekripsi kriptografi *Elgamal*. Secara umum proses kerja dekripsi dapat digambarkan seperti gambar *flowchart* 3.2



Gambar 3. *Flowchart* Dekripsi

Contoh

Siti ingin membangkitkan pasangan kuncinya. Siti memilih p = 2357, g = 2, dan x = 1751. Kemudian menghitung :

$y = g^x \bmod p = 21751 \bmod 2357 = 1185$

Jadi kunci publiknya (y = 1185, g = 2, p = 2357) dan kunci privatnya (x = 1751, p = 2357).

Enkripsi

Misalkan Ahmad ingin mengirim palinteks m = 2035 (nilai m masih berada di dalam selang [0, 2357 – 1]). Ahmad memilih bilangan acak k = 1520 (nilai k masih berada di dalam selang [0, 2357 – 1]). Kemudian Ahmad menghitung

$a = g^k \bmod p = 21520 \bmod 2357 = 1430$

$b = y^k m \bmod p = 11851520 \times 2035 \bmod 2357 = 697$

Jadi, cipherteks yang dihasilkan adalah (1430, 697). Ahmad mengirim cipherteks ini ke Siti.

Dekripsi

Siti mendeskripsi cipherteks dari Ahmad dengan melakukan perhitungan sebagai berikut :

$n = a^{p-1-x} \bmod p = 1430605 \bmod 2357 = 872$

$m = b \times n \bmod p = 697 \times 872 \bmod 2357 = 2035$

Plainteks yang didekripsi, 2035, sama dengan plainteks yang dikirim oleh Ahmad.

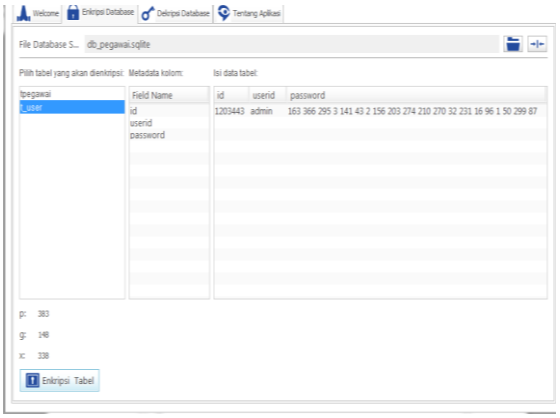
4. PENGUJIAN

Pengujian perangkat lunak ini dilakukan pada komputer dengan CPU ber-processor Intel Pentium core i3 1,4GHz, RAM 2048 MB dengan sistem operasi Windows Seven Ultimed.

Berikut ini adalah beberapa pengujian yang dilakukan pada program enkripsi dan dekripsi SQLite database dengan algoritma Elgamal ini. Pengujian yang ditampilkan adalah pengujian terhadap proses enkripsi, proses dekripsi serta lamanya waktu yang diperlukan dalam melakukan proses enkripsi dan dekripsi tersebut.

1. Proses Enkripsi

Pada proses pengujian enkripsi *SQLite database* dimulai dari menginputkan *file SQLite database* yang akan dienkripsi pada sistem aplikasi pengamanan database SQLite database kemudian hasil proses enkripsi dapat dilihat pada gambar 4 dibawah ini yang merupakan hasil enkripsi berupa *chipertext*

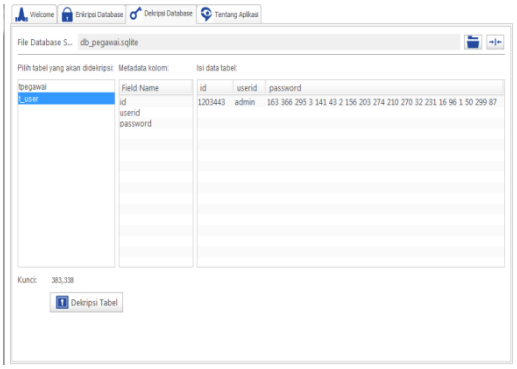


Gambar 4. Hasil Proses Enkripsi

Proses enkripsi diatas menggunakan kunci *public* yaitu : p = 383, g = 148, y = 295 dimana y didapatkan dari persamaan $y = g^x \bmod p = 148^{338} \bmod 383 = 295$ *output* hasil kunci *public* dan kunci *private*

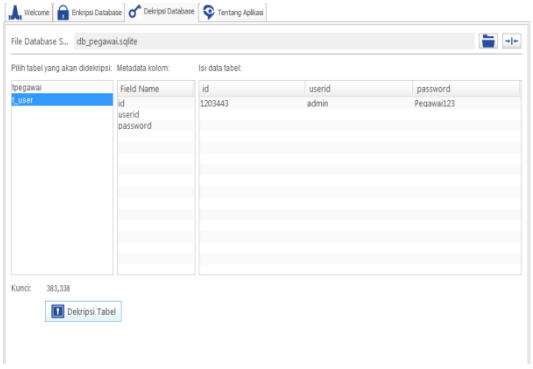
2. Proses Dekripsi

Pada proses pengujian dekripsi *SQLite database* dimulai dari menginputkan file *SQLite database* yang telah di enkripsi.



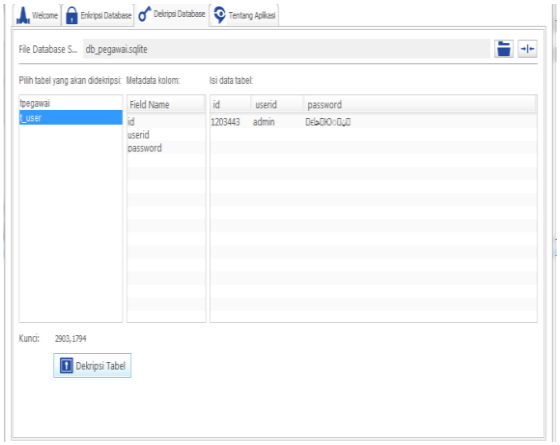
Gambar 5. Tampilan Pesan Chipertext

Hasil proses dekripsi dapat dilihat pada gambar 6 dibawah ini



Gambar 6. Hasil Proses Dekripsi

Jika kita menggunakan kunci *private* dan bilangan prima yang berbeda pada saat mengenkripsi *plaintetxt* menjadi *ciphertext* maka *ciphertext* pada saat didekripsikan tidak akan kembali lagi menjadi pesan aslinya (*plaintext*), seperti terlihat pada gambar 7.



Gambar 7. Hasil Proses Dekripsi Yang Tidak Berhasil

4.1 Pengujian terhadap waktu proses

Berdasarkan beberapa proses pengujian dengan menggunakan beberapa file dengan ukuran yang berbeda-beda didapatkan waktu proses secara keseluruhan seperti pada tabel 2.

Tabel 2. Pengujian ukuran terhadap waktu proses

NO	UKURAN FILE DATABASE	WAKTU (DETIK)	
		Enkripsi	Dekripsi
1	10kb	1,5	1,6
2	20kb	3,2	3,3
3	40kb	6,5	6,6
4	60kb	10	11
5	90kb	15	15,5
6	120kb	20	21

5. KESIMPULAN

Pada bab ini digunakan untuk memberikan kesimpulan dan saran dari hasil penelitian pengamanan SQLite database menggunakan kriptografi Elgamal. Beberapa hal yang dapat disimpulkan dari penelitian ini adalah sebagai berikut:

1. Sistem kriptografi Elgamal terdiri atas 2 proses utama yaitu proses enkripsi dan proses dekripsi. Untuk proses enkripsi dibutuhkan kunci public dan kunci private dan untuk proses dekripsi dibutuhkan bilangan prima dan kunci private sehingga dapat lebih optimal dalam mengamankan data.
2. Kriptografi Elgamal menggunakan konsep logaritma diskrit, bilangan prima dan bilangan acak untuk proses enkripsi dan dekripsi. Sehingga menyebabkan algoritma ini memiliki tingkat perhitungan matematik yang sulit. Apabila algoritma kriptografi dalam perhitungan matematiknya semakin sulit maka semakin aman algoritma tersebut untuk digunakan.
3. Kesimpulan untuk proses penerapan algoritma ini dikatakan aman dengan melakukan proses enkripsi dengan dua buah kunci sehingga algoritma ini standart untuk digunakan untuk enkripsi record tabel SQLite database.
4. Waktu untuk proses enkripsi dan dekripsi berbanding lurus dengan penambahan ukuran file. Nilai rata-rata kecepatan proses yang dihasilkan sebesar 6 KB/detik.

Daftar Pustaka

[1] Didik Prasetyo, 2006. *Keamanan SQLite Database pada RDBMS*, Jakarta, PT. Elex Media Komputindo.

[2] Dony Ariyus 2008, *Pengantar Ilmu kriptografi Teori Analisa dan Implementasi*, Yogyakarta, ANDI.

[3] Anandia Zelvina, Syahril Efendi, Dedy Arisandi, 2012. *Journal Perancangan Aplikasi Pembelajaran Kriptografi Kunci Public Elgamal Untuk Mahasiswa, Medan*.

[4] M. Taufiq Tamam, Wakhyu Dwiono, Tri Hartono, 2010. *Journal Penerepan Algoritma Kriptografi Elgamal Untuk Pengaman File Citra,* Yogyakarta.

[5] Danang Tri Massandy, 2009. *Journal Algoritma Elgamal Dalam Pengaman Pesan Rahasia*, Bandung.

[6] Mukhammad Ifanto, 2009. Journal Metode Enkripsi Dan Dekripsi Menggunakan Algoritma Elgamal