REPOSITOR, Vol. 2, No. 10, Oktober 2020, Pp. 1406-1415
ISSN : 2714-7975
E-ISSN : 2716-1382

Analisa Efisiensi Algoritma Hybrid El Gamal dan Short Range Natural Number pada Keamanan Pesan Berbasis Socket TCP

Aminudin*[1], Pitra Nur Dwijayanto Limbong[2], Sofyan Arifianto[3]
Program Studi Informatika, Universitas Muhammadiyah Malang
aminudin2008@gmail.com[1], limbongpitra2@gmail.com[2], sofyanarifianto37@gmail.com[3]

Abstrak

Perkembangan teknologi yang pesat dapat berdampak pada keamanan digital. Pengiriman pesan singkat merupakan salah satu pengimplementasian pada era digital. Pengirman pesan sangat berdampak pada keamanannya. Kriptografi dibutuhkan dalam proses pengamanan pesan. Penelitian ini menguji algoritma Hybrid (El Gamal SRNN) yang merupakan hasil improvisasi dari penelitian sebelumnya. Algoritma Hybrid akan dianalisa efisiensinya dalam segi performa dan keamanannya. Efisiensi performa meliputi tiga hal yaitu waktu proses pembangkitan kunci, waktu proses enkripsi dan waktu proses dekripsi. Efisiensi keamanan akan diuji menggunakan serangan kombinasi baby step-giant step dan factorization attack. Algoritma Hybrid dirancang menggunakan inputan bilangan prima yang akan menghasilkan kunci public dan privat tambahan hasil dari bilangan acak variable u dan a yang akan menghasilkan masing-masing kunci publik (y, g, pEL, n, e, u^a) dan kunci privat (x, pEL, d, a, u). Penambahan variable kunci privat dan public ini dimaksudkan agar keamanan dari algoritma ini lebih baik dan sulit untuk dipecahkan. Performa algoritma Hybrid yang diajukan lebih buruk dari segi pembangkitan kunci, waktu enkripsi dan dekripsi yang mana waktu yang diperlukan masing-masing adalah 1.429, 1.407, 1.516 lebih lambat dari algoritma Hybrid (El Gamal RSA). Pengujian keamanan menunjukkan bahwa algoritma Hybrid (El Gamal SRNN) yang diajukan lebih baik ketimbang algoritma Hybrid (El Gamal RSA) dilihat dari waktu eksekusi yang masih belum ditemukan variable u dan a belum bisa dipecahkan.

Kata kunci: *bilangan prima, kunci public, kunci privat, algoritma hybrid*

Abstract

The rapid development of technology can have an impact on digital security. Text messaging is one of the implementations in the digital age. Messaging greatly impacts its security as well. Cryptography is needed in the process of securing messages. This study tested the Hybrid algorithm (El Gamal SRNN) which was the result of improvisation from previous research. Hybrid algorithm will be analyzed for efficiency in terms of performance and safety. Performance efficiency includes three things: key generation process time, encryption process time and decryption process time. Safety efficiency will be tested using a combination of baby step-giant step attacks and factorization attacks. The Hybrid algorithm is designed using prime number input which will generate additional public and private keys resulting from random numbers variable u and a which will produce each public key (y, g, pEL, n, e, u^a) and private key (x, pEL, d, a, u). The addition of private and public key variables is intended so that the security of this algorithm is better and difficult to solve. The performance of the proposed Hybrid algorithm is worse in terms of key generation, encryption and decryption time where the time required is 1,429, 1,407, 1,516, slower than Hybrid algorithm (El Gamal RSA). Security testing shows that the proposed Hybrid algorithm (El Gamal SRNN) is better than the Hybrid algorithm (El Gamal RSA) seen from the time of execution that the variables u and a have not been found yet.

Keywords*: prime number, public key, private key, hybrid algorithm*

1. Pendahuluan

Instant messaging menggunakan jaringan yang bersifat public dan rentan terhadap penyadapan data[1]. Masalah keamanan ini dapat diatasi menggunakan algortima kriptografi. Teknik kriptografi sangat lekat dalam proses pengiriman atau pertukaran data salah satunya pada

proses pengiriman pesan. Komunikasi yang biasa digunakan dalam proses pengiriman pesan menggunakan koneksi Socket TCP. Koneksi ini menggunakan system server dan client[11] yang mana sever akan bertugas sebagai penengah antara client dan client lain. Jaringan dengan koneksi ini rentan karena menggunakan koneksi public, atas dasar inilah diperlukan Teknik kriptografi sebagai pengaman data antara 2 pihak. Algoritma El Gamal dan RSA merupakan algoritma asimetrik (kunci public)[2][3]. Algoritma El Gamal bergantung pada logaritma diskrit[4][5] yang mana menggunakan satu buah kunci privat x. Algoritma RSA bergantung pada nilai faktorisasi prima[6][7] yang akan membentuk satu buah kunci privat d. Pada penjelasan diatas maka dapat disimpulkan bahwa algoritma El Gamal dan RSA masih bisa dipecahkan. Pada dasarnya penggunaan algoritma RSA dan El Gamal bergantung pada nilai prima yang di bangkitkan pada awalan pemrosesan yang mana akan menentukan hasil dari variable-variable yang lain. Semakin besar nilai prima p pada algoritma El Gamal[8] dan semakin besar nilai prima p dan q pada RSA maka semakin panjang bit hasil faktorisasi dan masing-masing kunci privat.

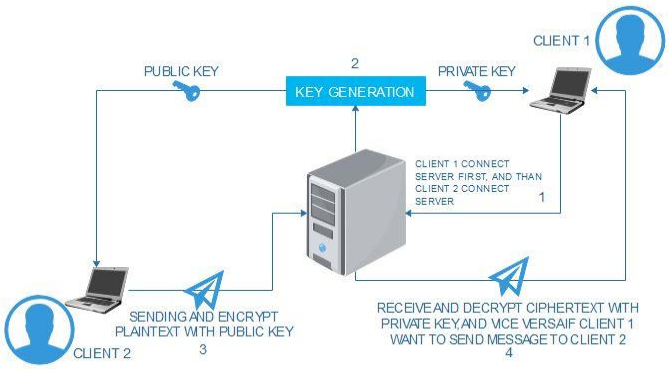
Algoritma Kriptografi yang telah dijelaskan diatas menunukkan bahwa penggunaan beberapa algoritma asimetrik masih bisa dipecahkan menggunakan beberapa serangan yang menggunakan input kunci public. Penggunaan algoritma El Gamal masih menimbulkan masalah yang terdapat pada kunci privat x yang dapat dipecahkan menggunakan proses penyerangan baby step-giant step[5][9]. Proses ini akan menyamakan nilai baby step dan giant step hasil dengan jumlah iterasi dari akar prima. Algoritma RSA pun masih memiliki kekurangan karena hasil nilai n yang mana menjadi kunci public masih dapat dipecahkan dengan menggunakan fermat factorization attack[5][10] untuk menemukan nilai kunci privat d. Adanya penggunaan improvisasi algoritma RSA diperlukan untuk menambahkan keamanan pada masing-masing kunci yang diperlukan dalam pembentukan proses pengamanan pesan.

Beberapa peneliti yang relevan telah meneliti untuk menambahkan ketahanan dari algoritma RSA seperti menggunakan 2 nilai modulus dengan menggunakan niliai prima p, q, r, s yang dilakukan oleh Balram Smawi dkk[12]. H.Sandityas melakukan penelitian menggunakan kombinasi algoritma El Gamal dan RSA[5]. Ruchita Patil melakukan penelitian dengan menggunakan algoritma improvisasi RSA (Short Range Natural Number)[13][14]. Penelitian ini bertujuan untuk mengembangkan algoritma Hybrid El Gamal RSA karena masih dapat dipecahkan dengan kombinasi serangan baby step-giant step dan factorization attack yang telah dilakukan oleh H.Sandityas dengan mengkombinasikan algoritma El Gamal dan Short Range Natural Number. Short Range Natural Number menggunakan penambahan variable u dan a pada proses pembangkitan kunci dan menambahkan kunci privat u^a serta menambahkan variabel u dan a sebagai kunci public. Algortitma ini akan diimplementasikan pada aplikasi pesan dengan Socket TCP. Perbandingan efisiensi akan diteliti dari segi waktu pembangkitan kunci, enkripsi dan dekripsi pada masing-masing algoritma Hybrid (El Gamal RSA) dan algoritma Hybrid (El Gamal SRNN). Perbandingan keamanan juga akan diuji menggunakan kombinasi serangan baby step-giant step dan factorization attack.

2. Metode Penelitian

2.1 Rancangan Instant Messaging

Algoritma Hybrid (El Gamal RSA) dan algoritma Hybrid (El Gamal SRNN) akan diterapkan pada instant messaging.



Gambar 1. Skema server dan client TCP

Gambar 1 menunjukkan bahwa client akan menuliskan pesan ke penerima dan sebelum dikirimkan akan melewati proses penyandian. Penerima pesan harus mempunyai kunci privat untuk membuka pesan sedangkan server hanya mengirim kunci public dan pesan saja. Kunci publik hanya bisa didapatkan oleh semua yang ada didalam jaringan pengiriman pesan atau local. Kunci privat hanya dipunyai oleh pengirim dan penerima pesan. Server selaku penengah hnya akan menyampaikan pesan ke client penerima pesan.

2.2 Algoritma Hybrid (El Gamal RSA)

Algoritma Hybrid ini menggunakan penggabungan antara algoritma El Gamal dan RSA. Proses penyandian akan melalui algoritma El Gamal lalu hasil penyandian algoritma El Gamal akan di enkripsi lagi menggunakan algoritma RSA. Proses penyandian pesan ini terdapat tiga proses yaitu generate key, enkiripsi dan dekripsi. Algoritma Hybrid ini membutukan tiga nilai prima yaitu nilai prima p untuk El Gamal dan prima p dan q untuk algoritma RSA. Algoritma ini akan menghasilkan kunci public (y, g, pEL, n, e) dan kunci privat (x, pEL, d, n).

Algoritma 1. Pembangkitan Kunci Hybrid El Gamal RSA

Input : Panjang bit
Output : kunci publik (y, g, pEL, n, e) dan kunci privat (x, pEL,d)

1. Generate pEL
2. Generate g dan x
3. Generate p dan q
4. $y = g^x mod\ pEL$
5. $n = p.q$
6. $phi = (p-1)(q-1)$
7. Generate e , $\gcd(e, phi) = 1$
8. $\gcd(e,d)\ mod\ phi = 1$

Proses pembangkitan kunci menggunakan panjang bit bilangan prima yang akan diteliti. Nilai pEL merupakan inputan prima algoritma El Gamal dan nilai n merupakan hasil perkalian inputan nilai p dan q. Nilai x pada kunci privat merupakan nilai acak pada yang bergantung pada nilai pEL yang mana nilai x harus memenuhi syarat pEL > x. Variable e kunci public ditemukan pada proses $\gcd(e, phi) = 1$ yang mana nilai $phi = (p-1)(q-1)$. Nilai e digunakan untuk enkripsi pesan yang bersifat kunci public. Nilai d pada kunci privat digunakan untuk mendekripsi pesan yang mana nilai d ditemukan menggunakan rumus $\gcd(e,d)\ mod\ phi = 1$.

Algoritma 2. Enkripsi Hybrid El Gamal RSA

Input : kunci publik (y, g, pEL, n, e), plaintext (M)
Output : Ciphertext (c)

1. Konversi m ke blok-blok char
2. Generate $k,\ \ 1 \leq k \leq pEL - 2$
3. For m[0] to m[n]
4. $a = g^k\ mod\ pEL$
5. $b = y^k\ m\ mod\ pEL$
6. $c = b^e mod\ n$
7. return m
8. end for

Proses pengamanan pesan dimulai dengan mengkonversi pesan ke dalam bilangan ASCII. Bilangan k merupakan bilangan acak untuk proses enkripsi pertama. Variablel a dan b merupakan hasil keluaran enkripsi tahap pertama yang akan kemudian dienkripsi lagi pada tahap kedua dengan masing-masing variable a dan b sebagai inputan. Variabe tersebut dienkripsi menjadi variable c menggunakan kunci dengan variable e.

Algoritma 3. Dekripsi Hybrid El Gamal RSA

Input : ciphertext (c) , kunci private (x, pEL, d, n)
Output : Plaintext (M)

1. For c[0] to c[n] do
2. $m = c^d\ mod\ n$
3. $C = m.a^{pEL-1-x}\ mod\ pEL$

4. end for
5. return m
6. Konversi m ASCII to plaintext

Proses dekripsi menggunakan kunci privat dan ciphertext. Ciphertext akan dipecah menjadi blok-blok tertentu dan akan melalui proses dekripsi tahap pertama yaitu dengan menggunakan kunci privat d. Hasil dekripsi tahap pertama akan menjadi inputan untuk dekripsi tahap kedua mengunakan variabel x. Hasil dekripsi ASCII akan dikonversi ke dalam karakter.

2.3 Algoritma Hybrid (El Gamal SRNN)

Algoritma Hybrid ini menggunakan penggabungan antara algoritma El Gamal dan SRNN. Proses penyandian akan melalui algoritma El Gamal lalu hasil penyandian algoritma El Gamal akan di enkripsi lagi menggunakan algoritma SRNN. Proses penyandian pesan ini terdapat tiga proses yaitu generate key, enkiripsi dan dekripsi. Algoritma Hybrid ini membutukan tiga nilai prima yaitu nilai prima p untuk El Gamal dan prima p dan q untuk algoritma RSA. Algoritma ini akan menghasilkan kunci public (y, g, pEL, n, e, u^a) dan kunci privat (x, pEL, d, a, u).

Algoritma 4. Pembangkitan Kunci Hybrid El Gamal SRNN

Input : Panjang bit
Output : kunci publik (y, g, pEL, n, e, u^a) dan kunci privat (x, pEL, d, a, u)

1. Generate pEL
2. Generate g dan x
3. Generate p dan q
4. Hitung $y = g^x mod\ pEL$
5. $n = p.q$
6. $phi = (p-1)(q-1)$
7. Pilih bilangan bulat e dengan syarat $\gcd(e, phi) = 1$
8. $\gcd(e,d)\ mod\ phi = 1$
9. Generate nilai u yang mana $u < phi - 1$
10. Generate nilai a yang mana $phi > a > u$
11. u^a

Nilai pEL merupakan inputan prima algoritma El Gamal dan nilai n merupakan hasil perkalian inputan nilai p dan q. Nilai x pada kunci privat merupakan nilai acak pada yang bergantung pada nilai pEL yang mana nilai x harus memenuhi syarat pEL > x. Variable e kunci public ditemukan pada proses $\gcd(e, phi) = 1$ yang mana nilai $phi = (p-1)(q-1)$. Nilai e digunakan untuk enkripsi pesan yang bersifat kunci public. Nilai d pada kunci privat digunakan untuk mendekripsi pesan yang mana nilai d ditemukan menggunakan rumus $\gcd(e,d)\ mod\ phi = 1$. Variable u dibangkitkan dengan syarat $u < phi - 1$ serta variable a dbangkitkan dengan syarat $phi > a > u$.

Algoritma 5. Enkripsi Hybrid El Gamal SRNN

Input : kunci publik (y, g, pEL, n, e, u^a), plaintext (M)
Output : Ciphertext C

1. Konversi m ke blok-blok char
2. Generate $k,\ \ 1 \leq k \leq pEL - 2$
3. For m[0] to m[n]
4. $a = g^k\ mod\ pEL$
5. $b = y^k\ m\ mod\ pEL$
6. $c = (bu^a)^e mod\ n$
7. return m
8. end for

Proses pengamanan pesan dimulai dengan mengkonversi pesan ke dalam bilangan ASCII. Bilangan k merupakan bilangan acak untuk proses enkripsi pertama. Variablel a dan b merupakan hasil keluaran enkripsi tahap pertama yang akan kemudian dienkripsi lagi pada tahap kedua dengan masing-masing variable a dan b sebagai inputan. Variabe tersebut dienkripsi menjadi variable c menggunakan kunci dengan variable e dan u^a.

Algoritma 6. Dekripsi Hybrid El Gamal SRNN

Input : ciphertext (C), kunci private (x, pEL, d, n, a, u)
Output : Plaintext (M)

1. For c[0] to c[n] do
2. $v = u^{(phi-a)} \ mod \ n$
3. $m = (v^e c)^d \ mod \ n$
4. $C = m.a^{pEL-1-x} \ mod \ pEL$
5. end for
6. return m
7. Konversi m ASCII to plaintext

Proses dekripsi menggunakan kunci privat dan ciphertext. Ciphertext akan dipecah menjadi blok-blok tertentu dan akan melalui proses dekripsi tahap pertama yaitu dengan menggunakan kunci privat u dan a untuk mendapatkan variable v. variable v kemudian akan digunakan untuk hasil dekripsi tahap awal yang akan dikalkulasi menggunakan kunci privat d. Hasil dekripsi tahap pertama akan menjadi inputan untuk dekripsi tahap kedua mengunakan variabel x. Hasil dekripsi ASCII akan dikonversi ke dalam karakter.

2.4 Baby step-giant step

Keamanan algoritma el-gamal tergantung pada masalah logaritma diskrit dengan modulo yang besar. Metode penyerangan baby step-giant step merupakan suatu metode dengan mencocokkan hasil operasi antara 2 list yaitu baby step dan giant step. Pseudocode baby step-giant step terdapat pada algoritma 7.

Algoritma 7. Baby step-giant step

Input : public key pEL
Output : private key x

1. $N = \sqrt{pEL - 1}$
2. Inisialisasi $j = 0$
3. For $j > N$
4. $g^j mod \ pEL$. List baby step
5. $y.g^{-N.j} mod \ pEL$. List giant step
6. Array baby step = giant step
7. End for
8. $x = bs + gs$

Untuk mencari nilai x langkah pertama yang harus dilalui adalah menghitung jumlah daftar baby step . Maka hasil dari N akan dijadikan patokan batas daftar baby step. Daftar giant step akan diproses sebanyak perkalian dengan y. Proses pencarian nilai x diambil jika daftar baby step sama dengan giant step setiap index yang sama. Proses pendaftaran ke dalam array penyimpanan sebagai berikut ;

a. Baby step, $g^j mod \ pEL, 0 \leq j \leq N$
b. Giant step, $y.g^{-N.j} mod \ pEL$

2.5 Factorization attack

Algoritma RSA dan SRNN terdapat bilangan (n) yang merupakan hasil perkalian 2 bilangan prima yang degenerate secara random pada proses pembangkitan kunci. Algoritma RSA sangat bergantung pada nilai n ini. Umumnya attacker akan menggunakan bilangan n ini sebagai celah untuk memfaktorkan bilangan dan mendapatkan nilai (d) lalu mendekripsi ciphertext dengan menghasilkan plaintext asli. Pseudocode Factorization Attack terdapat pada algoritma 8.

Algoritma 8. Factorization Attack

Input : public key (N, e, u[a])
Output : private key (d, a, u)

1. $k = 1, x = 2, y = x + 1$
2. While $k^2 \leq N$ do

3. $k++$
4. $h^2 = k^2 - N$
5. End while
6. $p = k + h$
7. $q = k - h$
8. $phi = (p-1)(q-1)$
9. $d = e^{-1} mod\ phi$
10. if $x^y < u^a$, $y++$
11. if $x^y \geq u^a\ \&\ x^y = u^a$
12. $x = u\ \&\ y = a$

Faktorisasi merupakan rumus lawan dari perkalian untuk memcahkan suatu bilangan bulat. Jika penyerang dapat menemukan nilai d dari kunci public n maka attacker akan dengan mudah memecahkan nilaid menggunakan rumus $\gcd(e,d)\ mod\ phi = 1$. Pada algoritma SRNN pun pemecahan terhadap nilai d sama dengan mekanisme pemecahan terhadap RSA tetapi SRNN menggunakan variable tambahan u dan a sebagai kunci private. Penjelasan algoritma 8 sebagai berikut;

1. Hitung bilangan bulat positif $k \approx \sqrt{n}$ dengan syarat $k^2 > n$ jika tidak memenuhi kondisi tersebut maka $k++$
2. Hitung nilai h dengan syarat $k^2 - n = h^2$ jika h bilangan bulat maka hitung $p = k + h$ dan $q = k - h$ dan jika h bilangan pecahan maka $k++$
3. Hitung nilai u dan a dengan syarat $u < a\ dan\ u^a = u^a$ jika tidak maka $u++$ dan $a = u + 1$

3. Hasil Penelitian dan Pembahasan

Pengujian meliputi pengujian aplikasi, pengujian performa dan pengujian keamanan. Khusus untuk pengujian performa dan keamanan akan menjadi tolak ukur kualitas efisiensinya.

3.1. Pengujian Aplikasi

Pengujian aplikasi ini akan menguji aplikasi sesuai dengan test case[15].

Tabel 1. Pengujian skenario aplikasi

No	Skenario Pengujan	Test Case	Hasil yang Diharapkan	Hasil Pengujian
1	Membuka aplikasi server	Buka aplikasi	Aplikasi terbuka	Sesuai yang diharapkan
2	Menjalankan server	Start server	Membuat server sukses dengan ip dan port tertentu	Sesuai yang diharapkan
3	Client satu membuka aplikasi	Buka aplikasi	Aplikasi terbuka	Sesuai yang diharapkan
4	Inisialisasi aplikasi ke server	Masukkan ip, port dan username dan klik tombol connect	Koneksi berhasil	Sesuai yang diharapkan
5	Client dua membuka aplikasi	Buka aplikasi	Aplikasi terbuka	Sesuai yang diharapkan
6	Inisialisasi aplikasi ke server	Masukkan ip, port dan username dan klik tombol connect	Koneksi berhasil	Sesuai yang diharapkan
7	Memilih algoritma client satu	Pilih algoritma dan panjang bit	Berhasil memilih algoritma	Sesuai yang diharapkan
8	Memilih algortima client dua	Pilih algoritma dan panjang bit	Berhasil memilih algoritma	Sesuai yang diharapkan

9	Memasukkan kunci privat	Memasukkan kunci privat dari client pengirim	Berhasil menginput kunci privat	Sesuai yang diharapkan
10	Client satu mengirim pesan	Mengirimkan pesan	Berhasil mengirim pesan enkripsi	Sesuai yang diharapkan
11	Client dua menerima pesan	Menerima pesan	Berhasil menerima dan medekripsi pesan	Sesuai yang diharapkan

Kesimpulan yang dapat ditarik dari percobaan pada tabel 1 bahwa pengimplementasian algoritma pada aplikasi sesuai dengan yang diharapkan. Percobaan dari 1 sampai 11 memiliki scenario yang berbeda sesuai dengan proses jalannya aplikasi. Hal ini membuktikan bahwa algoritma bisa diimplementasikan dalam proses pengiriman pesan berbasis TCP.

3.2. Pengujian performa

3.2.1 Pengujian waktu pembangkitan kunci

Pengujian pembangkitan kunci menggunakan Panjang bit bilangan prima yaitu 8, 16, 24 dan 32bit. Masing-masing algoritma akan dibandingkan berapa rata-rata waktu yang dibutuhkan masing-masing algoritma dalam proses pembangkitan kunci dengan menggunakan satuan millisecond.

Tabel 2. Perbandingan waktu pembangkitan kunci

Panjang bilangan prima (bit)	Waktu pembangkitan kunci (ms)	
	Hybrid (El Gamal RSA) (ms)	Hybrid (El Gamal SRNN) (ms)
8	4.292	4.235
16	5.860	5.207
24	6.067	7.082
32	6.533	7.392
Rata-rata	4.550	5.979

Pada tabel 2 merupakan hasil pengujian pembangkitan kunci algoritma Hybrid. Pengujian dilakukan 4 kali dengan panjang bit 8, 16, 24 dan 32 bit. Hasil pengujian dengan masing-masing panjang bit tersebut menunjukkan bahwa algoritma yang diajukan sedikit lebih lambat dibandingkan dengan algoritma pada penelitian sebelumnya dengan rata – rata waktu perbedaan pembangkitan kunci sebesar 1.429. Dampak yang sangat terlihat karena adanya variabel yang perlu ditambahkan pada setiap kunci public maupun privat. Hasil pengujian waktu pembangkitan kunci algoritma single enkripsi pada tabel 2 dapat ditarik kesimpulan bahwa panjang bit bilangan prima yang digunakan berpengaruh pada waktu yang dibutuhkan setiap algoritma untuk membangkitkan kunci. Semakin banyak bit yang digunakan maka semakin lama waktunya. Waktu pembangkitan kunci algoritma Hybrid (El Gamal – RSA) lebih baik dibandingkan dengan algoritma Hybrid (El Gamal – SRNN) karena proses pembangkitan kunci pada algoritma Hybrid (El Gamal – RSA) lebih sedikit dibandingkan dengan algoritma Hybrid (El Gamal – SRNN)

3.2.2 Pengujian waktu enkripsi

Pengujian enkripsi menggunakan Panjang bit bilangan prima yaitu 8, 16, 24 dan 32 bit. Penggunaan karakter pada pengujian ini menggunaan Panjang karakter 50, 100 dan 160. Masing-masing algoritma akan dibandingkan berapa rata-rata waktu yang dibutuhkan masing-masing algoritma dalam proses enkripsi dengan menggunakan satuan millisecond.

Tabel 3. Perbandingan waktu enkripsi

Panjang karakter	Panjang bilangan prima (bit)	Waktu enkripsi	
		Hybrid (El Gamal RSA) (ms)	Hybrid (El Gamal SRNN) (ms)
50	8	4.646	5.193
	16	4.951	5.583

	24	6.322	6.247
	32	6.577	7.284
125	8	7.248	7.703
	16	9.580	10.293
	24	12.792	12.987
	32	13.200	15.020
200	8	10.954	12.072
	16	11.217	13.849
	24	12.064	17.956
	32	15.994	18.237
Rata -rata		9.628	11.035

Pada tabel 3 merupakan hasil pengujian waktu enkripsi algoritma Hybrid. Pengujian dilakukan 4 kali dengan panjang bit 8, 16, 24 dan 32 bit dengan masing-masing pengujian dengan panjang karakter sebanyak 50, 125 dan 200 karakter. Hasil pengujian dengan masing-masing panjang bit tersebut menunjukkan bahwa algoritma yang diajukan sedikit lebih lambat dibandingkan dengan algoritma pada penelitian sebelumnya dengan rata – rata waktu perbedaan enkripsi sebesar 1.407. Hasil pengujian waktu enkripsi algoritma Hybrid pada tabel 3 dapat ditarik kesimpulan bahwa panjang bit bilangan prima dan panjang karakter yang digunakan berpengaruh pada waktu yang dibutuhkan setiap algoritma dalam proses enkripsi. Semakin panjang bit dan karakter yang digunakan maka semakin lama waktu enkripsinya. Waktu enkripsi algoritma Hybrid (El Gamal – RSA) lebih baik dibandingkan dengan Hybrid (El Gamal – SRNN) karena proses enkripsi pada algoritma Hybrid (El Gamal – RSA) hanya menggunakan nilai e sedangkan pada algoritma Hybrid (El Gamal – SRNN) menggunakan nilai e dan u^a yang besar hasil penambahan variable pada proses pembangkitan kunci.

3.2.3. Pengujian waktu dekripsi

Pengujian dekripsi menggunakan Panjang bit bilangan prima yaitu 8, 16, 24 dan 32 bit. Penggunaan karakter pada pengujian ini menggunaan Panjang karakter 50, 100 dan 160. Masing-masing algoritma akan dibandingkan berapa rata-rata waktu yang dibutuhkan masing-masing algoritma dalam proses dekripsi dengan menggunakan satuan millisecond.

Tabel 4. Perbandingan waktu dekripsi

Panjang karakter	Panjang bilangan prima (bit)	Waktu dekripsi	
		Hybrid (El Gamal RSA) (ms)	Hybrid (El Gamal SRNN) (ms)
50	8	4.058	4.479
	16	6.734	7.749
	24	8.871	9.990
	32	10.852	13.256
125	8	7.594	8.517
	16	10.748	12.072
	24	11.526	13.989
	32	14.350	15.874
200	8	8.969	9.742
	16	12.269	14.261
	24	15.706	18.509
	32	20.022	21.449
Rata -rata		10.974	12.490

Pada tabel 4 merupakan hasil pengujian waktu enkripsi algoritma Hybrid. Pengujian dilakukan 4 kali dengan panjang bit 8, 16, 24 dan 32 bit dengan masing-masing pengujian dengan panjang karakter sebanyak 50, 125 dan 200 karakter. Hasil pengujian dengan masing-masing panjang bit tersebut menunjukkan bahwa algoritma yang diajukan sedikit lebih lambat dibandingkan dengan algoritma pada penelitian sebelumnya dengan rata – rata waktu perbedaan enkripsi sebesar 1.516 . Hasil pengujian waktu dekrispi pada tabel 4 dapat ditarik kesimpulan bahwa panjang bit bilangan prima dan panjang karakter yang digunakan berpengaruh pada waktu yang dibutuhkan setiap algoritma dalam proses dekripsi. Semakin panjang bit dan karakter yang digunakan maka semakin lama waktu dekripsinya. Waktu dekripsi algoritma Hybrid (El Gamal –

RSA) lebih baik dibandingkan dengan Hybrid (El Gamal – SRNN) karena proses dekripsi pada algoritma Hybrid (El Gamal – RSA) hanya menggunakan nilai d sedangkan pada algoritma Hybrid (El Gamal – SRNN) menggunakan nilai d, a, u yang besar.

3.3. Pengujian Keamanan

Pengujian keamanan pada penelirian ini menggunakan panjang bit 16 dan 32 bit. Waktu ekesekusi pengujian menggunakan satuan milisecond. Kolom status kunci merupakan keterangan apakah kunci ditemukan atau belum ditemukan dan berhenti proses pemecahan. Masing-masing algoritma akan dibandingkan berapa rata-rata waktu yang dibutuhkan masing-masing untuk memecahkan kunci privat menggunakan satuan millisecond.

Tabel 5. Perbandingan pengujian keamanan

Algoritma	Panjang bit bilangan prima (bit)	Waktu eksekusi (ms)	Status kunci
Hybrid (El Gamal – RSA)	16	109.634	Kunci privat sesuai
	32	132138.506	Kunci privat sesuai
Hybrid (El Gamal – SRNN)	16	-	Kunci privat d dan x ditemukan , kunci privat u dan a belum ditemukan
	32	-	Kunci privat d dan x ditemukan , kunci privat u dan a belum ditemukan

Kesimpulan yang dapat ditarik dari percobaan pada tabel 5 adalah algoritma Hybrid (El Gamal – RSA) masih bisa diserang menggunakan Hybrid attack (fermat factorization & baby step giant step) dengan percobaan menggunakan panjang bit 16 dan 32 bit. Serangan ini dapat menemukan kunci privat d pada algoritma RSA dan x pada algoritma El Gamal. Pada percobaan 16 bit menghabiskan waktu selama 109.634 ms sedangkan pada percobaan 32 bit menghabiskan waktu selama 132138.506 ms. Hal ini dikarenakan semakin panjang bit bilangan prima maka semakin lama waktu pemecahannya karena pemfaktoran nilai n semakin besar. Hybrid attack (fermat factorization & baby step giant step) belum mampu memecahkan kunci privat u dan a dari algoritma Hybrid (El Gamal – SRNN) karena hasil pemangkatan u^a sebagai kunci public memiliki panjang bit yang relative besar sehingga sangat sulit untuk menemukan kunci privat u dan a.

4. Kesimpulan

Algoritma yang diajukan pada penelitian ini Hybrid (El Gamal SRNN) mampu memaksimalkan kekurangan penelitian sebelumnya yang menggunakan algoritma Hybrid (El Gamal RSA) pada segi keamanannya. Pengujian keamanan menggunakan panjang bit 16 dan 32 bit diatas menunjukkan bahwa algoritma yang diajukan masih belum bisa dipecahkan. Keamanan algoritma ini terletak pada penambahan variabel u dan a. algortima ini masih memiliki kelemahan pada segi performa yang telah diteliti menunjukkan bahwa waktu pembangkitan kunci, enkripsi dan dekripsi masih belum efisien. Pengujian performa pada algoritma yang diajukan(El Gamal SRNN) masih lebih lambat dibandingkan algoritma Hybrid (El Gamal RSA). Penelitian ini dapat dikembangkan dengan menggunakan media selain pengiriman pesan. Media lain seperti pengamanan file dan pengiriman file jaringan local. Pengembangan algoritma pun dapat dilakukan untuk meningkatkan processing speed algoritma ini dengan menganalisa kembali agar proseccing speed lebih baik.

Refrensi

[1] S. H. Pramono and O. Setyawati, “Optimasi Keamanan Instant message pada Sistem Operasi Android,” vol. 9, no. 2, pp. 173–178, 2015.
[2] W. Stallings, *Cryptography and Network Security PrinciPles and Practice Sixth edition*, Sixth edit. Boston: T. Johnson, Ed., boston: PEARSON, 2014.
[3] R. Tripathi and S. Agrawal, “Comparative Study of Symmetric and Asymmetric

Cryptography Techniques," vol. 1, no. 6, pp. 68–76, 2014.
[4] A. F. Helmi, S. Arifianto, J. T. Informatika, and U. M. Malang, "ANALISA KOMBINASI ALGORITMA MERKLE-HELLMAN KNAPSACK DAN ANALYSIS OF A COMBINATION OF MERKLE-HELLMAN ALGORITHMS AND," vol. 5, no. 3, pp. 325–334, 2018.
[5] H. Sandityas, "Analisa Hybrid Kriptosistem RSA dan EL-GAMAL pada Instant Messaging berbasis socket TCP," Malang, 2018.
[6] A. Gadhing Putra, Aminudin, and S. Arifianto, "Improvisasi Algoritma RSA Menggunakan Generate Key ESRKGS pada Instant Messaging Berbasis Socket TCP," 2019.
[7] Aminudin, K. Anggit, and S. Arifianto, *The implementation of Asymmetric Algorithms Dual Modulus RSA in Applications Chat*. Malang: Universitas Muhammadiyah Malang, 2018.
[8] R. Singh and S. Kumar, "Elgamal ' s Algorithm in Cryptography," vol. 3, no. 12, pp. 3–6, 2012.
[9] P. A. Kameswari, T. Surendra, and B. Ravitheja, "Shank ' s Baby-Step Giant-Step Attack Extended To Discrete Log with Lucas Sequences," vol. 12, no. 1, pp. 9–16, 2016.
[10] B. R. Ambedkar and S. S. Bedi, "A New Factorization Method to Factorize RSA Public Key Encryption," vol. 8, no. 6, pp. 242–247, 2011.
[11] L. Kalita, "Socket Programming," vol. 5, no. 3, pp. 4802–4807, 2014.
[12] B. Swami, R. Singh, and S. Choudhary, "Dual Modulus RSA Based on Jordan-totient Function," *Procedia Technol.*, vol. 24, pp. 1581–1586, 2016.
[13] M. Sharma, "A Hybrid Cryptosystem Approach for File Security by using Merging Mechanism," pp. 713–717, 2016.
[14] H. Kumar and A. Singh, "An Efficient Implementation of Digital Signature Algorithm with SRNN Public Key Cryptography," no. 1, pp. 54–57, 2012.
[15] S. Hozeng, "Perancangan Aplikasi Enkripsi Menggunakan Algoritma AES Berbasis Android Encryption Application Design Using Android-Based AES Algorithm," pp. 130–135, 2019.