ENKRIPSI DAN DEKRIPSI GAMBAR RGB DENGAN ALGORITMA RIVEST SHAMIR ADLEMEN PADA SMARTPHONE BERBASIS ANDROID

Akbar Riwanda[1], Hendrawaty[2]

[1,2] *Jurusan Teknologi Informasi dan Komputer Politeknik Negeri Lhokseumawe*
Jln. B.Aceh Medan Km.280 Buketrata 24301 INDONESIA

[1]akbarriwanda@gmail.com

[2]waty.hendra@yahoo.com

Abstrak— Seiring perkembangan jaman, Internet menjadi media yang paling banyak digunakan untuk melakukan komunikasi dan mengirimkan informasi, begitu juga dengan penggunaan *smartphone* yang berbasis *android*. Dengan perkembangan teknologi pada *smartphone* dan perkembangan aplikasi pada perangkat *mobile* tersebut memungkinkan penggunanya untuk melakukan pengiriman data seperti gambar melalui jaringan internet. Gambar termasuk salah satu data yang banyak dipertukarkan/dikirimkan melalui internet. Akan tetapi pengiriman data lewat internet masih rentan terhadap pencurian data, sehingga diperlukan suatu cara yang aman untuk mengirimkan gambar melalui internet. Pada penelitian ini diimplementasikan metode kriptografi untuk menjaga kerahasiaan *file* gambar yang dikirimkan melalui internet agar aman dari pihak yang tidak berkepentingan. Dari permasalahan tersebut dirancang dan dibangun suatu aplikasi berbasis *android* yang dapat digunakan untuk enkripsi *file* gambar yang akan dikirim melalui internet dan juga dapat digunakan untuk dekripsi gambar yang telah teracak menjadi gambar aslinya. Pada Penelitian ini, Algoritma yang digunakan adalah *Rivest Shamir Adleman* (RSA), Jenis gambar yang digunakan adalah RGB 24-bit berformat JPG/JPEG dengan dimensi hasil normalisasi 250 x 250 pixel, dan sampel *file image* yang diuji sebanyak 20 buah. Dari hasil pengujian menggunakan *smartphone Asus ZE500KL* dengan *android* versi 5.0 dan *Oppo A37* dengan *android* versi 5.0.2 diperoleh bahwa aplikasi yang dibuat sudah berhasil melakukan enkripsi dan dekripsi.

Kata kunci— Algoritma RSA, *Rivest Shamir Adleman* (RSA), *Android*, Kriptogafi, Enkripsi Gambar, Dekripsi Gambar.

*Abstract***—** *As the times progress, the Internet became the most widely used media to communicate and transmit information, as well as the use of android-based smartphone. With the development of technology on smartphones and the development of applications on mobile devices that allow users to send data such as images via the Internet network. Images are among the data that are often exchanged / transmitted over the internet. However, data transmission over the internet is still vulnerable to theft, so it needs a safe way to transmit images over the internet. In this study implemented cryptographic methods to maintain the confidentiality of image files sent over the internet to be safe from unauthorized parties..*

Keywords— RSA Algorithm, Rivest Shamir Adleman (RSA), Android, Kriptogafi, Image Encryption, Image Decryption.

I. PENDAHULUAN

Semakin berkembangnya teknologi informasi di zaman sekarang ini membuat berbagai hal dapat dilakukan melalui media internet. Mengirim informasi berupa *file* dokumen, gambar, musik dan juga lain sebagainya sangatlah mudah. Sehingga dalam waktu singkat seluruh informasi yang ingin dibagikan dapat diakses oleh penerima yang dituju. Dalam setiap detik terdapat ribuan bahkan sampai jutaan data yang dikirimkan melalui media internet yang ada diseluruh dunia, baik yang digunakan secara pribadi antara orang per-orang, bahkan antar perusahaan. Dalam mengalirnya informasi melalui internet, bukan tidak mungkin untuk setiap data yang lalu lalang itu dapat diketahui oleh pihak lain yang tidak berhak, banyak fakta yang dapat dilihat bahwa data-data tersebut dapat diketahui oleh orang lain. Untuk itu sangatlah diperlukan suatu keamanan yang dapat digunakan untuk menjaga agar setiap informasi yang dikirimkan melalui media internet dapat terjaga kerahasiaannya sehingga tidak dapat diketahui oleh pihak yang tidak berhak. Gambar merupakan suatu *file* yang sering digunakan terlebih dimedia sosial, namun gambar juga sering digunakan oleh jasa layanan seperti fotografer, desainer logo, dan lain-lain. Untuk itu, dibutuhkan suatu keamanan agar *file* gambar tersebut aman saat dikirim melalui media internet.

Beberapa penelitian yang berhubungan dengan sistem keamanan dengan metode kriptografi sudah pernah dilakukan sebelumnya. Berikut penjelasan secara garis besar beberapa penelitian yang terkait dengan enkripsi dan dekripsi file gambar dan smartphone android.

Pada tahun 2014, Saranya et al.[1] telah melakukan study yang membahas tentang sejarah algoritma RSA dan membahas secara detail tentang cara kerja algoritma RSA sebagai algoritma kriptografi mulai dari proses pembangkitan kunci, proses enkripsi dan proses dekripsi. Pada tahun 2014, Ali E et al.[2] telah melakukan penelitian yang membahas enkripsi gambar grayscale menggunakan algoritma RSA, dan melakukan perbandingan dengan algoritma lain seperti DES dan Blowfish. Pada tahun Pada tahun 2015, *Andro Alif Rakhman & Achmad Wahid Kurniawan* [3] telah melakukan sebuah penelitian tentang Kriptografi dengan menggunakan Algoritma *Kriptografi Rivest Shamir Adleman (RSA*) Dan *Vigenere Cipher* Pada Gambar *Bitmap 8 Bit*. Pada tahun 2016, *Siswo Wardoyo, Zaldi Imanullah* dan *Rian Fahrizal* [4] telah melakukan sebuah penelitian tentang Enkripsi dan Dekripsi File Pada Perangkat Mobile Berbasis Android. *Rini Wati Lumbangaol* [5] telah melakukan penelitian dengan menggunakan Algoritma RSA untuk mengamankan file gambar. Pada tahun 2015, S.Anandakumar [6] telah melakukan penelitian tentang implementasi algoritma *Kriptografi RSA* Untuk mengenkripsi gambar. Pada tahun 2017, Samson Chepury [7], telah melakukan enkripsi dan dekripsi gambar RGB dengan menggunakan matlab. Metode yang digunakan pada penelitian tersebut adalah RSA.

Berdasarkan latar belakang dan penelitian-penelitian yang telah disebutkan diatas, maka penelitian ini bertujuan membahas tentang rancang bangaun aplikasi enkripsi dan

dekripsi gambar RGB 24 bit yang dapat berjalan pada smartphone berbasis android. Untuk proses enkripsi dan dekripsi-nya digunakan algoritma RSA. Bahasa pemograman yang digunakan adalah *java* untuk *platform android*, dan versi *android* yang digunakan adalah v.4.2 hingga v.5.0.

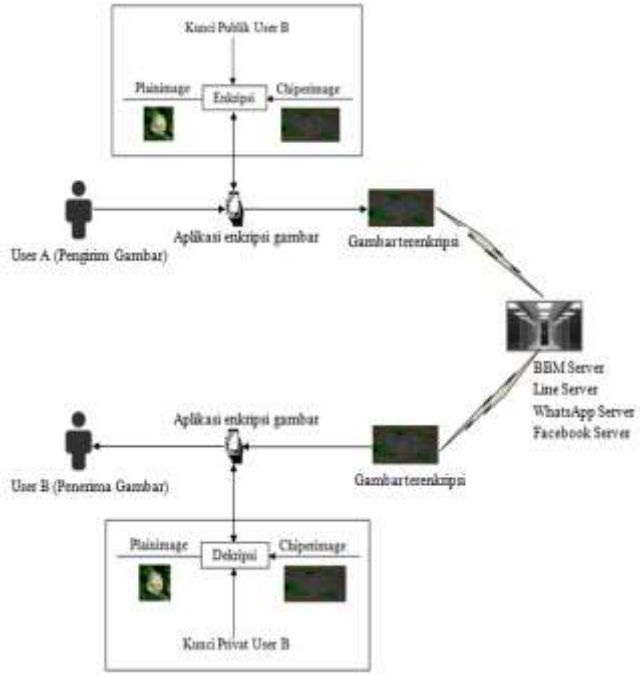
RSA merupakan singkatan dari Ron Rivest, Adi Shamir and Leonard Adleman, yang menemukan algoritma ini pada tahun 1977 [8]. Algoritma RSA merupakan teknik kriptografi modern dan merupakan algoritma kunci asimetris. Algoritma tersebut memiliki dua buah kunci yaitu kunci *public* dan kunci *private*. Kunci *public* digunakan untuk enkripsi pesan dan kunci *private* digunakan untuk dekripsi pesan. RSA merupakan algoritma kriptografi kunci publik pertama yang praktis dan yang banyak digunakan untuk mengamankan transmisi data. Dalam kriptografi RSA, kunci publik tidak bersifat rahasia (diketahui oleh pengirim dan penerima) sedangkan kunci privat hanya boleh diketahui oleh pihak pembuat/pemilik kunci saja (tidak boleh diketahui oleh pihak lain) [9].

Kekuatan algoritma RSA terletak pada tingkat kesulitan dalam memfaktorkan bilangan prima yang besar menjadi faktor-faktor prima. Pemfaktoran dilakukan untuk memperoleh kunci privat. Selama pemfaktoran bilangan besar menjadi faktor-faktor prima belum ditemukan, maka selama itu pula keamanan algoritma RSA tetap terjamin [10]. Selain sulitnya memfaktorkan bilangan yang besar pada pembentukan kunci pada algoritma *Rivest Shamir Adleman (RSA), Rivest Shamir Adleman (RSA)* menawarkan kemampuan untuk mengendalikan panjang kunci yang digunakan oleh penggunanya sehingga dapat menyesuaikan dengan kebutuhan pengguna. Skema algoritma kriptografi kunci publik (asimetris) *Rivest Shamir Adleman (RSA)* terdiri dari tiga proses yaitu pembentukan kunci, proses enkripsi, dan proses dekripsi [11].

II. METODOLOGI PENELITIAN

Perancangan penerapan algoritma RSA pada penelitian ini dapat dilihat pada Gambar 1. Pada Gambar 1 dapat dilihat bahwa, *user* A (pengirim) meng-enkripsi gambar yang akan dikirimkannya ke *user* B (penerima) dengan kunci publik *user* B. *User* A kemudian mengirimkan *cipherimage* kepada *user* B melalui *smartphone*. Kemudian user B men-dekripsi *cipherimage* tersebut dengan kunci privat miliknya untuk mendapatkan gambar asli.

Telah disebutkan sebelumnya bahwa, skema algoritma kriptografi *RSA* terdiri dari tiga proses yaitu pembentukan kunci, proses enkripsi, dan proses dekripsi. Berikut penjelasan tentang masing-masing proses tersebut.



Gambar 1. Ilustrasi Perancangan Penerapan Algoritma *RSA* Pada Aplikasi Enkripsi Dekripsi Gambar

A. Proses Pembangkitan Kunci

Tahap-tahap pembentukan kunci adalah:

1. Tentukan dua bilangan prima besar p dan q secara acak dan dirahasiakan. Nilai $p \neq q$, p dan q memiliki ukuran sama. sebaiknya $p \neq q$, sebab jika $p = q$ maka $n = p^2$ sehingga p dapat diperoleh dengan numeric akar pangkat dua dari n.
2. Kemudian dihitung n RSA (*modulus*) dengan rumus :

$$n = p * q \quad (1)$$

Sifat dari n tidak rahasia.

3. Kmudian hitung $\varphi(n)$ dengan rumus :

$$\varphi(n) = (p-1)*(q-1) \quad (2)$$

4. Piih sebuah bilangan bulat sebagai kunci publik (e), yang relatif prima terhadap $\varphi(n)$ dan $1 < e < \varphi(n)$. e relatif prima terhadap $\varphi(n)$ artinya faktor pembagi terbesar keduanya adalah 1, secara matematis disebut *gcd* $(e, \varphi(n)) = 1$. Untuk mencarinya dapat digunakan algoritma *Euclid.* Sifat bilangan e tidak rahasia.
5. Bangkitkan kunci privat (d) dengan rumus:

$$d = (1 + k.\ \varphi(n)) / e \quad (3)$$

6. Kunci publik = (e,n) dan kunci privat (d,n)

B. Proses Enkripsi

Pada proses enkripsi, Plainteks disusun menjadi blok-blok M_1, M_2, …,M_n sedemikian sehingga setiap blok merepresentasikan nilai di dalam rentang 0 sampai n – 1. Kemudian setiap blok M_i dienkripsi menjadi blok C_i dengan rumus :

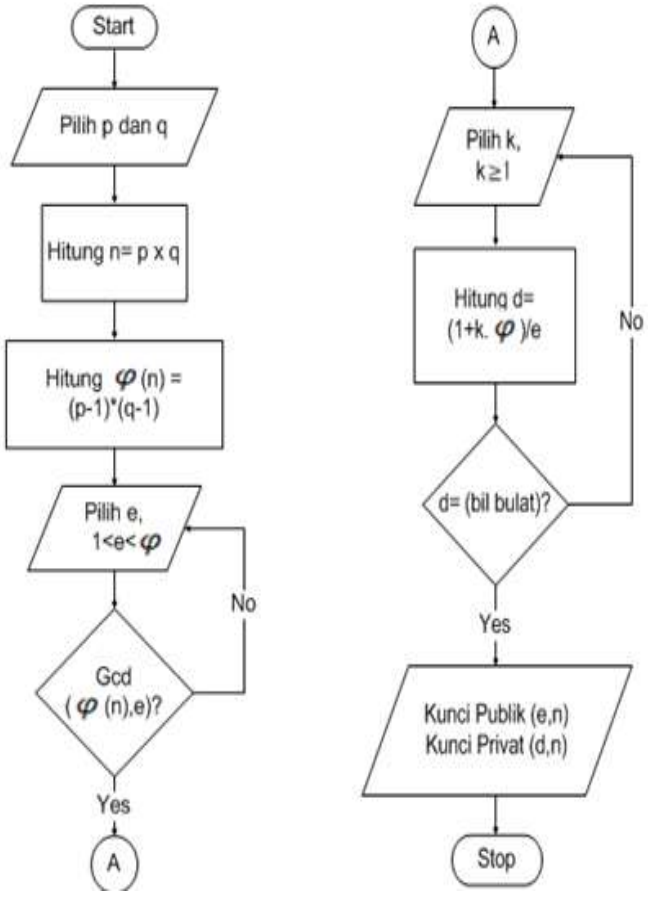
$$C_i = M_i^{\,e} \bmod n \qquad (4)$$

C. Proses Dekripsi

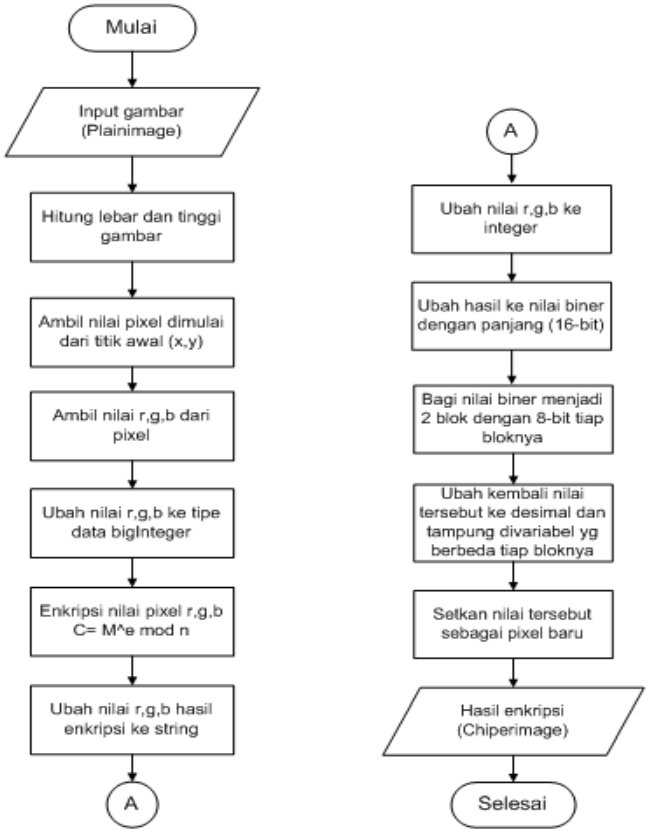
Setiap blok chiperteks *Ci* didekripsikan kembali menjadi blok M_i dengan rumus :

$$M_i = C_i^{\,d} \bmod n \qquad (5)$$

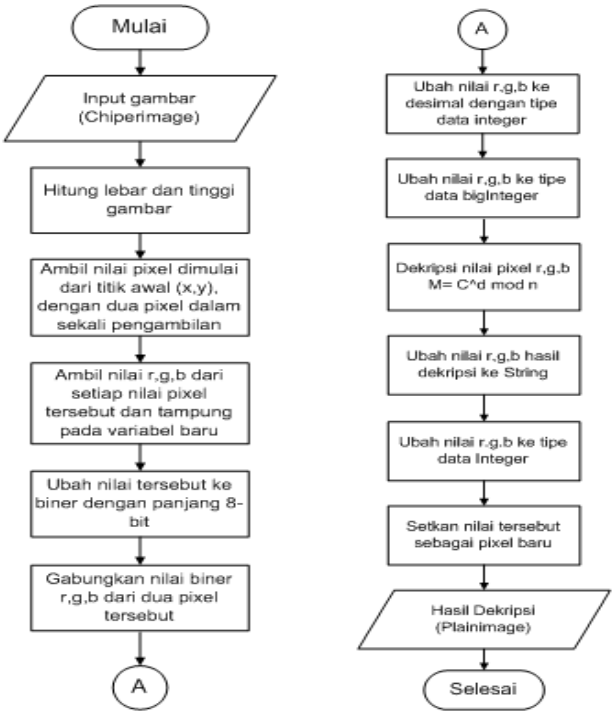
Perancangan *flowchart* proses pembuatan kunci, proses enkripsi, dan proses dekripsi masing-masing dapat dilihat pada Gambar 2., Gambar 3, dan Gambar 4.



Gambar 2. *Flowchart* Pembuatan Kunci



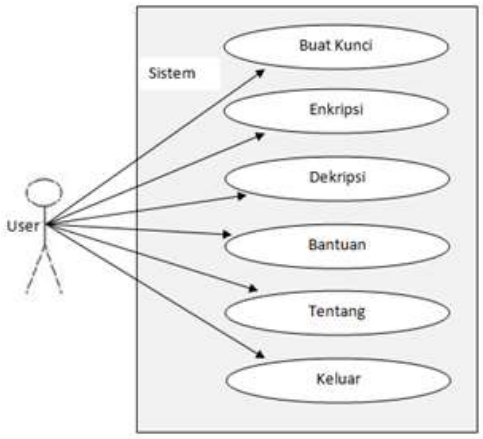
Gambar 3. *Flowchart* Perancangan Proses Enkripsi



Gambar 4. *Flowchart* Perancangan Proses Dekripsi

Perancangan *Use case diagram* Halaman utama dari aplikasi enkripsi dan dekripsi gambar dengan *smartphone*

berbasis *android* pada penelitian ini dapat dilihat pada Gambar 5.



Gambar 5. *Use case Diagram* Aplikasi Enkripsi dan Dekripsi Gambar

Pada penelitian ini, semua proses mulai dari pembangkitan kunci, proses enkripsi dan proses dekripsi dibangun dengan menggunakan bahasa pemrograman java. Aplikasi yang sudah dibuat kemudian dijalankan pada dua buah smartphone yang berbasis android yaitu *smartphone ASUS ZaneFone Laser ZE500KL* dengan sistem operasi android 5.0.2, dan *smartphone Oppo A37* yang menggunakan android 5.2.

III. HASIL DAN PEMBAHASAN

Pada tahap ini dilakukan pembahasan hasil pengujian yang terdiri dari hasil tampilan *user interface*, hasil proses pembangkitan kunci, hasil proses enkripsi dan hasil proses dekripsi. Gambar 6 menunjukkan tampilan menu utama dan Gambar 7 menunjukkan tampilan untuk pembuatan kunci pada *smartphon*e.

Gambar 6. Tampilan Menu Utama

Gambar 7. Tampilan Menu Buat Kunci

Tabel I memperlihatkan beberapa hasil pengujian dari pembangkitan kunci.

Tabel 1
Hasil Pengujian Pembangkitan Kunci

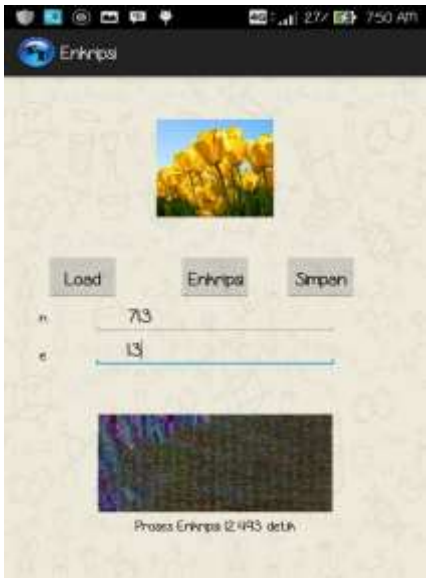
No	Kunci Publik	Kunci Private
1.	e =5 , n =7031	d=1373, n= 7031
2.	e =7 , n =7979	d=3343 , n=7979
3.	e =5 , n =11303	d=6653 , n=11303
4.	e =5 , n =11227	d=8813 , n=11227

Berdasarkan hasil dari tabel I dapat dilihat kunci yang dihasilkan berbeda-beda dan memiliki panjang kunci maksimal 14 bit. Kunci yang dihasilkan/dibangkitkan ada dua yaitu kunci publik (e,n) dan kunci privat (d,n). Panjang kunci yang dibangkitkan pada penelitian ini masih relatif pendek yaitu 14 bit. Untuk meningkatkan keamanan dari gambar yang di enkripsi sebaiknya kunci yang digunakan jauh lebih panjang.

Dari beberapa pengujian pembangkitan kunci yang telah dilakukan, diperoleh bahwa rata-rata waktu yang dibutuhkan untuk pembangkitkan kunci dengan menggunakan smartphone android pada penelitian ini adalah 0,028 detik pada *smartphone ASUS ZaneFone Laser ZE500KL* yang menggunakan sistem operasi android 5.0.2 dan 0,25 detik pada *smartphone Oppo A37* yang menggunakan android 5.2.

Dengan memilih salah satu pasangan kunci yang telah dibangkitan (kunci publik dan privat), dilakukan proses enkripsi dan dekripsi. Gambar 8 memperlihatkan tampilan untuk proses enkripsi yang ada pada smartphone android. Proses enkripsi dilakukan dengan cara menekan tombol Load untuk menginputkan gambar yang akan dienkripsi. Kemudian

nilai kunci public (e , n) diinputkan. Dengan menekan tombol Enkripsi maka gambar tersebut diproses, dan hasil enkripsinya dan lama waktu proses enkripsi akan tampil.



Gambar 8. Tampilan Menu Enkripsi

Tabel II memperlihatkan hasil yang diperoleh dari proses enkripsi menggunakan kunci publik (e = 5, n =7031).

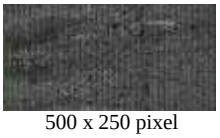
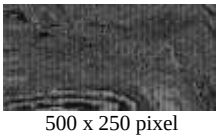
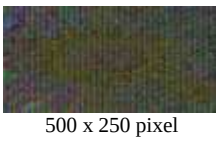
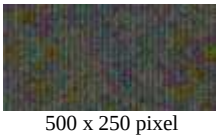
Pada Tabel II dapat dilihat bahwa ukuran gambar asli (plainimage) mengalami perubahan setelah dienkripsi. Ukuran pixel dari hasil enripsi lebih besar dari gambar aslinya. Setelah proses enkripsi, *Plainimage yang* sebelumnya memiliki ukuran dimensi hasil normalisasi 250 x 250 pixel, berubah menjadi 500 x 250 pixel.Perubahan tersebut disebabkan karena metode yang diterapkan pada saat melakukan proses enkripsi.

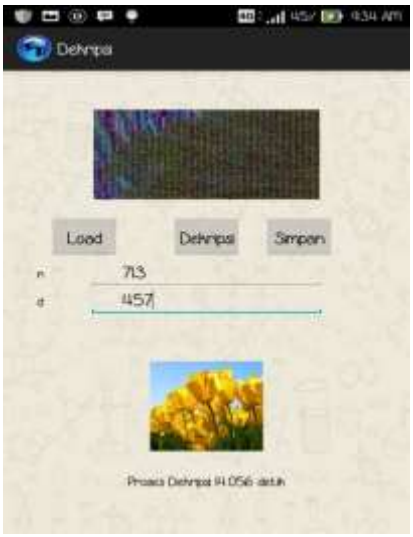
Perubahan juga terjadi pada tampilan gambar. Setelah proses enkripsi, Gambar asli (*Plainimage)* yang sebelumnya dapat dimengerti isinya berubah menjadi gambar yang tidak dapat dimengerti isinya sama sekali (teracak). Dengan demikian dapat dilihat bahwa proses enkripsi dengan kunci publik yang telah dibangkitkan berhasil dilakukan.

Dari 20 file gambar yang telah diuji sebagai sampel, diperoleh bahwa semua file gambar tersebut berhasil dienkripsi. kecepatan proses enkripsi rata-rata yang diperoleh adalah besar 15,6724 detik.

Gambar 9 memperlihatkan tampilan untuk proses dekripsi pada smartphone android. Pada tampilan tersebut terdapat tombol Load untuk menginputkan *cipherimage yang akan didekripsi*, f*ield* untuk menginputkan kunci privat (nilai d dan nilai n), tombol Dekripsi untuk melakukan proses dekripsi, *space* untuk menampilkan *cipherimage* yang akan didekripsi, *space* untuk menampilkan hasil dekripsi, tombol Simpan untuk menyimpan gambar hasil proses dekripsi, dan pada tampilan tersebut juga terdapat keterangan lamanya/waktu proses dekripsi berlangsung.

Tabel 2
Hasil Pengujian Proses Enkripsi

No	Gambar asli (*Plainimage*)	Hasil Enkripsi (*Cipherimage*)	Waktu Enkripsi
1	 250 x 250 pixel 36,6 Kb	 500 x 250 pixel 175 Kb	15,783 detik
2	 250 x 250 pixel 32,2 Kb	 500 x 250 pixel 156 Kb	15,601 detik
3	 250 x 250 pixel 50,4 Kb	 500 x 250 pixel 343 Kb	15,928 detik
4	 250 x 250 pixel 59,0 Kb	 500 x 250 pixel 341 Kb	15,549 detik
5	 250 x 250 pixel 52,0 Kb	 500 x 250 pixel 350 Kb	15,501 detik



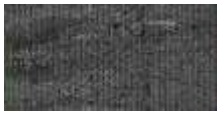
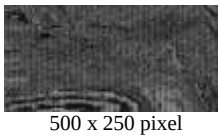
Gambar 9. Tampilan Menu Dekripsi

Tabel III memperlihatkan beberapa contoh hasil yang diperoleh dari proses dekripsi *cipherimage* dengan menggunakan kunci privat (d=1373, n = 7031). Dari Tabel III dapat dilihat bahwa setelah proses dekripsi, ukuran gambar hasil enkripsi *(Cipherimage)* berubah menjadi ukuran gambar aslinya. *Cipherimage* yang sebelumnya memiliki ukuran 500 x 250 pixel berubah ukurannya menjadi 250 x 250 pixel kembali.

Perubahan juga terjadi pada tampilan gambar. Setelah proses dekripsi, *Cipherimage* yang sebelumnya tidak dapat dimengerti isinya sama sekali (teracak) berubah menjadi gambar yang dapat dimengerti isinya (seperti aslinya). Dengan demikian dapat dilihat bahwa proses dekripsi dengan kunci private yang telah dibangkitkan telah berhasil.

Dari penjelasan hasil yang diperoleh tersebut dapat dilihat bahwa, proses dekripsi sudah berhasil dengan baik karena telah dapat mengembalikan cipherimage kebentuk aslinya (keadaan gambar sebelum dienkripsi). Dari 20 sampel *file* gambar yang diuji, kecepatan rata-rata untuk proses dekripsi adalah 18,8912 detik.

Tabel 3
Hasil Pengujian Proses Dekripsi

No	Hasil Enkripsi (*Cipherimage*)	Hasil Dekripsi (*Plainimage*)	Waktu Enkripsi
1	 500 x 250 pixel 175 Kb	 250 x 250 pixel 65,2 Kb	18.992 detik
2	 500 x 250 pixel 156 Kb	 250 x 250 pixel 57,6 Kb	18.95 detik
3	 500 x 250 pixel 343 Kb	 250 x 250 pixel 125 Kb	18.779 detik
4	 500 x 250 pixel 341 Kb	 250 x 250 pixel 136 Kb	18.774 detik
5	 500 x 250 pixel 350 Kb	 250 x 250 pixel 126 Kb	18.95 detik

IV. KESIMPULAN

Setelah melakukan penelitian dan pembahasan maka dapat disimpulkan bahwa proses pembangkitan kunci pada penelitian ini berhasil dengan baik, karena sudah dapat menghasilkan pasangan kunci *private* dan kunci *public* yang berbeda-beda dan dapat digunakan untuk proses enkripsi maupun dekripsi. Rata-rata waktu yang dibutuhkan untuk membangkitkan kunci pada penelitian ini adalah 0,028 detik pada *smartphone ASUS ZaneFone Laser ZE500KL* android 5.0.2 dan 0,25 detik pada *smartphone Oppo A37* android 5.2. Panjang kunci yang dibangkitkan pada aplikasi ini masih relative pendek yaitu 14 bit. Untuk meningkatkan keamanan dari gambar yang di enkripsi sebaiknya kunci yang digunakan jauh lebih panjang. Berdasarkan 20 gambar yang telah diuji diperoleh bahwa proses enkripsi dan dekripsi gambar pada penelitian ini telah berhasil dilakukan dengan rata-rata waktu enkripsi adalah 16,38325 detik dan rata-rata waktu dekripsi adalah 18,8895 detik. Ukuran gambar hasil enkripsi (*Cipherimage*) memiliki lebar dua kali lebih besar dari lebar gambar aslinya, sedangkan tingginya sama dengan tinggi gambar aslinya.

REFERENSI

[1] Saranya, Vinothini, Vasumathi, "A Study on RSA Algorithm for Cryptography", International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 5, 2014, ISSN: 0975-9646.

[2] Ali E. Taki El-Deen, El-Sayed A. El-Badawy, Sameh N. Gobran, "Digital Image Encryption Based on RSA Algorithm", IOSR Journal of Electronics and Communication Engineering (IOSR-JECE), Volume 9, Issue 1, Ver. IV (Jan. 2014), PP 69-73, e-ISSN: 2278-2834,p- ISSN: 2278-8735.

[3] Rakhman, A.A, Kurniawan, A.W., "Implementasi Algoritma Kriptografi *Rivest Shamir Adleman* (Rsa) Dan *Vigenere Cipher* Pada Gambar Bitmap 8 Bit", Techno.COM, Vol 14, No. 2, Mei 2015.

[4] Wardoyo,S., Imanullah, Z., Fahrizal, R.2016."Enkripsi dan Dekripsi *File* Dengan Algoritma *Blowfish* Pada Perangkat Mobile Berbasis *Android*", Jurnal Teknik Elektro Vol:5, No. 1.

[5] Lumbangaol, W.R,. "Aplikasi Pengamanan Gambar Dengan *Algoritma Rivest-Shamir Adleman* (RSA)". Medan, STIMIK Budidarma Medan.

[6] S.Anandakumar, "Image Cryptography Using RSA Algorithm in Network Security", IJCSET(www.ijcset.net), Vol 5, Issue 9, 326-330, September 2015, ISSN:2231-0711.

[7] Samson Chepuri, "An RGB Image Encryption using RSA Algorithm", International Journal of Courrent Trends in Engineering & Research (IJCTER), Volume 3, Issue 3, March 2017, e-ISSN 2455-1392.

[8] W. Stallings, Cryptography and Network Security: Principles and Practices, 3rd edition, Prentice Hall, NJ, 2003.

[9] K.Sony , Desowja Shaik, B.Divya Sri , G.Anitha," Improvised Asymmetric Key Encryption Algorithm Using MATLAB", IOSR Journal of Electronics and Communication Engineering (IOSR-JECE),Volume 10,Issue 2, (Mar - Apr.2015), eISSN: 2278-2834,p-ISSN: 2278-8735.

[10] Munir, Rinaldi. 2006. *Kriptografi*. Bandung: Informatika.

[11] Sunita, "Image Encryption/Decryption Using RSA Algoritm", International Journal of Computers Science and Mobile Applications, Vol.5, Issue. 5, May-2017, pg. 1-14, ISSN: 2321-8363