Desain Steganografi untuk Keamanan Gambar dengan Algoritma RSA dan LSB Berbasis Android

Edy Victor Haryanto
Universitas Potensi Utama
Jl. KL. Yos Sudarso Km. 6,5 No. 3A, Tanjung Mulia, Medan
E-mail: edyvictor@gmail.com

Abstrak

Keamanan data sangatlah penting saat ini apalagi dikirim melalui handphone android agar tidak diketahui oleh orang lain. Dalam penelitian ini dengan menggunakan metode RSA untuk mengeknripsi file plaintext dan juga menggunakan kunci sebelum dikirim ke sipenerima, plaintext tersebut akan diubah kedalam bentuk ASCII kemudian diubah ke dalam bentuk biner agar dapat disisip kedalam gambar dengan menggunakan LSB. Dengan menggunakan teknik kriptografi dengan algoritma RSA menjadi pilihan untuk mengubah pesan menjadi tidak terbaca lagi karena sampai saat ni algoritma RSA dinilai yang masih bagus tingkat keamanan nya. Pesan yang disembunyikan ke menyisipkan bit – bit pesan kedalam dalam pixel atau bit – bit pada gambar, sehingga perubahan gambar sebelum dan sesudah dilakukan penyisipan pesan tidak akan berubah dan tidak dapat dilihat oleh indra manusia. Dalam hal ini objek yang digunakan adalah gambar berekstension JPG ukuran citra 3888x2592.

Kata Kunci*—Keamanan, Kriptografi, Steganografi, RSA, LSB.*

Abstract

Data security is very important at this time let alone sent via android mobile so that it is not known by others. In this study using the RSA method to encrypt the plaintext file and also use the key before sending it to the recipient, the plaintext will be converted into ASCII format and then converted into binary form so that it can be inserted into the image using LSB. By using cryptographic techniques with the RSA algorithm it becomes an option to change the message to become unreadable because until now the RSA algorithm is judged to be a good level of security. The message is hidden to insert the message bits into pixels or bits in the image, so that changes in the image before and after the insertion of the message will not change and can not be seen by the human senses. In this case the object used is a JPG image size of 3888x2592.

Keywords*—Security, Cryptography, Steganography, RSA, LSB*

1. PENDAHULUAN

Saat ini perkembangan teknologi sudah tidak dapat terbendung lagi karena hal ini sudah memberikan banyak manfaat di berbagai aspek kehidupan. Contohnya dalam hal komunikasi, saat ini penggunaan internet sudah menjadi kebutuhan dimana hal ini memudahkan seseorang untuk bersosialiasasi dengan orang lain tanpa harus bertemu.

Keamanan tentu tidak dapat dilupakan seiring berkembangnya teknologi, bukan tidak mungkin seseorang dapat kehilangan hal yang tidak pernah terpikirkan sebelumnya hanya karena ia mengabaikan aspek keamanan. Dengan adanya masalah ini maka diperlukan sebuah cara untuk mengamankan kerahasiaan file, salah satu nya dengan cara mengubah pesan rahasia tersebut menjadi pesan acak yang tidak terbaca lagi, sehingga tidak dapat diketahui oleh orang lain yang tidak diinginkan. Cara tersebut dinamakan teknik kriptografi, dimana pesan yang akan dirahasiakan diubah

dengan melakukan perhitungan matematika sehingga menghasilkan pesan yang tidak dapat dimengerti.

Pengiriman data melalui handphone tentunya masih mudah diketahui atau disadap oleh orang lain apabila keamanan data tersebut tidak terjamin, sehingga data yang dikirim tentunya akan bisa dibaca oleh orang lain, maka dari itu pada penelitian ini membuat sebuah aplikasi yang dapat digunakan dalam ponsel android dengan menggunakan teknik kriptografi modern yaitu RSA berfungsi untuk mengkonversi pesan plainteks yang akan dikirim kedalam bentuk ASCII kemudian diubah kedalam bentuk biner agar dapat disisip ke dalam sebuah gambar dengan menggunakan metode LSB.

Dengan mengkombinasi metode kriptografi dengan metode steganografi yaitu Hill Chiper dan LSB dapat membantu menyembunyikan pesan kedalam gambar sehingga gambar yang sudah disisipi pesan tersebut tidak berubah warna dan kapasitasnya sehingga tidak memunculkan kecurigaan orang[3].

Barkah dalam penelitiannya dengan menggunakan metode F5 membuat aplikasi dalam bidang steganografi untuk membantu mengamankan sebuah pesan yang disisip kedalam sebuah gambar sehingga pesan tersebut tidak mudah diketahui oleh orang lain dan pada penelitian ini tidak menggunakan kriptografi sehingga data yang disembunyikan dapat mempengaruhi kapasitas gambar[4].

Dengan menggunakan teknik Hash-LSB dan kombinasi dengan RSA dalam menyembunyikan pesan ke dalam gambar yang berformat jpg [5][6].

2. METODE PENELITIAN

2.1. RSA

Pada tahun 1977, Ronald L. Rivest, Adi Shamir, dan Leonard M. Adleman merumuskan algoritma praktis yang mengimplementasikan sistem kriptografi kunci publik yang disebut dengan sistem kriptografi RSA. Sepasang kunci yang dipakai pada kedua proses ini adalah kunci publik (*e,n*) sebagai kunci enkripsi dan kunci privat *d* sebagai sebagai kunci dekripsi dimana *e*, *d* dan *n* adalah bilangan bulat positif. Algoritma RSA adalah sebuah *block cipher algorithm* (algoritma yang bekerja per blok data) yang mengelompokkan plaintext menjadi blok-blok terlebih dahulu sebelum dilakukan enkripsi hingga menjadi ciphertext [2].

Proses enkripsi RSA dilakukan dengan membangkitkan kunci dari dua buah bilangan prima untuk mendapatkan nilai n, Θn, e, dan d. Sebagai contoh digunakan dua buah bilangan prima p = 31 dan q = 37. Rumus pembangkit kunci algoritma RSA adalah sebagai berikut *:*

a. Mencari nilai n :
   $n = pxq$
b. Mencari nilai Θn :
   $\Theta n = (p-1)x\,(q-1)$
c. Mencari nilai e :
   e = 2
   While Θn mod e ≠ 0
   e = e + 1
   End While
d. Mencari nilai d :
   $U_1 = 1$
   $U_2 = 0$
   $U_3 = \Theta n$
   $V_1 = 0$
   $V_2 = 1$
   $V_3 = e$
   While $V_3 = 0$
      $Q = Int(U_3 / V_3)$
      $N_1 = U_1 - (Q \times V_1)$

$N_2 = U_2 - (Q \times V_2)$
$N_3 = U_3 - (Q \times V_3)$

$U_1 = V_1$
$U_2 = V_2$
$U_3 = V_3$
$V_1 = N_1$
$V_2 = N_2$
$V_3 = N_3$
End While

Kemudian untuk melakukan proses enkripsi diperlukan algoritma enkripsi untuk mengubah plainteks menjadi cipherteks. Berikut adalah algoritma enkripsi RSA :

$$Ci = P\frac{e}{i}\ mod\ n$$

C_i = Cipherteks hasil enkripsi
P_i = Plainteks atau pesan asli
e = Kunci enkripsi
n = Nilai n

Setelah proses enkripsi berhasil tentu harus ada cara agar cipherteks bisa dikembalikan menjadi plainteks, yaitu dengan cara dekripsi. Berikut adalah algoritma dekripsi RSA :

$$Pi = C\frac{d}{i}\ mod\ n$$

P_i = Plainteks atau pesan asli
C_i = Cipherteks hasil enkripsi
d = Kunci dekripsi
n = Nilai n

2.2. LSB

LSB adalah salah satu teknik dari steganografi yang masih sering digunakan dan masih sederhana serta sangat mudah diterapkan kedalam sebuah aplikasi. LSB ini menggunakan gambar yang berformat digital sebagai tempat penyisipan teks atau pesan. Pada susunan bit di dalam sebuah byte (1 *byte* = 8 bit), ada bit yang paling berarti (*most significant bit* atau MSB) dan bit yang paling kurang berarti (*least significant bit* atau LSB). Sebagai contoh byte 11010010, angka bit 1 (pertama, digaris-bawahi) adalah bit MSB, dan angka bit 0 (terakhir, digaris-bawahi) adalah bit LSB. Bit yang cocok untuk diganti adalah bit LSB, sebab perubahan tersebut hanya mengubah nilai *byte* satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Sebagai contoh sebuah segmen pixel gambar sebelum disisipkan bit pesan adalah :

11001010	01101110	11100111	10011001
00110111	10110011	10101011	00011111

Akan disisipkan sebuah pesan rahasia yang sudah dikonversikan menjadi bentuk biner misalkan "10110011", setiap bit dari pesan rahasia tersebut akan menggantikan bit – bit terakhir dari pixel gambar menjadi :

1100101**1**	0110111**0**	1110011**1**	1001100**1**
0011011**0**	1011001**0**	1010101**1**	0001111**1**

2.3. Pengujian

MSE merupakan hasil kesalahan rata – rata kuadrat dari gambar asli dengan gambar hasil. Pengujian MSE dikatakan baik jika mempunyai nilai yang rendah. Rumus untuk menghitung MSE adalah :

$$\boldsymbol{MSE} = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}|f(x,y) - g(x,y)^2|$$

[3]

PSNR diukur dalam satuan 65decibel (dB). Untuk melakukan perhitungan PSNR, terlebih dahulu harus dicari nilai MSEnya. Kualitas citra dikatakan baik jika nilai PSNR semakin besar. Rumus PSNR adalah :

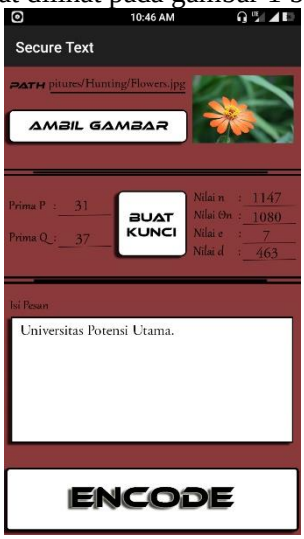
$$\boldsymbol{PSNR} = {}^{20}log\left(\frac{MAX}{\sqrt{MSE}}\right)$$

[3]

3. HASIL DAN PEMBAHASAN

Pesan yang digunakan merupakan kumpulan karakter yang terdapat pada tabel ASCII, dimana pesan tersebut kemudian di enkripsi dengan kriptografi algoritma RSA yang kemudian disisipkan kedalam citra gambar dan akan di ekstrak kembali dari citra gambar menjadi cipherteks dengan metode LSB yang selanjutnya akan di dekripsi menjadi plainteks. Citra gambar yang digunakan sebagai *cover object* adalah citra gambar dengan format file *.jpg dengan ukuran citra 3888x2592.

Berikut adalah menu encode dapat dilihat pada gambar 1 berikut ini :



Gambar 1. Tampilan Menu Encode

Pada *layout* encode kita dapat melakukan proses enkripsi dan *embedding* pesan. Sebagai contoh, pesan yang akan kita sisipkan adalah "UNIVERSITAS POTENSI UTAMA.". Pertama kita harus memilih gambar yang akan dijadikan sebagai *cover object* dengan cara klik *button* Ambil Gambar, maka sistem akan membuka *Gallery* untuk memilih gambar. Selanjutnya kita akan mencari terlebih dahulu pasangan kunci yang akan digunakan untuk proses encode dan decode dengan cara memasukkan 2 buah bilangan prima p dan q yang berbeda, dalam percobaan ini adalah 31 dan 37. Untuk proses perhitungan dapat dilihat sebagai berikut :

a. Mencari nilai n :
   $n = pxq$
   = 31 x 37
   = 1147
b. Mencari nilai Θn :
   $\Theta n = (p-1)x\ (q-1)$
   = (31-1) x (37-1)
   = (30) x (36)
   = 1080
c. Mencari nilai e :
   e = 2
   While Θn mod e ≠ 0
   e = e + 1
   End While

Proses 1 :
Θn mod e = 1080 mod 3
= 0
e = 3 + 1
e = 4
Proses 2 :
Θn mod e = 1080 mod 4
= 0
e = 4 + 1
e = 5
Proses 3 :
Θn mod e = 1080 mod 5
= 0
e = 5 + 1
e = 6
Proses 4 :
Θn mod e = 1080 mod 6
= 0
e = 6 + 1
e = 7
Proses 5 :
Θn mod e = 1080 mod 7
= 2
e = 7

d. Mencari nilai d :
$U_1 = 1$
$U_2 = 0$
$U_3 = Θn$
$V_1 = 0$
$V_2 = 1$
$V_3 = e$
While $V_3 = 0$
$Q = Int(U_3 / V_3)$
$N_1 = U_1 - (Q \times V_1)$
$N_2 = U_2 - (Q \times V_2)$
$N_3 = U_3 - (Q \times V_3)$

$U_1 = V_1$
$U_2 = V_2$
$U_3 = V_3$
$V_1 = N_1$
$V_2 = N_2$
$V_3 = N_3$
End While
Proses 1:
$Q = Int(U_3 / V_3)$
$= Int(1080 / 7)$
$= 154$
$N_1 = U_1 - (Q \times V_1)$
$= 1 - (154 \times 0)$
$= 1$
$N_2 = U_2 - (Q \times V_2)$
$= 0 - (154 \times 1)$
$= -154$
$N_3 = U_3 - (Q \times V_3)$

= 1080 - (154 x 7)
= 1080 - 1078
= 2
U_1 = 0
U_2 = 1
U_3 = 7
V_1 = 1
V_2 = -154
V_3 = 2
Proses 2:
Q = Int(U_3 /V_3)
= Int(7 / 2)
= 3
N_1 = U_1 - (Q x V_1)
= 0 - (3 x 1)
= -3
N_2 = U_2 - (Q x V_2)
= 1 - (3 x 154)
= 463
N_3 = U_3 - (Q x V_3)
= 7 - (3 x 2)
= 7 - 6
= 1
U_1 = 1
U_2 = -154
U_3 = 2
V_1 = -3
V_2 = 463
V_3 = 1
Proses 1:
Q = Int(U_3 /V_3)
= Int(2 / 1)
= 2
N_1 = U_1 - (Q x V_1)
= 1 - (2 x -3)
= 7
N_2 = U_2 - (Q x V_2)
= -154 - (2 x 463)
= -154 – 926
= -1080
N_3 = U_3 - (Q x V_3)
= 2 - (2 x 1)
= 2 - 2
= 0
U_1 = -3
U_2 = 463
U_3 = 1
V_1 = 7
V_2 = -1080
V_3 = 0

Maka diperoleh kunci RSA dengan nilai n = 1147, Θn = 1080, e = 7, dan d = 463. Selanjutnya akan melakukan proses enkripsi yaitu mengubah pesan menjadi bentuk yang tidak terbaca dengan rumus sebagai berikut :

$$Ci = P\frac{e}{i}\ mod\ n$$

Enkripsi Pertama :
U = 85
$C_i = P_i^e \bmod n$
$= 85^7 \bmod 1147$
$= 32057708828125 \bmod 1147$
$= 122$

Enkripsi Kedua :
n = 110
$C_i = P_i^e \bmod n$
$= 110^7 \bmod 1147$
$= 194871710000000 \bmod 1147$
$= 1035$

Enkripsi Ketiga :
i = 105
$C_i = P_i^e \bmod n$
$= 105^7 \bmod 1147$
$= 140710042265625 \bmod 1147$
$= 117$

Enkripsi Keempat :
v = 118
$C_i = P_i^e \bmod n$
$= 118^7 \bmod 1147$
$= 318547390056832 \bmod 1147$
$= 552$

Enkripsi Kelima :
e = 101
$C_i = P_i^e \bmod n$
$= 101^7 \bmod 1147$
$= 107213535210701 \bmod 1147$
$= 64$

Enkripsi Keenam :
r = 114
$C_i = P_i^e \bmod n$
$= 114^7 \bmod 1147$
$= 250226879128704 \bmod 1147$
$= 1003$

Enkripsi Ketujuh :
s = 115
$C_i = P_i^e \bmod n$
$= 115^7 \bmod 1147$
$= 266001988046875 \bmod 1147$
$= 548$

Enkripsi Kedelapan :
i = 105
$C_i = P_i^e \bmod n$
$= 105^7 \bmod 1147$
$= 140710042265625 \bmod 1147$

= 117

Enkripsi Kesembilan :
t = 116
C_i = P_i^e mod n
= 116^7 mod 1147
= 282621973446656 mod 1147
= 277

Enkripsi Kesepuluh :
a = 97
C_i = P_i^e mod n
= 97^7 mod 1147
= 80798284478113 mod 1147
= 791

Enkripsi Kesebelas :
s = 115
C_i = P_i^e mod n
= 115^7 mod 1147
= 266001988046875 mod 1147
= 548

Enkripsi Keduabelas :
[space] = 32
C_i = P_i^e mod n
= 32^7 mod 1147
= 34359738368 mod 1147
= 1055

Enkripsi Ketigabelas :
P = 80
C_i = P_i^e mod n
= 80^7 mod 1147
= 20971520000000 mod 1147
= 660

Enkripsi Keempatbelas :
o = 111
C_i = P_i^e mod n
= 111^7 mod 1147
= 207616015289871 mod 1147
= 629

Enkripsi Kelimabelas :
t = 116
C_i = P_i^e mod n
= 116^7 mod 1147
= 282621973446656 mod 1147
= 277

Enkripsi Keenambelas :
e = 101
C_i = P_i^e mod n
= 101^7 mod 1147

= 107213535210701 mod 1147
= 64

Enkripsi Ketujuhbelas :
n = 110
C_i = P_i^e mod n
= 110^7 mod 1147
= 194871710000000 mod 1147
= 1035

Enkripsi Kedelapanbelas :
s = 115
C_i = P_i^e mod n
= 115^7 mod 1147
= 266001988046875 mod 1147
= 548

Enkripsi Kesembilanbelas :
i = 105
C_i = P_i^e mod n
= 105^7 mod 1147
= 140710042265625 mod 1147
= 117

Enkripsi Keduapuluh :
[space] = 32
C_i = P_i^e mod n
= 32^7 mod 1147
= 34359738368 mod 1147
= 1055

Enkripsi Keduapuluhsatu :
U = 85
C_i = P_i^e mod n
= 85^7 mod 1147
= 32057708828125 mod 1147
= 122

Enkripsi Keduapuluhdua :
t = 116
C_i = P_i^e mod n
= 116^7 mod 1147
= 282621973446656 mod 1147
= 277

Enkripsi Keduapuluhtiga :
a = 97
C_i = P_i^e mod n
= 97^7 mod 1147
= 80798284478113 mod 1147
= 791

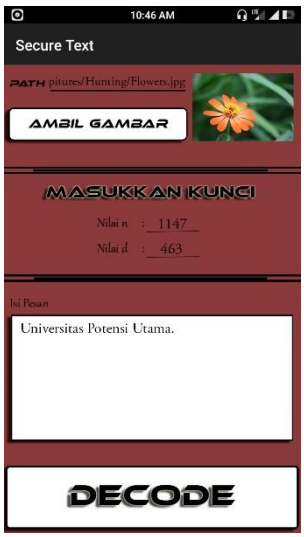
Enkripsi Keduapuluhempat :
m = 109
C_i = P_i^e mod n

= 109^7 mod 1147
= 182803912081669 mod 1147
= 1093

Enkripsi Keduapuluhlima :
a = 97
C_i = P_i^e mod n
= 97^7 mod 1147
= 80798284478113 mod 1147
= 791

Enkripsi Keduapuluhenam :
. = 46
C_i = P_i^e mod n
= 46^7 mod 1147
= 435817657216 mod 1147
= 1015

Setelah proses enkripsi selesai maka ditambahkan karakter "{" sebagai penanda pada setiap karakter. Maka didapatkan cipherteks dari hasil enkripsi algoritma RSA adalah "1 2 2 1 2 3 1 0 3 5 1 2 3 1 1 7 1 2 3 5 5 2 1 2 3 6 4 1 2 3 1 0 0 3 1 2 3 5 4 8 1 2 3 1 1 7 1 2 3 2 7 7 1 2 3 7 9 1 1 2 3 5 4 8 1 2 3 1 0 5 5 1 2 3 6 6 0 1 2 3 6 2 9 1 2 3 2 7 7 1 2 3 6 4 1 2 3 1 0 3 5 1 2 3 5 4 8 1 2 3 1 1 7 1 2 3 1 0 5 5 1 2 3 1 2 2 1 2 3 2 7 7 1 2 3 7 9 1 1 2 3 1 0 9 3 1 2 3 7 9 1 1 2 3 1 0 1 5". Setelah di enkripsi maka sistem akan menyisipkan cipherteks dengan mengubah setiap karakter menjadi biner dan disisipkan kedalam *pixel* citra gambar menggunakan steganografi LSB. Saat berhasil maka sistem akan memunculkan dialog pesan "Pesan telah di encode".



Gambar 3. Tampilan Menu Decode

Pada *layout* decode sistem akan melakukan proses ekstraksi dan dekripsi yang mana ekstraksi dilakukan terhadap citra gambar yang sebelumnya sudah disisipi cipherteks hasil enkripsi algoritma RSA. Pertama ambil gambar pada *button* Ambil Gambar yang kemudian sistem akan membuka *galery* untuk memilih gambar. Selanjutnya masukkan nilai n dan d yang sebelumnya sudah dibangkitkan ketika proses encode. Lalu klik Decode dan sistem akan mengekstraksi gambar yang sudah disisipkan gambar. Jika nilai n dan d benar, maka cipherteks yang dihasilkan sama dengan cipherteks hasil enkripsi. Sesuai dengan enkripsi sebelumnya, maka nilai n dan d adalah 1147 dan 463. Hasil ekstraksi citra gambar berupa cipherteks yaitu "1 2 2 1 2 3 1 0 3 5 1 2 3 1 1 7 1 2 3 5 5 2 1 2 3 6 4 1 2 3 1 0 0 3 1 2 3 5 4 8 1 2 3 1 1 7 1 2 3 2 7 7 1 2 3 7 9 1 1 2 3 5 4 8 1 2 3 1 0 5 5 1 2 3 6 6 0 1 2 3 6 2 9 1 2 3 2 7 7 1 2 3 6 4 1 2 3 1 0 3 5 1 2 3 5 4 8 1 2 3 1 1 7 1 2 3 1 0 5 5 1 2 3 1 2 2 1 2 3 2 7 7 1 2 3 7 9 1 1 2 3 1 0 9 3 1 2 3 7 9 1 1 2 3 1 0 1 5". Kemudian sistem akan memisahkan karakter

asli dengan karkter penanda sehingga didapatkan nilai asli cipherteks yaitu“122 1035 117 552 64 1003 548 117 277 791 548 1055 660 629 277 64 1035 548 117 1055 122 277 791 1093 791 1015” Setelah itu dilakukan proses dekripsi dengan rumus :

$$Pi = C\frac{d}{i}\ mod\ n$$

Kemudian hasil dari perhitungan nya yaitu P_i yang merupakan nilai desimal ASCII diubah menjadi bentuk karakter ASCII agar plainteks dapat dibaca kembali.

Dekripsi Pertama :

C_i = 122

P_i = C_i^d mod n

= 122^{463} mod 1147

= 9,651650840205358246697375194232 1e+965 mod 1147

= 85 = U

Dekripsi Kedua :

C_i = 1035

P_i = C_i^d mod n

= 1035^{463} mod 1147

= 8,267647545492804675936783871555 8e+1395 mod 1147

= 110 = n

Untuk hasil selanjutnya dapat dilihat pada tabel 1 berikut ini.

Tabel 1. Hasil Dekripsi ke 3 -26

	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
Hasil Deskripsi	i	v	e	r	s	i	t	a	s	space	p	o	t	e	n	s	i	space	u	t	a	m	a	46

Plainteks dari hasil dekripsi diatas adalah “Universitas Potensi Utama.”. Dengan demikian maka hasil perhitungan manual dengan menggunakan rumus algoritma RSA dengan hasil dari sistem adalah sama.

4. KESIMPULAN

Dari hasil penelitian yang telah dijabarkan diatas maka dapatlah diambil kesimpulan antara lain :

1. Perhitungan manual enkripsi dilakukan sebanyak 26 kali perhitungan dan deskripsi juga sama dan hasil perhitungan manual deskripsi dapat mengembalikan pesan yang sudah enkripsi seperti semula dalam bentuk plaintext.
2. Pesan yang akan disisip kedalam sebuah gambar berformat jpg terlebih dahulu di lakukan ke dalam ASCII dan setelah itu diubah kedalam bentuk biner dan disisip ke dalam pixel gambar rgb
3. Aplikasi ini dibangun dengan menggunakan android studio dan diaplikasikan kedalam handphone android.

5. SARAN

Ada beberapa saran yang diusulkan yaitu :

1. Diharapkan teknik penyisipannya dapat digunakan metode yang lain misalkan MSB
2. Dan pesan yang akan disisip disarankan dapat menggunakan word

DAFTAR PUSTAKA

[1] Rakhmat, B., & Fairuzabadi, M. (2010). Steganografi Menggunakan Metode Least Significant Bit Dengan Kombinasi Algoritma Kriptografi Vigenère Dan Rc4. *Jurnal Dinamika Informatika*, *5*(2), 1-17.

[2] Alvianto, A. R., & Darmaji, D. (2015). Pengaman Pengiriman Pesan Via SMS dengan Algoritma RSA Berbasis Android. *Jurnal Sains dan Seni ITS*, *4*(1), A1-A6..

[3] Arif, M. H., & Fanani, A. Z. (2016). Kriptografi Hill Cipher dan Least Significant Bit untuk Keamanan Pesan pada Citra. *CSRID (Computer Science Research and Its Development Journal)*, *8*(1), 60-72..

[4] Akbar, M. B., & Haryanto, E. V. (2018). Aplikasi Steganografi dengan Menggunakan Metode F5. *JUSITI: Jurnal Sistem Informasi dan Teknologi Informasi*, *4*(2), 165-176.

[5] Halder, R., Sengupta, S., Ghosh, S., & Kundu, D. (2016). A Secure Image Steganography Based on RSA Algorithm and Hash-LSB Technique. *IOSR Journal of Computer Engineering (IOSR-JCE)*, *18*(1).

[6] Apau, R., & Adomako, C. (2017). Design of Image Steganography based on RSA Algorithm and LSB Insertion for Android Smartphones. *International Journal of Computer Applications*, *164*(1), 0975-8887.

[7] Rambe, M. R., Haryanto, E. V., & Setiawan, A. (2018). Aplikasi Pengamanan Data dan Disisipkan Pada Gambar dengan Algoritma RSA Dan Modified LSB Berbasis Android. *Konferensi Nasional Sistem Informasi (KNSI) 2018*.

[8] Joko Dewanto, dkk, 2013. Pembuatan Aplikasi RSA Dengan Android. *Forum Ilmiah*, Vol. 10, No. 2.

[9] Gede Wisnu Bhaudhayana, dkk, 2015. Implementasi Algoritma Kriptografi AES 256 Dengan Metode Steganografi LSB Pada Gambar Bitmap. *Jurnal Ilmiah Komputer Universitas Udayana*. Vol. 8, No. 2.

[10] Haryanto, E. V. (2015). Penerapan Metode Adaptif Dalam Penyembunyian Pesan Pada Citra. Proceedings Konferensi Nasional Sistem dan Informatika

[11] Haryanto, E. V. (2015). Perbandingan Metode Robinson 5 Level Dan Prewit Dalam Mendeteksi Tepi Citra Digital. *Proceedings Konferensi Nasional Sistem dan Informatika (KNS&I)*. (KNS&I).