

IMPLEMENTASI STEGANOGRAFI *TEXT TO IMAGE* MENGGUNAKAN METODE *ONE BIT LEAST SIGNIFICANT BIT* BERBASIS ANDROID

Aditya Aziz Fikhri¹, Hendrawaty²

^{1,2}Program Studi Teknik Informatika, Jurusan Teknologi Informasi dan Komputer, Politeknik Negeri Lhokseumawe, Jalan Banda Aceh-Medan Km.280,3 Buketrata, Lhoksueumawe, 24301 PO.BOX 90 Telepone (0645) 42670 Fax 42785, Indonesia

E-mail: aditfreedom11@gmail.com¹, waty.hendra@gmail.com².

Abstrak

Informasi merupakan hal yang sangat penting dan berharga, dewasa ini informasi sering dipertukarkan melalui internet. Kemajuan teknologi *smartphone* terutama *smartphone* Android, memungkinkan penggunanya saling berkirim data/informasi melalui internet dengan mudah dan cepat menggunakan beberapa aplikasi yang ada pada *smartphone* seperti *WhatsApp*, *E-Mail*, *Telegram*, dan *Facebook*. Data/informasi yang dikirim melalui internet masih rawan terhadap pencurian dan penyadapan. Oleh karena itu dibutuhkan cara untuk mengamankan data/informasi yang akan dikirim. Steganografi merupakan salah satu teknik yang dapat digunakan untuk mengamankan data/informasi. Steganografi adalah teknik yang digunakan untuk menyembunyikan pesan kedalam suatu media seperti gambar, *audio* dan *video*. Penelitian ini membahas tentang implementasi steganografi pada *smartphone* Android yang dapat digunakan menyembunyikan pesan teks ke dalam media gambar RGB 24 bit (*cover image*), dan dapat juga digunakan untuk mengambil kembali pesan teks dari gambar RGB 24 bit yang telah disisipi (*Stego image*). Media Gambar yang digunakan sebagai *input (cover image)* berformat JPEG dan PNG. Algoritma yang digunakan untuk penyembunyian dan pengambilan pesan teks ke dan dari gambar adalah *One bit Least Significant Bit*. Dari 5 sampel pesan teks dan 5 *cover image*, penyembunyian pesan teks kedalam *cover image* berhasil dengan baik, dengan tingkat keberhasilan 100% dan waktu rata-rata penyisipan yaitu 0,009 detik. Dari 5 sampel *stego image* yang telah bersisi pesan teks, pengambilan pesan juga berhasil dengan tingkat keberhasilan 100 % dan waktu rata-rata ekstraksi yaitu 0,0036 detik. Waktu yang digunakan untuk ekstraksi lebih cepat dibandingkan dengan penyisipan, output gambar yang dihasilkan (*stego image*) berformat PNG, dengan ukuran 900 x 900 *pixel*.

Kata Kunci: Steganografi, *Stego Image*, *One Bit Least Significant Bit*, *Android*

1. Pendahuluan

Informasi merupakan hal yang sangat penting dan berharga, dewasa ini informasi sering dipertukarkan melalui internet sehingga rentan terhadap pencurian informasi. Banyak jenis informasi yang menggunakan pengamanan khusus seperti pin, *password*, dan data diri. Informasi tersebut tidak boleh diketahui oleh pihak lain karena dapat memberikan kerugian kepada pemilik informasi.

Dengan hadirnya *smartphone* android pada saat ini, Memungkinkan penggunanya untuk berbagi informasi dengan mudah dan cepat melalui internet. Salah satu cara pengguna android berbagi informasi yaitu dengan media *chatting*, yakni mengirim pesan secara pribadi dari satu pengguna ke pengguna lainnya menggunakan aplikasi Android seperti *WhatsApp*, *Facebook*, *Telegram* dan *E-Mail*. Sebagai contoh yaitu memberi informasi yang sangat penting dan rahasia seperti *password* sebuah akun media sosial atau memberikan informasi pin sebuah brankas dengan media *chatting* tersebut. Tetapi kenyataan yang terjadi saat ini, isi sebuah *chat* pun dapat diketahui orang

lain karena kemungkinan untuk disadap atau diretas. Di era *modern* komunikasi digital, ada beberapa teknik yang digunakan untuk menyembunyikan informasi dalam media tertentu. Salah satu teknik tersebut adalah *steganography*[1]

Steganography merupakan teknik yang banyak digunakan untuk menyembunyikan keberadaan informasi (message). *Steganography* mencegah pihak yang tidak diinginkan mencurigai bahwa ada informasi yang disembunyikan[2]. Dengan menggunakan Steganografi, pesan teks yang bersifat rahasia dapat disembunyikan ke dalam sebuah gambar sebelum dikirim melalui internet. Walaupun gambar yang telah disisipi pesan rahasia tersebut berhasil dicuri oleh pihak lain, tetapi pesan teks yang ada didalamnya tetap tidak dapat diketahui. Pihak pencuri informasi tidak akan menyadari ada pesan rahasia pada gambar, selama gambar tersebut tidak mengundang kecurigaan.

Untuk mengamankan pesan yang akan dikirimkan melalui internet dengan menggunakan aplikasi *chatting* pada *smartphone* android, maka dibuat sebuah aplikasi steganografi berbasis android yang berfungsi untuk

menyembunyikan pesan teks kedalam gambar dan berfungsi untuk mengekstrak gambar untuk mendapatkan kembali pesan teks yang telah disembunyikan. Media Gambar yang digunakan sebagai input (*cover image*) berformat JPEG dan PNG karena kedua format gambar tersebut sering dipertukarkan melalui internet. Output gambar yang dihasilkan (*stego image*) berformat PNG, dengan ukuran 900 x 900 pixel dikarenakan ukuran tersebut merupakan salah satu ukuran yang banyak dipertukarkan di internet dan beberapa *smartphone* yang digunakan dalam pembuatan aplikasi tidak dapat membuka ukuran gambar yang sangat besar. Algoritma yang digunakan untuk penyembunyian dan pengambilan pesan teks ke dan dari gambar adalah *One Bit Least Significant Bit*. Aplikasi steganografi yang dibuat dapat berjalan pada *smartphone android* 4.2 dan di atasnya.

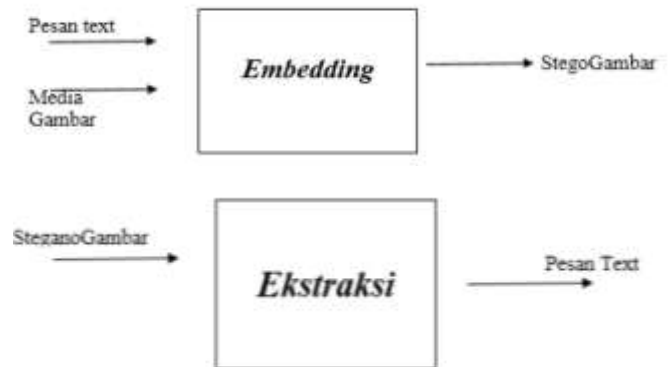
Dengan adanya aplikasi steganografi *one bit least significant bit* berbasis android ini dapat membantu mengamankan pesan teks yang akan dikirimkan melalui internet dengan menggunakan aplikasi yang ada pada *smartphone android* seperti *WhatsApp*, *E-Mail*, dan *Telegram*.

2. Landasan Teori

A. Steganography

Steganography berasal dari dua bahasa Yunani “*steganos*” yang artinya tersembunyi dan “*graphos*” yang artinya menulis. Steganografi sering dimaksudkan sebagai *secret writing* atau *data hiding*[3]. Tujuan utama *steganography* adalah untuk meningkatkan keamanan komunikasi dengan menyisipkan pesan rahasia ke dalam gambar digital, memodifikasi pixel-pixel yang tidak berarti dari gambar digital tersebut[4]. Dengan kata lain Steganografi adalah seni dan ilmu menulis atau menyembunyikan pesan dengan suatu cara sehingga selain *sender* dan *receiver*, tidak ada seorangpun yang mengetahui atau menyadari bahwa ada suatu pesan rahasia[5].

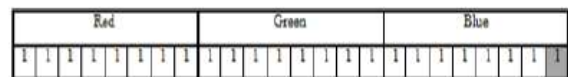
Terdapat dua proses utama dalam Steganografi yaitu penyisipan (*embedding*) dan penguraian (*extraction*). Proses *embedding* yaitu proses untuk menyisipkan pesan ke dalam media *cover* sedangkan proses *extracting* yaitu proses menguraikan pesan pada media *stego*. Gambar 1. Memperllihatkan proses penyisipan pesan *Text* kedalam media gambar, dimana output yang dihasilkan dari proses tersebut adalah sebuah *stegoGambar* (gambar yang telah disisipi pesan *Text*). Dan pada Gambar 1 juga dapat dilihat proses ekstraksi pesan *Text* dari *stegiGambar*, dimana output dari proses tersebut adalah pesan *text*.



Gambar 1. Proses penyisipan dan ekstraksi data pada steganografi

B. One bit LSB

Suatu metode yang terkenal dan sederhana pada steganografi adalah menyembunyikan data rahasia ke *least-significant bit (LSB)* dari setiap pixel dalam sebuah gambar[6]. *Stego One bit LSB* merupakan jenis dari metode LSB yang menyisipkan data pada satu bit LSB pada setiap pixelnya[7], seperti yang terlihat pada Gambar 2. Dengan metode tersebut dibutuhkan 8 pixel dari gambar RGB 24 bit untuk menyisipkan pesan teks yang panjangnya 1 karakter saja (satu karakter terdiri dari 8 bit). Bit pada gambar RGB 24 bit yang mengalami perubahan adalah bit pada warna biru saja, dengan jumlah 1 lebih tinggi atau 1 lebih rendah nilainya. Sehingga metode ini dapat membuat *Stego Image* terlihat lebih kecil serta lebih halus perubahannya saat setelah bit pesan disisipkan kedalam *Cover Image*. Gambar 2. memperlihatkan cara penyisipan dengan menggunakan *Stego One bit Least Significant Bit (Stego One Bit LSB)* [7].



Gambar Stego One Bit LSB

Sebagai ilustrasi, pada Gambar 3. diperlihatkan urutan bit sebanyak 3 piksel dari *cover image* RGB 24 bit.

R	G	B
00100111	10100111	00100110
00100111	10100111	00100110
00100111	10100111	00100110

Gambar 3. Bit cover image

Misalkan bit pesan yang akan disisipkan yaitu **110**, maka dengan menggunakan metode *One bit LSB* akan dihasilkan urutan bit pada *stego image* seperti yang tampak pada Gambar 4.

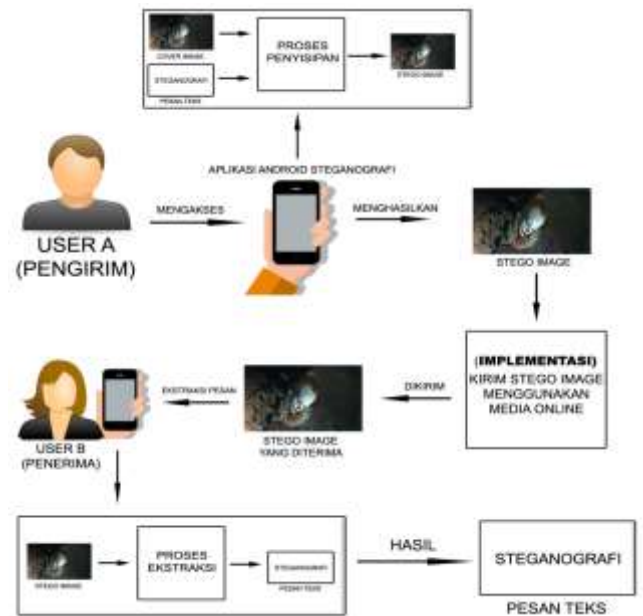
R	G	B
(00100111)	10100111	00100110
(00100111)	10100111	00100111
(00100111)	10100111	00100111

Gambar 4. Bit *stego image*

Dari Gambar 4 dapat dilihat bahwa bit yang tercetak tebal menunjukkan bit pesan yang telah disisipi ke dalam cover image (Gambar 3). Dari gambar tersebut juga dapat dilihat bahwa setiap bit pesan akan disisipi pada bit terkecil yang ada di setiap pixel. Dengan demikian untuk menyisipkan pesan yang panjangnya 3 bit (seperti contoh), maka diperlukan 3 pixel. Dan untuk menyisipkan 1 byte pesan, maka diperlukan 8 pixel.

2. Metode Penelitian

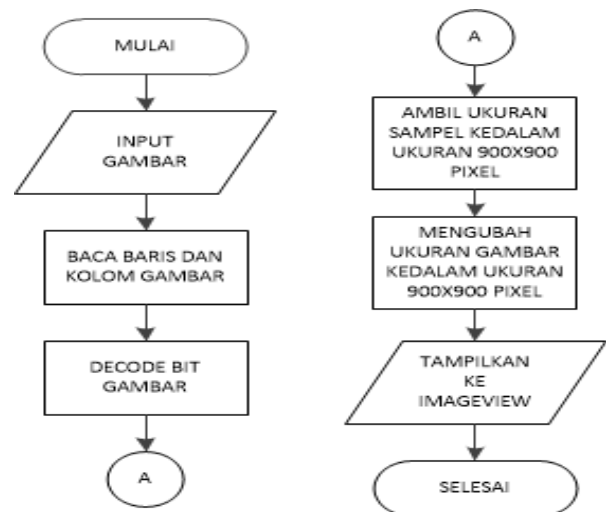
Ilustrasi keseluruhan dari sistem yang dibangun dapat dilihat pada Gambar 5. pada gambar tersebut dapat dilihat bahwa *User A* (pengirim) mengakses aplikasi Steganografi *One Bit LSB* yang terdapat pada *smartphone android user A* untuk menyembunyikan pesan *text* kedalam media gambar (*cover image*). *Output* dari proses penyisipan tersebut adalah sebuah gambar yang didalamnya sudah berisi pesan *text* (*stego image*). Kemudian *stego image* tersebut dikirimkan kepada *User B* (penerima) melalui media *online* seperti *Whatsapp*, *Telegram*, atau *E-mail*. *User B* (penerima) yang telah menerima *stego image* kemudian mengakses aplikasi Steganografi *One bit LSB* yang ada pada *smartphone android User B* untuk mengekstraksi pesan *text* yang terdapat di dalam *stego image*. *Output* dari proses ekstraksi yang dilakukan oleh *User B* tersebut berupa pesan *text* yang sebelumnya telah disisipi ke dalam *cover image* oleh *User A* (pengirim).



Gambar 2. Ilustrasi Sistem Secara Keseluruhan

A. Flowchart normalisasi Cover Image

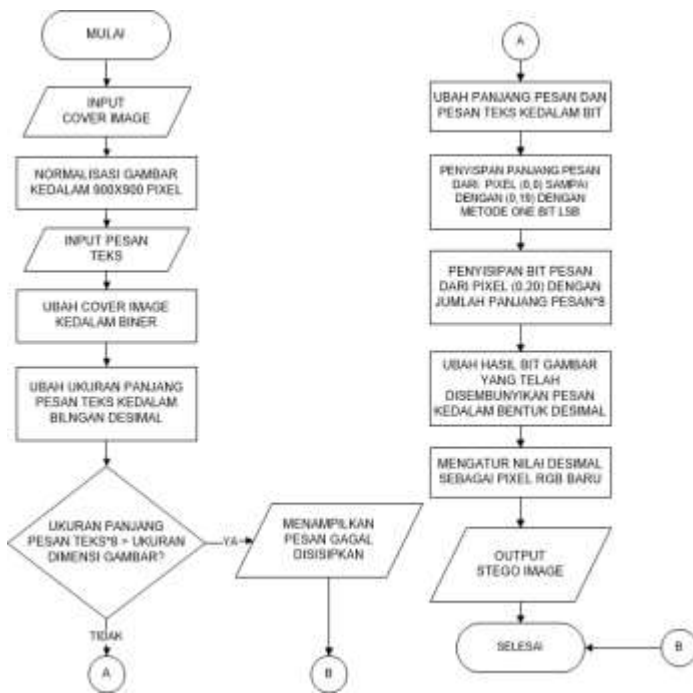
Perancangan normalisasi *cover image* pada penelitian ini digambarkan dengan *flowchart* pada Gambar 6.



Gambar 3. Flowchart normalisasi *cover image*

B. Flowchart Penyisipan Pesan Text

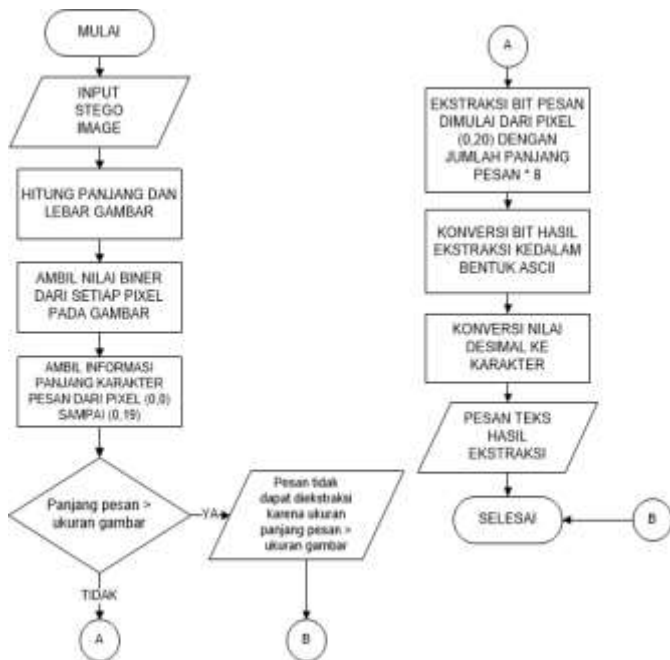
Perancangan proses penyisipan pesan *text* kedalam *cover image* pada penelitian ini digambarkan dengan *flowchart* pada Gambar 7.



Gambar 4. Flowchart proses penyisipan pesan teks kedalam cover image

C. Flowchart Ekstraksi Pesan text

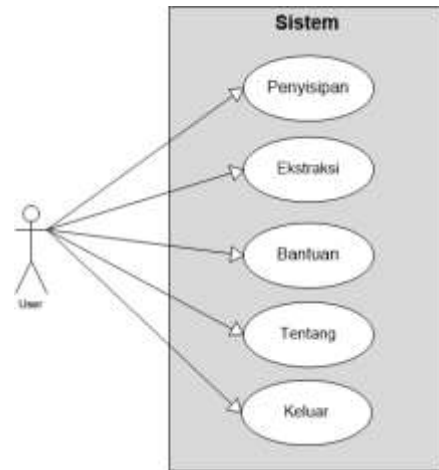
Perancangan proses ekstraksi pesan teks dari stego image pada penelitian ini digambarkan dengan flowchart pada Gambar 8.



Gambar 5. Flowchart ekstraksi pesan dari stego image

D. Perancangan Use case diagram

Perancangan Use case diagram dari Aplikasi steganografi pada penelitian ini dapat dilihat pada Gambar 9.





Gambar 6. Use case diagram Aplikasi Steganografi









3. Hasil dan Pembahasan

Pada tahap ini dilakukan pengujian aplikasi yang sudah dirancang sebelumnya seperti tampilan *User Interface*, proses penyisipan pesan teks kedalam gambar RGB 24-bit (*cover image*), ekstraksi pesan teks dari gambar RGB 24-bit (*stego image*) yang telah mengalami proses pengiriman melalui media *online*.

Tabel 1 memperlihatkan hasil proses penyisipan pesan teks dengan panjang pesan yang bervariasi mulai dari yang pendek hingga pesan yang panjang kedalam gambar (*cover image*) RGB 24-bit yang ukurannya juga bervariasi, kemudian dinormalisasi dengan ukuran 900x900 pixel.

Tabel 1. Hasil penyisipan pesan teks kedalam cover image

No	Gambar Input	Panjang Pesan Yang Disisipkan	Gambar Output	Lama proses
1	 1024x575 pixel 17,38 KB Format JPEG	A (1 karakter)	 900x900 pixel 4.46 KB Format PNG	0.005 detik

2	 900x900 pixel 439 KB Format PNG	AA (2 karakter)	 900x900 pixel 347 KB Format PNG	0.006 detik
3	 900x900 pixel 96.6 KB Format JPG	Hai (3 Karakter)	 900x900 pixel 347 KB Format PNG	0.006 detik
4	 300x300 pixel 44.1 KB Format JPG	SAYA (4 Karakter)	 900x900 pixel 166 KB Format PNG	0.007 detik
5	 300x300 pixel 131 KB Format PNG	politeknik negeri lhokseumawe (29 karakter)	 900x900 pixel 166 KB Format PNG	0.021 detik

Pengujian penyisipan panjang pesan teks merupakan penyisipan pertama yang dilakukan sistem dalam proses penyisipan yang dimulai dari pixel (1,1) sampai dengan (1,20), sebagai pembahasan diambil satu sampel hasil penyisipan pada table 1 yaitu nomor 1. Pesan yang diinputkan yaitu "A", dengan panjang karakter yaitu 1 karakter = 00000000000000000001₍₂₎. Nilai RGB pada gambar sebelum disisipi panjang pesan :

	R	G	B
Pixel (1,1)	10011010	11011001	11101010
Pixel (1,2)	10011010	11011001	11101010
Pixel (1,3)	10011010	11011001	11101010
Pixel (1,4)	10011010	11011001	11101010
Pixel (1,5)	10011010	11011001	11101010
Pixel (1,6)	10011010	11011001	11101010

Pixel (1,7)	10011010	11011001	11101010
Pixel (1,8)	10011010	11011001	11101010
Pixel (1,9)	10011010	11011001	11101010
Pixel (1,10)	10011010	11011001	11101010
Pixel (1,11)	10011010	11011001	11101010
Pixel (1,12)	10011010	11011001	11101010
Pixel (1,13)	10011010	11011001	11101010
Pixel (1,14)	10011010	11011001	11101010
Pixel (1,15)	10011010	11011001	11101010
Pixel (1,16)	10011010	11011001	11101010
Pixel (1,17)	10011010	11011001	11101010
Pixel (1,18)	10011010	11011001	11101010
Pixel (1,19)	10011010	11011001	11101010
Pixel (1,20)	10011010	11011001	11101010

Nilai RGB pada gambar setelah disisipi panjang pesan :

	R	G	B
Pixel (1,1)	10011010	11011001	11101011
Pixel (1,2)	10011010	11011001	11101010
Pixel (1,3)	10011010	11011001	11101010
Pixel (1,4)	10011010	11011001	11101010
Pixel (1,5)	10011010	11011001	11101010
Pixel (1,6)	10011010	11011001	11101010
Pixel (1,7)	10011010	11011001	11101010
Pixel (1,8)	10011010	11011001	11101010
Pixel (1,9)	10011010	11011001	11101010
Pixel (1,10)	10011010	11011001	11101010
Pixel (1,11)	10011010	11011001	11101010
Pixel (1,12)	10011010	11011001	11101010
Pixel (1,13)	10011010	11011001	11101010
Pixel (1,14)	10011010	11011001	11101010
Pixel (1,15)	10011010	11011001	11101010
Pixel (1,16)	10011010	11011001	11101010
Pixel (1,17)	10011010	11011001	11101010
Pixel (1,18)	10011010	11011001	11101010
Pixel (1,19)	10011010	11011001	11101010
Pixel (1,20)	s10011010	11011001	11101010

Dari hasil perhitungan proses penyisipan panjang pesan dapat diambil kesimpulan bahwa nilai dari 1 karakter = 00000000000000000001₍₂₎ telah disisipi pada bit LSB pada warna biru dimulai dari pixel (1,1) hingga pixel (1,20). Nilai desimal R dan G setelah proses penyisipan tidak ada perubahan, sedangkan nilai B berubah satu lebih tinggi apabila yang disisipkan nilai 1 dan apabila nilai yang disisipkan yaitu 0 maka nilai desimal dari pixel B tidak ada perubahan. Dengan demikian dapat dibuktikan bahwa proses penyisipan panjang pesan pada aplikasi sudah berjalan dengan benar.

Pengujian penyisipan pesan teks merupakan penyisipan yang dilakukan sistem setelah proses penyisipan panjang pesan yang dimulai dari pixel (1,21) sampai dengan (1,28) dengan pesan yang diinputkan yaitu "A" dengan bilangan ASCII yaitu 41₍₁₆₎ = 01000001₍₂₎. Nilai RGB pada gambar sebelum disisipi pesan teks :

	R	G	B
Pixel (1,21)	10011010	11011001	11101010
Pixel (1,22)	10011010	11011001	11101010
Pixel (1,23)	10011010	11011001	11101010
Pixel (1,24)	10011010	11011001	11101010
Pixel (1,25)	10011010	11011001	11101010
Pixel (1,26)	10011010	11011001	11101010
Pixel (1,27)	10011010	11011001	11101010
Pixel (1,28)	10011010	11011001	11101010



Nilai RGB pada gambar setelah disisipi pesan teks:

	R	G	B
Pixel (1,21)	10011010	11011001	11101011
Pixel (1,22)	10011010	11011001	11101010
Pixel (1,23)	10011010	11011001	11101010
Pixel (1,24)	10011010	11011001	11101010
Pixel (1,25)	10011010	11011001	11101010
Pixel (1,26)	10011010	11011001	11101010
Pixel (1,27)	10011010	11011001	11101011
Pixel (1,28)	10011010	11011001	11101010

Dari Tabel 1 dapat dilihat bahwa pada saat proses penyisipan pesan, sistem akan menormalisasi ukuran gambar kedalam ukuran 900x900 pixel. Ukuran pesan yang pendek membutuhkan waktu lebih cepat dibandingkan pesan yang lebih panjang. Dari 5 sampel proses penyisipan diperoleh rata-rata waktu penyisipan $(0.005 + 0.006 + 0.006 + 0.007 + 0.021) / 5 = 0.009$ detik.

Hasil pengujian proses ekstraksi menunjukkan bahwa semua *stego image* berhasil diekstraksi. pesan teks hasil ekstraksi sama persis seperti pesan teks saat disisipi. Terdapat 5 sampel yang digunakan sebagai *stego image* seperti pada Tabel 2. Proses ekstraksi pesan teks dimulai dari mengekstraksi informasi panjang pesan teks terlebih dahulu dimulai dari pixel (1,1) sampai dengan pixel (1,20) dan kemudian dilanjutkan dengan ekstraksi pesan teks dari pixel (1,21) sampai dengan ukuran panjang pesan teks yang dikalikan dengan 8. Tabel 2 merupakan hasil dari ekstraksi pesan teks dari *stego image*.

Tabel 2 Hasil ekstraksi pesan teks dari *stego image*

No	Stego Image	Hasil ekstraksi	Lama proses
1	 900x900 pixel, 4.46 KB Format PNG	A (1 karakter)	0.003 detik
2		AA (2 karakter)	0.003 detik

	900x900 pixel, 347 KB Format PNG		
3	 900x900 pixel, 347 KB Format PNG	Hai (3 Karakter)	0.003 detik
4	 900x900 pixel, 166 KB Format PNG	SAYA (4 Karakter)	0.004 detik
5	 900x900 pixel, 166 KB Format PNG	politeknik negeri lhokseumawe (29 karakter)	0.005 detik

Ekstraksi panjang pesan teks merupakan ekstraksi pertama yang dilakukan sistem dalam proses ekstraksi. Pembahasan ini menggunakan sampel pada tabel 2 yaitu nomor 1. Pada pembahasan ini akan diperlihatkan nilai pixel dari gambar secara perhitungan manual. Informasi panjang pesan yang diekstrak yaitu dari pixel (1,1) sampai dengan pixel (1,20).

Nilai biner RGB 24-bit dari pixel *stego image* :

	R	G	B
Pixel (1,1)	10011010	11011001	11101011
Pixel (1,2)	10011010	11011001	11101010
Pixel (1,3)	10011010	11011001	11101010
Pixel (1,4)	10011010	11011001	11101010

Pixel (1,5)	10011010	11011001	11101010
Pixel (1,6)	10011010	11011001	11101010
Pixel (1,7)	10011010	11011001	11101010
Pixel (1,8)	10011010	11011001	11101010
Pixel (1,9)	10011010	11011001	11101010
Pixel (1,10)	10011010	11011001	11101010
Pixel (1,11)	10011010	11011001	11101010
Pixel (1,12)	10011010	11011001	11101010
Pixel (1,13)	10011010	11011001	11101010
Pixel (1,14)	10011010	11011001	11101010
Pixel (1,15)	10011010	11011001	11101010
Pixel (1,16)	10011010	11011001	11101010
Pixel (1,17)	10011010	11011001	11101010
Pixel (1,18)	10011010	11011001	11101010
Pixel (1,19)	10011010	11011001	11101010
Pixel (1,20)	10011010	11011001	11101010

Nilai biner panjang pesan yang diekstraksi dari *stego image* :
000000000000000000001₍₂₎ = 1₍₁₀₎ karakter

Ekstraksi pesan teks merupakan ekstraksi yang dilakukan sistem setelah proses ekstraksi panjang pesan, jumlah panjang karakter yang telah diekstraksi yaitu sebanyak 1 karakter dengan proses ekstraksi sebanyak $1 * 8 = 8$ kali. Pixel pada *stego image* yang akan diekstraksi setelah didapatkan informasi panjang pesan yaitu pixel (1,21) sampai dengan (1,28).







Nilai pixel RGB pada *stego image* :

	R	G	B
Pixel (1,21)	10011010	11011001	11101011
Pixel (1,22)	10011010	11011001	11101010
Pixel (1,23)	10011010	11011001	11101010
Pixel (1,24)	10011010	11011001	11101010
Pixel (1,25)	10011010	11011001	11101010
Pixel (1,26)	10011010	11011001	11101010
Pixel (1,27)	10011010	11011001	11101011
Pixel (1,28)	10011010	11011001	11101010

Nilai biner panjang pesan yang diekstraksi dari *stego image* :
01000001₍₂₎ = 41₍₁₆₎ ASCII = karakter A

Dari Tabel 2. dan dari pembahasan yang dilakukan dapat dilihat bahwa proses ekstraksi dari *stego image* pada penelitian ini telah berhasil dengan baik. Waktu rata-rata yang diperoleh untuk 5 proses ekstraksi berdasarkan Tabel 2. adalah $(0.003 + 0.003 + 0.003 + 0.004 + 0.005) / 5 = 0.0036$ detik.

Tabel 3. memperlihatkan hasil ekstraksi *stego image* yang telah dikirim melalui beberapa media *online* seperti *Telegram*, *Whatsapp*, dan *E-mail*.

No	Stego Image Sebelum Dikirimkan	Stego Image Setelah Dikirimkan	Hasil ekstrak pesan	Keterangan
1	 900x900 pixel 75.2 KB Format PNG	 900x900 pixel 73.4 KB Format PNG	PNL 2014	Dikirim lewat: <i>Telegram</i> Pesan Berhasil diekstrak
2	 900x900 pixel 5.28 KB Format PNG	 900x900 pixel 5.28 KB Format PNG	kode loker : 1224	Dikirim lewat: <i>WhatsApp</i> Pesan Berhasil diekstrak
3	 900x900 pixel 161 KB Format PNG	 900x900 pixel 157 KB Format PNG	Nanggr oe Aceh Daruss alam	Dikirim lewat: <i>E-mail</i> Pesan Berhasil diekstrak

Tabel 3. Hasil ekstraksi dari *stego image* yang dikirim melalui media *online*

Dari hasil yang tampak pada Tabel 3, dapat dilihat bahwa *stego image* (yang berisi pesan teks) yang dikirimkan lewat internet dengan menggunakan aplikasi *Telegram*, *Whatsapp*, dan *E-mail* dapat diekstrak dengan baik oleh penerima, dimana pesan hasil ekstraksi yang diperoleh penerima, sama persis dengan pesan teks yang sebelumnya disisipi oleh pengirim ke dalam *cover image*.

4. Simpulan

Setelah melakukan penelitian dan pembahasan mengenai implementasi steganografi *text to image* menggunakan metode *one bit least significant bit* berbasis *android*, dapat diambil simpulan bahwa:

1. Aplikasi dapat dijalankan dengan baik, karena sudah berhasil melakukan proses penyisipan pesan teks ke *cover image* dan ekstraksi pesan teks dari *stego image*.
2. Penyisipan berbagai macam pesan teks ke dalam 5 buah *cover image* dengan normalisasi 900x900 pixel semuanya berhasil dilakukan, sehingga tingkat keberhasilan proses penyisipan pesan teks dari sampel yang dilakukan adalah 100 %, dengan rata-rata waktu penyisipan 0,009 detik.
3. Ukuran file gambar (*stego image*) yang dihasilkan dari proses penyisipan lebih besar jika ukuran *cover image* tersebut lebih kecil dari ukuran normalisasi gambar dan sebaliknya.
4. Proses ekstraksi pesan teks dari 5 buah *stego image* semuanya berhasil dilakukan, tingkat keberhasilan proses ekstraksi pesan teks dari sampel yang dilakukan

adalah 100 %, dengan rata-rata waktu ekstraksi 0,0036 detik.

5. *Stego image* yang dikirimkan lewat internet melalui *Whatsapp*, *Telegram*, dan *E-mail* dapat diekstrak dengan sempurna oleh pihak penerima.
6. Aplikasi Steganografi *One Bit Least Significant Bit* pada penelitian ini sudah berjalan dengan baik pada *smartphone* Evercoss A74D dengan Android versi 4.4 dan *smartphone* Xiaomi Redmi 4A dengan Android versi 7.12.

5. Daftar Pustaka

- [1] M. M Amin, M. Salleh, S. Ibrahim, M.R.K atmin, and M.Z.I. Shamsuddin “Information Hiding using Steganography” 4* National Conference on Telecommunication Technology Proceedings, Shah Alam, Malaysia. 2003 IEEE.
- [2] Westfeld, A., and G. Wolf, Steganography in a Video conferencing system, in proceedings of the second international workshop on information hiding, vol. 1525 of lecture notes in computer science, Springer, 1998. pp. 32-47.
- [3] Moerland, T., “Steganography and Steganalysis”, Leiden Institute of Advanced Computing Science, www.liacs.nl/home/tmoerl/privtech.pdf
- [4] Feng, J.B., Lin, I.C., Tsai, C.S., Chu, Y.P., 2006. Reversible watermarking: current status and key issues. *International Journal of Network Security* 2 (May), 161– 170.
- [5] Martono, & Irawan. (2013). Penggunaan Steganografi dengan Metode End of File (EOF) pada Digital Watermarking. *Jurnal TICOM*.
- [6] Deepesh Rawat, Vijaya Bhandar, Steganography Technique for Hiding Text Information in Color Image using Improved LSB Method. *International Journal of Computer Applications* (0975 – 8887) Volume 67– No.1, April 2013
- [7] Por, L. Y., Beh, D., Ang, T. F., & Ong, S. Y. (2013). An Enhanced Mechanism for Image Steganography Using Sequential Colour Cycle Algorithm . *The International Arab Journal of Information Technology*.1