Analisis Penggunaan Steganografi pada Foto di Aplikasi Pinterest

Natasha Tiovanny Silaban 18220101
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
natashatiovanny@gmail.com

***Abstract*— Steganografi adalah teknik yang digunakan untuk menyembunyikan informasi sensitif dalam media yang tampak biasa. Penelitian ini bertujuan untuk menganalisis penggunaan steganografi pada foto di aplikasi Pinterest menggunakan metode *least significant byte*. Analisis dari pengujian yang dilakukan menunjukkan bahwa gambar yang telah di-encode dan diunggah ke Pinterest masih memiliki pesan rahasia saat dilakukan *decode*. Penelitian ini memberikan wawasan yang lebih baik tentang penggunaan steganografi pada foto di Pinterest dan dapat menjadi dasar untuk pengembangan metode analisis yang lebih komprehensif, pengembangan aplikasi yang lebih baik, serta penelitian tentang aspek keamanan dan privasi terkait.Ke depannya, diharapkan pengembangan solusi dari pengecekan steganografi pada setiap foto yang akan diunggah ke internet agar tidak ada pihak yang dirugikan.**

Keywords—steganografi, pinterest, least significant byte, encode, decode

I. PENDAHULUAN

Pada masa yang serba digital ini, media sosial menjadi bagian yang penting dalam kehidupan sehari-hari. Media sosial memberikan kesempatan kepada penggunanya untuk berinteraksi dan berbagi cerita menggunakan berbagai jenis konten, seperti foto dan video. Salah satu aplikasi media sosial yang sering digunakan adalah Pinterest. Pinterest adalah sebuah *platform* yang memperbolehkan pengguna untuk menemukan, menyimpan, dan membagikan inspirasi melalui foto. Dengan jumlah pengguna aktif setiap hari yang tidak sedikit, Pinterest telah menjadi sumber inspirasi yang tidak terhitung bagi

Namun, pertumbuhan pesat dari sosial media dan banyaknya foto yang diunggah dan diunduh pada platform seperti Pinterest menimbulkan sebuah masalah baru mengenai foto-foto tersebut. Salah satu masalahnya adalah foto-foto tersebut dapat menjadi sebuah media dalam menyimpan pesan tersembunyi atau informasi rahasia. Bagi masyarakat awam yang tidak peka terhadap keberadaan pesan rahasia tersebut, pengunduhan foto dapat menimbulkan masalah lain seperti, penyusupan malware yang disebut dengan istilah stegosploit.

Steganografi, sebagai cabang dari keamanan informasi, adalah teknik yang bertujuan menyembunyikan pesan atau data rahasia di dalam media yang tampaknya biasa. Dalam konteks aplikasi Pinterest, siapa saja mungkin menggunakan steganografi untuk menyisipkan pesan tersembunyi dalam foto-foto mereka, yang hanya dapat diakses oleh siapapun yang mengunduhnya.

Pada makalah ini, akan dikumpulkan sampel foto-foto yang diunggah di aplikasi Pinterest dan dianalisis mengenai bukti penggunaan steganografi dalam foto-foto tersebut.

II. DASAR TEORI

A. Steganografi

Steganografi berasal dari Bahasa Yunani yaitu *steganos* dan *graphien*. *Steganos* berarti tersembunyi dan *graphien* yang berarti tulisan. Berdasarkan gabungan kedua kata tersebut, steganografi berarti tulisan tersembunyi. Dilansir dari *website* KBBI, steganografi memiliki arti praktik menyembunyikan pesan, gambar, video, atau berkas lainnya di dalam pesan, gambar, video, atau berkas lain. Steganografi sendiri ada dengan tujuan untuk menyembunyikan pesan atau data sensitif dalam suatu media tanpa menimbulkan kecurigaan bagi pihak yang melihatnya.

Steganografi ditulis oleh Herodatus pada tahun 485 – 525 sebelum masehi. Ia merupakan seorang sejarawan Yunani pada tahun 440 BC di dalam bukunya yang mengisahkan perang antara kerajaan Persia dan rakyat Yunani. Herodatus menceritakan bagaimana mereka memanfaatkan kepala budak sebagai media dalam menyampaikan pesan.

Pada zaman yang semakin maju ini, terdapat teknik steganografi digital yang memungkinkan penyembunyian pesan digital di dalam media digital seperti gambar, video, audio, atau teks. Istilah yang digunakan untuk media yang digunakan sebagai wadah untuk menyembunyikan pesan adalah *carrier file*. *Carrier file* dapat berupa teks, seperti txt, doc, html; audio dalam format wav, mp3; gambar, seperti bmp, jpeg, gif, png; dan video, baik dalam format mpeg, avi, dan mp4.

Selain itu, ada beberapa istilah yang terkait dengan steganografi. Istilah pertama adalah *embedded message* yang merujuk pada pesan yang disembunyikan. Istilah kedua adalah *cover-object* yang mengacu pada objek yang digunakan untuk menyembunyikan pesan yang tersembunyi. Istilah ketiga adalah *stego-object* atau *stegotext* yang merujuk pada objek yang telah mengandung pesan tersembunyi. Istilah terakhir adalah *stego-key* yang merupakan kunci yang digunakan untuk menyisipkan pesan dan mengeluarkan pesan dari stegotext.

Diagram alur proses steganografi ditunjukkan dalam Gambar 1 untuk memberikan gambaran visual tentang proses tersebut.

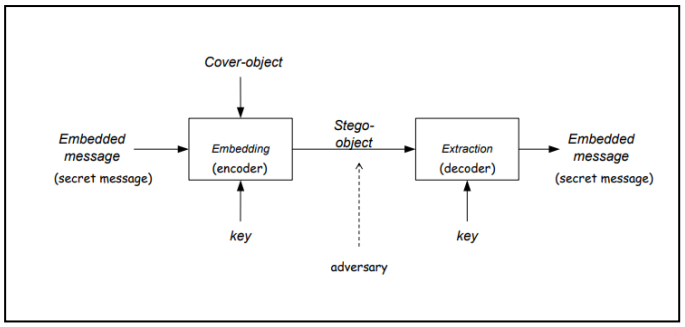

Fig 1. Diagram alur steganografi (Sumber: Slide Kuliah II4031 Kriptografi dan Koding: Tanda-tangan digital oleh Rinaldi Munir)

Ada beberapa faktor penting yang harus dipertimbangkan dalam menghasilkan steganografi yang efektif, yaitu kriteria *imperceptible, fidelity, recovery, dan capacity*. Pertama, kriteria *imperceptible* menunjukkan bahwa pesan rahasia yang disembunyikan tidak boleh terlihat secara visual atau terdengar secara audial oleh orang yang melihat atau mendengarkan media tersebut. Kriteria kedua adalah *fidelity* yang berarti kualitas objek penyamar tidak mengalami perubahan yang signifikan akibat penyisipan pesan rahasia. Kriteria ketiga adalah *recovery* yang menekankan bahwa pesan yang disembunyikan harus dapat diekstraksi kembali dengan benar tanpa kehilangan informasi penting. Terakhir, kriteria *capacity* yang menunjukkan bahwa kapasitas penyembunyian pesan harus sebesar mungkin, memungkinkan penyisipan pesan dengan ukuran yang lebih besar.

Kehadiran steganografi tentu terkait erat dengan kemunculan kriptografi. Steganografi tidak bertujuan untuk menggantikan kriptografi, tetapi keduanya ada untuk saling melengkapi. Kombinasi kriptografi dan steganografi dapat digunakan bersama-sama untuk meningkatkan keamanan pesan rahasia. Pendekatan ini melibatkan langkah-langkah berikut: pertama, pesan rahasia dienkripsi menggunakan algoritma kriptografi, dan kemudian pesan yang terenkripsi tersebut disisipkan ke dalam media lain seperti gambar, video, atau audio. Dengan demikian, pesan yang dienkripsi menjadi tidak terlihat secara langsung oleh orang yang tidak berwenang, dan informasi sensitif tetap aman dan tersembunyi dalam media yang tampaknya biasa.

Berdasarkan ranah operasinya, metode steganografi dibagi menjadi dua kelompok yaitu *spatial (time) domain methods* dan *transform domain methods*. Metode pertama, yaitu *spatial (time) domain methods* berarti memodifikasi langsung nilai *byte* dari *cover-object* yaitu nilai *byte* yang merepresentasikan intensitas pixel atau amplitudo. Salah satu contoh metode dalam domain spasial adalah metode *least significant bit* (LSB). Metode ini memanfaatkan *bit* terkecil dari setiap *byte* dalam *cover-object* untuk menyembunyikan pesan rahasia.

Sementara itu, metode *transform domain methods* memodifikasi hasil transformasi sinyal dalam ranah *transform* atau hasil transformasi dari ranah spasial ke ranah lain. Salah satu contoh metode dalam domain transformasi adalah metode spread spectrum. Metode ini memanfaatkan teknik penyebaran spektrum untuk menyembunyikan pesan rahasia dengan memodifikasi koefisien transformasi dalam domain frekuensi.

Pada makalah ini, metode yang akan digunakan dalam merahasiakan informasi adalah *Least Significant Bit*. Metode ini mengambil keuntungan dari bit terkecil atau *least significant* bit dari nilai *pixel* dalam gambar untuk menyembunyikan pesan rahasia. *Least significant bit* memiliki nilai yang paling sedikit pengaruh terhadap representasi visual dari gambar, sehingga modifikasi pada bit ini diharapkan tidak terdeteksi secara visual.

Sebagai contoh, terdapat bit 11010001 dan kita akan mengubah nilai byte satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Jika bit sebelumnya kita misalkan menyatakan warna merah, maka 11010000 juga menyatakan warna merah yang memiliki perbedaan sangat-sangat sedikit dibandingkan dengan warna merah sebelumnya. Perubahan warna merah tersebut sangat sedikit sehingga tidak dapat dibedakan oleh mata manusia. Berikut adalah contoh lain pada sebuah gambar yang semua bit *least significant byte*-nya dibalikkan dari semula 0 menjadi 1 dan dari yang semula 1 menjadi 0.

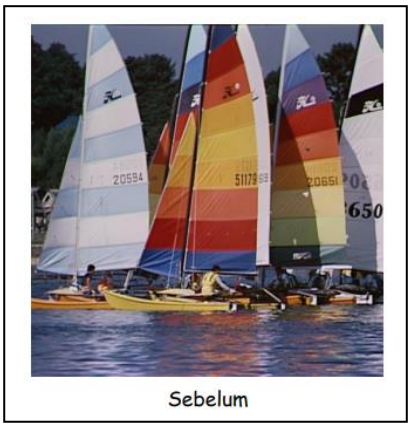

Fig 2. Contoh gambar sebelum dilakukan *encoding* (Sumber: Slide Kuliah II4031 Kriptografi dan Koding: Tanda-tangan digital oleh Rinaldi Munir)

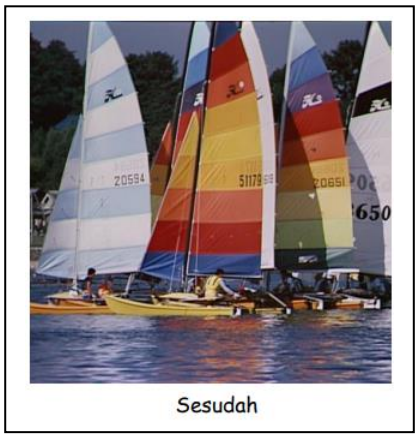

Fig 3. Contoh gambar sebelum dilakukan *encoding* (Sumber: Slide Kuliah II4031 Kriptografi dan Koding: Tanda-tangan digital oleh Rinaldi Munir)

Sama seperti dengan penjelasan mengenai contoh sebelumnya, perubahan *least significant bit* pada contoh gambar tidak akan memberikan perbedaan yang signifikan pada mata manusia. Hal tersebut dikarenakan penglihatan mata manusia yang terbatas dalam melihat warna sehingga siapapun yang melihatnya tidak akan merasakan perubahan yang ada.

Untuk dapat mengeluarkan kembali bit pesan yang tersembunyi dalam gambar, lakukan langkah-langkah berikut. Pertama, *byte-byte* dalam gambar dibaca, dan kemudian *least significant byte* dari setiap *byte* diambil. Selanjutnya, bit-bit yang diambil tersebut dirangkai kembali untuk memperoleh pesan rahasia yang semula tersembunyi. Dengan melakukan proses pembacaan dan pengambilan *least significant byte* ini, kita dapat mengembalikan pesan rahasia ke bentuk aslinya. Langkah ini penting dalam proses ekstraksi pesan yang disembunyikan dengan metode *least significant byte*

Terdapat beberapa varian metode *least significant byte*, yaitu *sequential*, acak, m-bit *least significant byte*, dan enkripsi. Dalam metode *sequential*, bit-bit pesan disisipkan ke dalam piksel gambar secara berurutan mengikuti urutan piksel yang ditemui. Untuk ekstraksi pesan, *pixel* akan dibaca secara berurutan, dimulai dari *pixel* pertama hingga *pixel* yang menyimpan bit pesan terakhir. Bit yang diambil dari *least significant byte* kemudian disusun kembali untuk mendapatkan kembali bit-bit pesan asli.

Metode *sequential* ini merupakan metode yang sederhana dan mudah diimplementasikan. Dalam proses penyisipan dan ekstraksi, urutan *pixel* dapat dilakukan dengan mudah. Namun, metode ini juga memiliki beberapa kelemahan, seperti rentan terhadap serangan statistik dan rentan terhadap penghapusan atau perubahan jika terjadi kompresi atau pemrosesan lain pada gambar.

Pada metode acak, bit-bit pesan tidak disimpan pada *pixel* secara berurutan, tetapi dipilih secara acak. Pengacakan ini dilakukan menggunakan pembangkit bilangan acak-semu atau *pseudo-random number generator*. Dalam proses ini, sebuah *seed* atau kunci digunakan sebagai input untuk pembangkit bilangan acak, yang juga berfungsi sebagai *stego-key* atau kunci steganografi.

Untuk mengekstraksi pesan dari *stego-image* (gambar tersembunyi), posisi *pixel* yang menyimpan bit pesan dapat diketahui berdasarkan bilangan acak yang dihasilkan oleh *pseudo-random number generator*. Jika kunci yang digunakan saat ekstraksi sama dengan kunci yang digunakan saat penyisipan, maka bilangan acak yang dihasilkan juga akan sama. Dengan demikian, bit pesan yang acak di dalam gambar dapat tersusun kembali.

Pada metode m-bit *least significant byte*, digunakan lebih dari 1 bit *leasat significant byte* pada setiap *byte* untuk meningkatkan ukuran pesan yang disembunyikan. Metode ini memiliki kekurangan jika kita menggunakan banyak bit *least significant byte*, yaitu kualitas *stego-image* yang semakin menurun. Namun, di sisi lain, semakin besar pula ukuran pesan yang dapat disembunyikan.

Metode terakhir adalah enkripsi. Metode ini diawali dengan mengenkripsi pesan terlebih dahulu sebelum menyembunyikannya di dalam gambar. Enkripsi yang dilakukan dapat sesederhana melakukan XOR pada bit pesan dan bit kunci. Hal tersebut diperbolehkan karena jumlah bit kunci sama dengan jumlah bit pesan. Bit kunci akan dibangkitkan secara acak dan kunci untuk pembangkitan bit kunci akan menjadi *stego-key*. Jika teknik acak dipakai dalam memilih *pixel*, maka akan ada dua *stego-key*. Salah satunya digunakan untuk pembangkitan bit kunci dan salah satunya digunakan untuk pembangkitan posisi *pixel* yang dipilih untuk menyembunyikan pesan.

Metode *least significant byte* memiliki keunggulan dalam kemudahan implementasi dan kecepatan proses yang relatif tinggi. Namun, metode ini juga memiliki beberapa kelemahan, seperti rentan terhadap serangan statistik dan penghapusan atau perubahan yang dapat terjadi pada bit *least significant byte* saat gambar mengalami kompresi atau pemrosesan lainnya.

III. Rancangan Solusi dan Implementasi

Berikut adalah rancangan solusi untuk permasalahan yang telah dibahas pada bagian Pendahuluan.

A. Deskripsi Umum Solusi

Pada masalah ini, akan disisipkan pesan rahasia pada sebuah gambar menggunakan steganografi. Penyisipan pesan rahasia sendiri akan dibuat dengan mengimplementasikan algortima steganografi metode *least significant byte*. Algoritma steganografi dipilih untuk diimplementasikan karena dibutuhkan penyisipan pesan rahasia yang kasat mata. Pengimplementasian tersebut bukanlah untuk menguji kemanan steganografi, melainkan menguji aplikasi Pinterest yang menjadi tempat pengunggahan dan pengunduhan gambar masyarakat pada umumnya.

B. Implementasi

Dalam implementasinya, terdapat 4 fungsi utama yang digunakan, yaitu fungsi *encode*, fungsi *decode*, fungsi *main* atau fungsi utama, dan fungsi modifikasi *pixel*.

1) Fungsi Encode

Proses *encode* akan menggunakan program yang menyisipkan pesan ke dalam gambar. Pengguna akan diminta untuk memasukkan nama gambar beserta *extension*-nya dan data yang akan disisipkan. Selain itu, fungsi ini akan membuka *file*, membuat salinan *file*, dan menyisipkan *pixel* yang sudah dimodifikasi. Data yang digunakan berasal dari variabel “data” dan menggunakan fungsi “modify_pixels” yang akan dijelaskan selanjutnya. Proses penyisipan dilakukan dengan memodifikasi nilai *pixel* agar merepresentasikan bit-bit data yang akan disisipkan. Gambar baru yang telah disisipkan data disimpan dalam variabel yang bernama “new_image”. Berikut adalah program fungsi “encode’.

```
def encode():
    image_name = input("Masukkan
nama file gambar (sertakan
extension): ")
    image =
Image.open(image_name, 'r')

    data = input("Masukkan data
yang akan di-encoded: ")
    if len(data) == 0:
        raise ValueError('Data
tidak boleh kosong')

    new_image = image.copy()

    #encoding image
    width = new_image.size[0]
    (x, y) = (0, 0)

    for pixel in
modify_pixels(new_image.getdata()
, data):
        new_image.putpixel((x,
y), pixel)
        if x == width - 1:
            x = 0
            y += 1
        else:
            x += 1

    new_image_name =
input("Masukkan nama file gambar
yang baru (sertakan extension):
")

new_image.save(new_image_name,
str(new_image_name.split(".")[1].
upper()))
```

Fig 4. Blok Kode Fungsi encode (sumber: dokumentasi pribadi)

2) Fungsi Decode

Proses *decode* akan mengekstraksi pesan dari gambar yang telah disisipkan pesan rahasia. Pengguna diminta untuk memasukkan nama gambar berserta *extention*-nya yang berisi pesan tersembunyi. Fungsi ini membuka gambar, membaca piksel-pikselnya, dan mengambil bit terakhir dari setiap *pixel* untuk membentuk data biner. Data biner tersebut kemudian dikonversi menjadi karakter sesuai dengan kode ASCII, dan pesan tersembunyi diekstraksi. Pada akhirnya, program akan mengembalikan pesan yang tersembunyi pada gambar. Berikut adalah program fungsi "decode".

```
def decode():
    image_name = input("Masukkan
nama file gambar (sertakan
extension): ")
    image =
Image.open(image_name, 'r')

    data = ''
    image_data =
iter(image.getdata())

    while True:
        pixels = [value for value
in image_data.__next__()[:3] +

image_data.__next__()[:3] +

image_data.__next__()[:3]]

        binary_string = ''

        for pixel_value in
pixels[:8]:
            if pixel_value % 2 ==
0:
                binary_string +=
'0'
            else:
                binary_string +=
'1'

        data +=
chr(int(binary_string, 2))
        if pixels[-1] % 2 != 0:
            return data
```

Fig 5. Blok Kode Fungsi decode (sumber: dokumentasi pribadi)

3) Fungsi Main

Proses pengeksekusian program akan diatur pada fungsi ini. Saat menjalankan program, program akan meminta

masukan dari pengguna untuk memilih ingin melakukan penyisipan pesan atau mengekstraksi pesan. Fungsi yang akan jalan selanjutnya akan dipanggil sesuai dengan pilihan dari pengguna. Berikut adalah kode program yang digunakan pada fungsi “main”.

```
def main():
    input = int(input("--
STEGANOGRAFI--\n"
                      "1.
Encode\n2. Decode\n"))

    if input == 1:
        encode()
    elif input == 2:
        print("Pesan rahasia: " +
decode())
    else:
        raise Exception("Mohon
masukkan input yang benar")
```

Fig 6. Blok Kode Fungsi main (sumber: dokumentasi pribadi)

4) Fungsi Modifikasi Pixel

Selain ketiga fungsi utama, terdapat fungsi “modify pixels”. Fungsi “modify pixels” berfungsi untuk mengonversi data yang akan disisipkan menjadi representasi biner 8-bit menggunakan nilai ASCII dari setiap karakter. Data yang dihasilkan berupa *list* dari kode biner. Selain itu, fungsi ini berguna untuk memodifikasi *pixel* gambar berdasarkan data biner 8-bit yang diberikan. Setiap *pixel* diambil sebanyak 3 nilai (RGB), dan nilai piksel dimodifikasi agar menjadi ganjil (*odd*) untuk bit 1 dan genap (*even*) untuk bit 0. Modifikasi dilakukan dengan memanipulasi nilai *pixel* sesuai dengan bit data yang akan disisipkan. Fungsi ini mengembalikan nilai piksel yang telah dimodifikasi. Berikut merupakan kode program yang digunakan pada fungsi “modify_pixels”.

```
def modify_pixels(pixels, data):
    new_data = []
    for char in data:

new_data.append(format(ord(char),
'08b'))

    data_length = len(new_data)
    pixel_data = iter(pixels)

    for i in range(data_length):
        pixel = [value for value
in pixel_data.__next__()[:3] +
pixel_data.__next__()[:3] +
pixel_data.__next__()[:3]]
```

```
for j in range(8):
            if new_data[i][j] ==
'0' and pixel[j] % 2 != 0:
                pixel[j] -= 1
            elif new_data[i][j]
== '1' and pixel[j] % 2 == 0:
                if pixel[j] != 0:
                    pixel[j] -= 1
                else:
                    pixel[j] += 1

        if i == data_length - 1:
            if pixel[-1] % 2 ==
0:
                if pixel[-1] !=
0:
                    pixel[-1] -=
1
                else:
                    pixel[-1] +=
1
        else:
            if pixel[-1] % 2 !=
0:
                pixel[-1] -= 1

        pixel = tuple(pixel)
        yield pixel[0:3]
        yield pixel[3:6]
        yield pixel[6:9]
```

Fig 7. Blok Kode Fungsi modify_pixels (sumber: dokumentasi pribadi)

IV. Pengujian dan Pembahasan

A. *Pengujian*

Berikut adalah kasus pengujian yang akan dilakukan.

TABLE I. Kasus Pengujian

No	Kasus	Ekspektasi *Output*
1	Pengunggahan gambar yang sudah di-*encoding*	Pengunggahan berhasil
2	*Decoding* gambar hasil unduhan dari Pinterest yang memiliki pesan rahasia	Mengeluarkan pesan rahasia

Pengujian akan langsung dilakukan pada aplikasi Pinterest dengan menggunakan program steganografi yang sudah dibuat sebelumnya. Berikut adalah penjelasan untuk setiap kasus pengujian.

*1) Kasus Pengunggahan Gambar yang Sudah Di-*encoding

Pada kasus ini, akan disisipkan pesan rahasia pada sebuah gambar. Berikut adalah gambar sebelum dan sesudah disisipkan pesan.

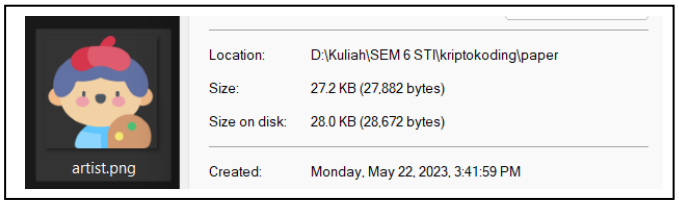

Fig 8. Gambar sebelum disisipkan pesan (sumber: dokumentasi pribadi)

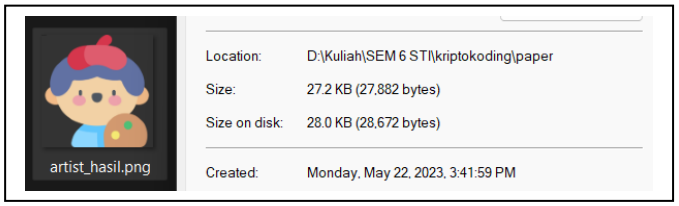

Fig 9. Gambar seusdah disisipkan pesan (sumber: dokumentasi pribadi)

Untuk pesan yang akan disisipkan, kita akan menggunakan kalimat "kena deh" dan menamakan gambar yang telah di-*encode* dengan nama "artist_hasil.png" sebagai contoh. Dapat dilihat pada gambar bahwa pesan telah tersisipkan pada gambar saat kita kembali melakukan *encoding*.

```
D:\Kuliah\SEM 6 STI\kriptokoding\paper>python -u "d:\Kuliah\
oding\paper\steganografi_gfg.py"
:: Welcome to Steganography ::
1. Encode
2. Decode
2
Enter the image name (with extension): artist_hasil.png
Decoded Word: kena deh
```

Fig 10. Hasil program saat melakukan *decode* pada *file* "artist_hasil.png" (sumber: dokumentasi pribadi)

Berikut dilampirkan bukti bahwa gambar yang sudah di-*encode* berhasil diunggah ke aplikasi Pinterest.

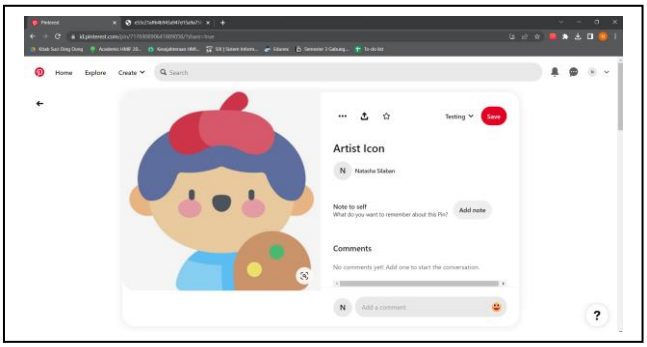

Fig 11. Pengunggahan *file* "artist_hasil.png" ke Pinterest (sumber: dokumentasi pribadi)

Bukti gambar menunjukkan bahwa gambar yang sudah disisipkan pesan berhasil diunggah ke Pinterest.

2) Kasus Decoding Gambar Hasil Unduhan Dari Pinterest yang Memiliki Pesan Rahasia

Pada kasus ini, akan dilakukan pengunduhan gambar yang telah disisipkan pesan rahasia melalui aplikasi Pinterest. Gambar yang telah diunduh akan dinamakan dengan "artisti_pinterest.png" kemudian akan digunakan sebagai *input* unttuk dilakukan *decoding*. Berikut merupakan gambar yang diunduh secara langsung dari Pinterest.

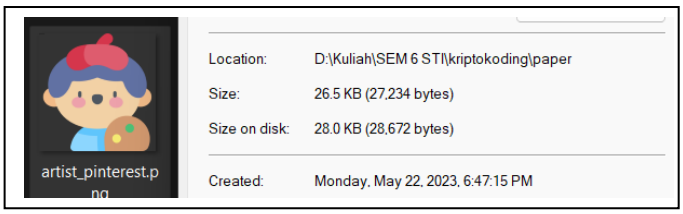

Fig 12. Gambar yang diunduh dari aplikasi Pinterest (sumber: dokumentasi pribadi)

Jika dilakukan *decoding* pada gambar tersebut, kita akan mendapatkan hasil pesan tersembunyi seperti berikut.

```
D:\Kuliah\SEM 6 STI\kriptokoding\paper>python -u "d:\Kuliah\S
oding\paper\steganografi_gfg.py"
:: Welcome to Steganography ::
1. Encode
2. Decode
2
Enter the image name (with extension): artist_pinterest.png
Decoded Word: kena deh
```

Fig 13. Hasil program saat melakukan *decode* pada file "artist_pinterest.png" (sumber: dokumentasi pribadi)

Bukti gambar menunjukkan bahwa pengunduhan gambar yang sudah disisipkan pesan dari Pinterest masih dapat menghasilkan pesan semula saat di-*decode*.

Berikut adalah *link pin* dari gambar yang telah memiliki pesan rahasia pada aplikasi Pinterest.

https://id.pinterest.com/pin/717690890641889058/

B. Pembahasan

Berikut adalah pembahasan dari setiap pengujian yang telah dilakukan.

*1) Kasus Pengunggahan Gambar yang Sudah Di-*encoding

Kasus pengujian ini melibatkan *input* gambar yang valid. Berdasarkan hasil pengujian yang telah dilakukan di atas, bukti gambar yang dilampirkan pada dokumen sesuai dengan ekspetasi. Artinya, pengujian untuk kasus ini berhasil.

2) Kasus Decoding Gambar Hasil Unduhan Dari Pinterest yang Memiliki Pesan Rahasia

Kasus pengujian ini melibatkan *input* gambar yang valid. Berdasarkan hasil pengujian yang telah dilakukan di atas, terbukti bahwa gambar yang sudah diunggah ke Pinterest masih bisa memiliki pesan rahasia tersisipkan di dalamnya saat diunduh oleh pengguna. Artinya, kasus pengujian untuk kasus ini berhasil dan sesuai dengan ekspertasi.

V. KESIMPULAN DAN SARAN

Pada makalah ini, dilakukan analisis berupa decoding pesan rahasia terhadap foto yang diunduh dari aplikasi Pinterest. Pada makalah ini dibuktikan bahwa pesan rahasia yang disisipkan ke gambar menggunakan steganografi metode *least significant byte* masih dapat dibaca oleh siapapun yang mengunduh gambar tersebut walaupun mereka menggunakan aplikasi Pinterest. Kedepannya diharapkan bahwa terdapat solusi untuk mencegah disisipkannya pesan rahasia yang berbahaya pada foto yang diunggah pada aplikasi Pinterest agar tidak merugikan siapapun yang mengunduh gambar tersebut.

UCAPAN TERIMA KASIH

Saya mengucapkan rasa syukur dan terima kasih kepada Tuhan Yang Maha Esa yang telah memberikan bimbingan dan pertolongan dalam penyusunan makalah ini. Tidak lupa, saya juga ingin mengucapkan terima kasih kepada kedua orang tua saya yang selalu memberikan dukungan dan dorongan kepada saya dalam menyelesaikan tugas ini. Saya juga ingin mengungkapkan rasa terima kasih kepada Pak Rinaldi Munir yang telah memberikan materi mengenai Kriptografi dan Koding dengan cara yang mudah dipahami.

Selain itu, ada banyak pihak lain yang turut membantu saya dalam proses penyusunan makalah ini. Saya berterima kasih kepada rekan-rekan saya yang juga mengambil mata kuliah Kriptografi dan Koding, kakak tingkat yang memberikan referensi yang bermanfaat dalam pembuatan makalah ini, serta para penulis yang mempublikasikan hasil penelitian mereka di internet. Semua kontribusi dan dukungan dari mereka sangat berarti bagi saya dalam menyelesaikan makalah ini.

REFERENSI

[1] Munir, Rinaldi. 2022. Slide Kuliah II4031 Kriptografi dan Koding: Steganografi

[2] Fridrich, J. (2009). Steganography in Digital Media: Principles, Algorithms, and Applications (2nd Edition). Cambridge University Press.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 22 Mei 2023

Natasha Tiovanny Silaban 18220101