Implementasi dan Analisis Algoritma RSA dan Steganografi untuk Menjaga Rahasia Foto KTP

Fernaldi Fauzie - 18219099
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: 18219099@std.stei.itb.ac.id

***Abstract*—Foto Kartu Tanda Penduduk (KTP) adalah gambar yang sangat penting dan dapat disalahgunakan bila dimiliki oleh pihak yang tidak bertanggung jawab. Sehingga, diperlukan upaya lebih untuk menjaga keamanan foto KTP tersebut. Penerapan algoritma RSA sebagai kriptografi yang digabung dengan steganografi merupakan salah satu solusi untuk menjaga rahasia foto KTP. Dengan menggabungkan algoritma RSA dengan steganografi, kita dapat meningkatkan keamanan foto KTP supaya foto KTP dapat dikirim ke pihak yang diinginkan tanpa didapatkan oleh pihak ketiga.**

Keywords— foto KTP, RSA, steganografi, keamanan

I. PENDAHULUAN

Kartu Tanda Penduduk (KTP) adalah kartu yang menunjukkan identitas resmi seorang penduduk sebagai bukti diri yang diterbitkan oleh instansi pelaksana yang berlaku di negara Indonesia. KTP wajib dimiliki oleh Warga Negara Indonesia (WNI) dan Warga Negara Asing (WNA) yang menetap di Indonesia, yang sudah berumur 17 tahun atau yang sudah pernah memiliki pasangan suami/istri. KTP mengandung berbagai informasi penting mengenai sang pemilik kartu, seperti Nomor Induk Kependudukan (NIK), nama lengkap, tempat & tanggal lahir, jenis kelamin, agama, status, golongan darah, alamat lengkap pemegang KTP (RT, RW, Kelurahan, dan Kecamatan), pekerjaan, pas foto, tempat dan tanggal dikeluarkannya KTP, tanda tangan pemilik KTP, dan nama dan nomor induk pegawai pejabat yang menandatanganinya.

KTP sebagai identitas jati diri, sering digunakan dalam kehidupan sehari-hari. Dalam kondisi tertentu, kita diminta untuk mengirim KTP dalam bentuk foto kepada pihak lain, seperti perusahaan. Foto KTP tersebut dapat dimanfaatkan untuk mendaftarkan diri ke suatu perusahaan sebagai pekerja tetap atau magang, mendaftar sebagai seorang pengguna sebuah aplikasi (yang memerlukan verifikasi identitas), mengganti nama penumpang pesawat yang *typo* di tiket pesawat, dll. Sehingga, foto KTP memiliki keamanan yang cukup rentan untuk dikirim dari suatu pihak ke pihak lain. Foto KTP yang dikirim tersebut dapat diserang oleh pihak ketiga seperti *hacker* yang tidak bertanggung jawab dan memiliki maksud yang tidak baik untuk menyalahgunakan foto KTP tersebut.

Salah satu cara untuk mengatasi permasalahan keamanan foto KTP adalah dengan melakukan enkripsi terhadap foto KTP tersebut untuk menjaga keamanan foto KTP tersebut, kemudian melakukan steganografi digital untuk menyembunyikan hasil enkripsi foto KTP tersebut sehingga mengurangi kemungkinan ditemukannya hasil enkripsi foto KTP oleh pihak ketiga yang tidak diharapkan.

Pada makalah ini, akan diajukan sistem sederhana yang dapat melakukan pembuatan kunci publik dan privat untuk enkripsi, melakukan enkripsi dan dekripsi file seperti foto KTP, menyembunyikan pesan rahasia ke file teks dan gambar, dan mengambil pesan rahasia dari suatu pesan yang telah disisipkan pesan rahasia.

II. DASAR TEORI

A. RSA

RSA adalah algoritma kunci-publik yang populer dan telah digunakan di mana-mana. Kunci publik digunakan untuk mengenkripsi pesan, sedangkan kunci privat digunakan untuk mendekripsi pesan yang telah dienkripsi. RSA ditemukan oleh Ronald Rivest, Adi Shamir, dan Len Adleman. Ketiga orang tersebut merupakan peneliti dari universitas *Massachussets Institute of Technology* (MIT). RSA sendiri memiliki kepanjangan yaitu Rivest-Shamir-Adleman. Algoritma RSA memiliki keamanan yang baik karena sulit untuk melakukan pemfaktoran bilangan bulat yang besar menjadi faktor-faktor prima.

Secara umum, proses RSA terbagi menjadi 3 bagian, yaitu pembangkitan kunci, enkripsi pesan, dan dekripsi pesan yang telah dienkripsi. Berikut merupakan langkah-langkah dalam melakukan pembangkitan kunci.

1. Pilih dua bilangan prima, yaitu p dan q
2. Hitung nilai n dengan melakukan perkalian p dan q
3. Hitung nilai fungsi *totient* $\phi(n)$ dengan melakukan perkalian antara (p-1) dan (q-1)
4. Pilih sebuah bilangan bulat e sebagai kunci publik, dengan syarat e harus relatif prima terhadap $\phi(n)$
5. Hitung d sebagai kunci dekripsi, yang didapatkan dari persamaan $ed \equiv 1 \pmod{\phi(n)}$ atau $d \equiv e - 1 \bmod (\phi(n))$

6. Sehingga, didapatkan kunci publik yaitu pasangan (e, n) dan kunci privat yaitu pasangan (d, n)

Kemudian, langkah-langkah melakukan enkripsi adalah sebagai berikut.

1. Ubah pesan yang ingin dienkripsi menjadi blok-blok plainteks: m_1, m_2, m_3, …. (dengan syarat: $0 <= m_i < n - 1$)
2. Hitung blok cipherteks c_i untuk blok plainteks m_i dengan menggunakan kunci publik e dengan persamaan $c_i = m_i^e \bmod n$

Berikutnya, langkah-langkah melakukan dekripsi adalah sebagai berikut.

1. Sebelumnya, telah didapatkan blok-blok ciperteks yaitu c_1, c_2, c_3, ….
2. Hitung blok plainteks m_i dari blok cipherteks c_i menggunakan kunci privat d dengan persamaan $m_i = c_i^d \bmod n$

B. Steganografi

Steganografi adalah ilmu dan seni menyembunyikan pesan rahasia ke dalam pesan lain atau objek fisik, sehingga selain pihak pengirim dan penerima, tidak ada pihak yang mencurigai keberadaan pesan yang disembunyikan tersebut.

Di dalam steganografi, terdapat terminologi sebagai berikut:

1. *Embedded message*: Pesan yang disembunyikan. Pesan dapat berbentuk teks, gambar, audio, video, dll.
2. *Cover-object*: Pesan yang digunakan untuk menyembunyikan *embedded message*. *Cover-object* dapat berbentuk teks, gambar, audio, video, dll.
3. *Stego-object*: *Cover-object* yang telah dimasukkan pesan *embedded message*
4. *Stego-key*: Kunci yang digunakan untuk menyisipkan pesna dan mengekstrasi pesan dari *stego-key*.

III. Implementasi dan Analisis

Pada makalah ini, diajukan penggabungan kriptografi dengan menggunakan RSA dan steganografi. Secara singkat, langkah-langkah yang perlu dilakukan adalah sebagai berikut:

1. Menghitung kunci publik dan privat, kemudian menyimpan kunci publik di dalam file dengan ekstensi file .pub dan menyimpan kunci privat di dalam file dengan ekstensi file .pri
2. Melakukan enkripsi terhadap gambar KTP. Hasil enkripsi akan berbentuk *string hex*.
3. Menyembunyikan hasil enkripsi (*string hex*) ke dalam file teks atau file gambar, sesuai dengan pilihan pengguna
4. Mengambil *string hex* yang disembunyikan di file teks atau file gambar
5. Melakukan dekripsi terhadap *string hex* tersebut untuk mendapatkan kembali gambar KTP

Dalam mengimplementasikan program dengan menggunakan bahasa Python, berikut merupakan *import* yang digunakan.

```
import random
from tkinter import *
from tkinter import Tk
from tkinter.filedialog import askopenfilename, asksaveasfile,
asksaveasfilename
from tkinter import messagebox
from stegano import lsb
```

A. Pembangkitan Kunci Publik dan Privat

Untuk menghitung kunci publik dan privat, dapat menggunakan fungsi sebagai berikut:

```
def generatePairNumbers():
    print("Generate p and q")
    p = 0
    q = 0
    while(not(isPrime(p))):
        p = int([random.randint(2**4, 2**12) for i in
range(1)][0])
    while(not(isPrime(q))):
        q = int([random.randint(2**4, 2**12) for i in
range(1)][0])
    return p, q

def isPrime(num):
    if num == 2:
        return True
    elif num < 2:
        return False
    else:
        for i in range(2, num, 1):
            if num % i == 0:
                return False
    return True

def gcd(x, y):
    while (y != 0):
        x, y = y, x % y
    return x

def generatePublicKey(phi):
    publicKeyCandidate = []
    for e in range(2, phi):
        if (gcd(e, phi) == 1):
            publicKeyCandidate.append(e)
    e = random.choice(publicKeyCandidate)
    return e

def generatePrivateKey(phi, e):
    for k in range(phi):
```

```
        d = (phi*k + 1)/e
        if (d % 1 == 0):
            return d

def generatePairKey():
    d = None
    while (d == None):
        p, q = generatePairNumbers()
        n = p*q
        phi = (p-1)*(q-1)
        e = generatePublicKey(phi)
        d = int(generatePrivateKey(phi, e))
        print("Checking if d is not None...")
    print("Pair key generated")
    return (e, n), (d, n)

def getPublicKey(pairKey):
    return pairKey[0]

def getPrivateKey(pairKey):
    return pairKey[1]

def savePublicKey(publicKey):
    files = [("Public Key", "*.pub")]
    file = asksaveasfile(
        filetypes=files, defaultextension=files, initialfile="public-
key")
    if file is None:
        return
    file.write(str(publicKey))
    file.close()

def savePrivateKey(privateKey):
    files = [("Private Key", "*.pri")]
    file = asksaveasfile(
        filetypes=files, defaultextension=files,
initialfile="private-key")
    if file is None:
        return
    file.write(str(privateKey))
    file.close()
```

Cara menghitung kunci publik dan kunci privat, kemudian menyimpannya di file dengan *extension* yang sesuai adalah sebagai berikut.

```
pairKey = generatePairKey()
publicKey = getPublicKey(pairKey)
privateKey = getPrivateKey(pairKey)

savePublicKey(publicKey)
savePrivateKey(privateKey)
```

B. Enkripsi Menjadi *String Hex*

Untuk mengenkripsi gambar KTP menjadi *string hex*, dapat digunakan fungsi sebagai berikut:

```
def readKey():
    keyFileName = askopenfilename()
    key = (open(keyFileName)).read()

    for char in ("() "):
        key = key.replace(char, "")
    key = key.split(",")

    key[0] = int(key[0])
    key[1] = int(key[1])

    return key

def openFileAsByteArray(Path):
    file = open(Path, "rb")
    data = file.read()
    file.close()

    byteArray = bytearray(data)
    return byteArray

def convertIntegerToHex(integerArray):
    hexArray = [0 for i in range(len(integerArray))]
    for i in range(len(integerArray)):
        hexArray[i] = hex(integerArray[i])[2:(len(integerArray))]
    return hexArray

def encryptPicture(e, n, Path):
    byteArray = openFileAsByteArray(Path)
    encryptionArray = [0 for i in range(len(byteArray))]
    for i, value in enumerate(byteArray):
        encryptionArray[i] = value**e % n

    hexArray = convertIntegerToHex(encryptionArray)
    hexString = " "
    hexString = hexString.join(hexArray)
    return hexString
```

Berikut merupakan cara menggunakan fungsi di atas.

```
publicKey = readKey() # Pilih public key yang telah di-save
fileToHide = askopenfilename() # Pilih foto KTP
hexString = encryptPicture(publicKey[0], publicKey[1],
fileToHide)
```

C. Menyembunyikan *Hex String* di File Teks

Berikut merupakan fungsi yang digunakan.

```
def hideHexInText(embeddedMessage, pathCoverText):
    embeddedMessageExist = False
    with open(pathCoverText) as coverText:
        lines = coverText.readlines()
        for line in lines:
```

```
        if line.rstrip('\n').startswith('<em>') and
line.rstrip('\n').endswith('</em>'):
            embeddedMessageExist = True


   if(embeddedMessageExist):
      numberLines = sum(1 for line in open(pathCoverText))
      with open(pathCoverText, "r") as coverText:
         data = coverText.readlines()
      data[numberLines-1] = "<em>" + embeddedMessage +
"</em>"


      with open(pathCoverText, "w") as coverText:
         coverText.writelines(data)
   else:
      with open(pathCoverText, "a") as coverText:
         coverText.write("\n")
         coverText.write("<em>" + embeddedMessage +
"</em>")
```

Berikut merupakan cara menggunakan fungsi di atas.

```
coverText = askopenfilename()
hideHexInText(hexString, coverText)
```

D. Menyembunyikan *Hex String* di File Gambar PNG

Berikut merupakan cara menyembunyikan *hex string* di file gambar PNG.

```
coverObject = askopenfilename()
stegoObject = lsb.hide(coverObject, hexString)

stegoObjectFileName = asksaveasfilename()
stegoObject.save(stegoObjectFileName)
```

Dalam menyembunyikan *hex string* ke dalam file gambar PNG, digunakan fungsi **lsb.hide** yang berasal dari modul stegano**.** Fungsi lsb.hide ini hanya efektif pada *cover object* dengan ekstensi PNG. Tidak dapat digunakan *cover* object dengan ekstensi file seperti JPEG karena dalam kasus JPEG, terdapat kompresi JPEG (*lossy compression*). stegoObject.save() digunakan untuk menghasilkan *stego-object* dalam bentuk gambar.

E. Mengambil *Hex String* dari File Teks

Berikut merupakan fungsi yang digunakan.

```
def readHiddenHexInText(pathStegoText):
   with open(pathStegoText) as stegoText:
      lines = stegoText.readlines()
      hiddenHexExist = False
      for line in lines:
         if line.rstrip('\n').startswith('<em>') and
line.rstrip('\n').endswith('</em>'):
            hiddenHexExist = True


      numberLines = len(lines)
      if (hiddenHexExist):
         with open(pathStegoText, "r") as stegoText:
            data = stegoText.readlines()
            data = data[numberLines-1]
         hiddenHexString = data[4:len(data) - 5:]
         return hiddenHexString
      else:
         messagebox.showinfo("Warning", "Tidak ada pesan
rahasia")
```

Berikut merupakan cara menggunakan fungsi di atas.

```
stegoText = askopenfilename()
hiddenHexString = readHiddenHexInText(stegoText)
```

F. Mengambil *Hex String* dari File Gambar PNG

Berikut merupakan cara mengambil *hex string* dari file gambar PNG.

```
stegoObjectFileName = askopenfilename()
hiddenHexString = lsb.reveal(stegoObjectFileName)
```

Dalam mengambil *hex string* yang telah disembunyikan di dalam file gambar PNG, digunakan fungsi **lsb.reveal** yang berasal dari modul stegano.

G. Dekripsi *Hex String* menjadi gambar KTP kembali

Berikut merupakan fungsi yang digunakan.

```
def decryptCipherText(d, n, hexString):
   hexArray = hexString.split()

   decryptionArray = [0 for i in range(len(hexArray))]
   for i in range(len(hexArray)):
      decryptionArray[i] = int(hexArray[i], 16)

   for i, value in enumerate(decryptionArray):
      decryptionArray[i] = value**d % n

   decryptionArray = bytearray(bytes(decryptionArray))
   return decryptionArray


def saveDecryptedFile(bytesArray):
   saveFileName = asksaveasfilename()
   with open(saveFileName, "wb") as binaryFile:
      binaryFile.write(bytesArray)
```

Berikut merupakan cara menggunakan fungsi di atas.

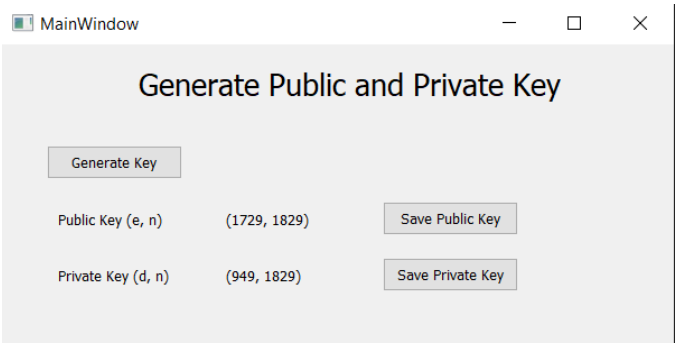
```
privateKey = readKey()  # Pilih private key yang telah di-save
decryptionArray = decryptCipherText(privateKey[0],
```

```
privateKey[1], hiddenHexString)
saveDecryptedFile(decryptionArray)
```

IV. TAMPILAN GUI

A. Tampilan GUI Pembangkitan Kunci Publik dan Privat

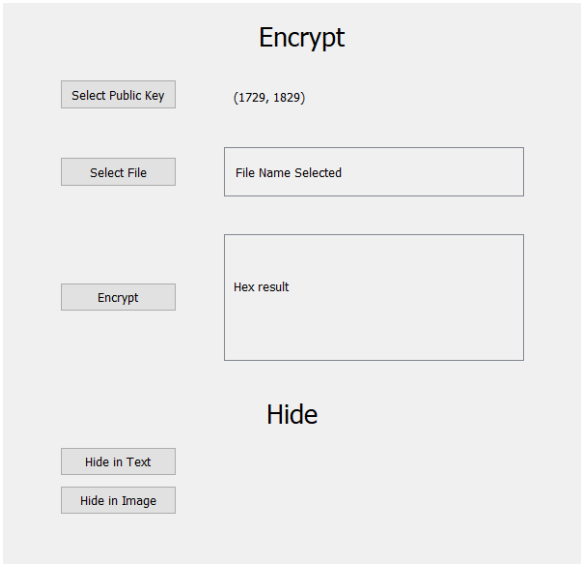
Berikut merupakan tampilan GUI sederhana setelah menekan tombol "*Generate Key*", sehingga ditampilkan kunci publik dan kunci privat yang telah dihitung oleh program.



Gambar 1 GUI Pembangkitan Kunci

B. Tampilan GUI Enkripsi dan Menyembunyikan File

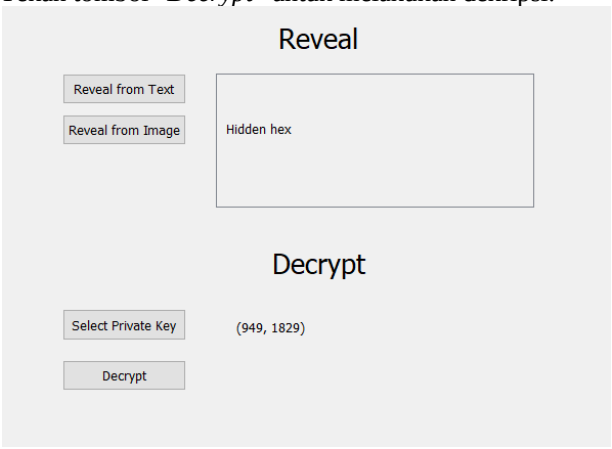
Berikut merupakan tampilan GUI enkripsi dan menyembunyikan file yang sederhana. Setelah memilih file kunci publik, akan ditampilkan nilai kunci tersebut dalam bentuk (e, n). Setelah memilih file untuk dienkripsi, akan ditampilkan filename yang dimiliki file tersebut. Contoh: "D:/ktp-1.jpg". Kemudian, setelah menekan *encrypt*, pada kotak "*Hex result*" akan ditampilkan hasil enkripsi berupa *hex string* yang dapat di-*scroll* secara vertikal. *Hide in Text* berguna untuk menyembunyikan *string hex* ke teks, sedangkan *Hide in image* berguna untuk menyembunyikan *string hex* ke gambar.



Gambar 2 GUI Enkripsi dan Penyembunyian

C. Tampilan GUI Mengambil File dan Dekripsi

Berikut merupakan tampilan GUI mengambil file dan dekripsi yang sederhana. Setelah memilih *reveal from text* atau *reveal from image*, di kotak "*Hidden hex*", akan ditampilkan pesan yang sebelumnya disembunyikan yang dapat di-*scroll* secara vertikal. Kemudian, setelah memilih file kunci privat, akan ditampilkan nilai kunci privat tersebut dalam bentuk (d, n). Tekan tombol "*Decrypt*" untuk melakukan dekripsi.



Gambar 3 GUI Pengambilan dan Dekripsi

V. PENGUJIAN

Berikut merupakan contoh gambar KTP yang akan disembunyikan.

Gambar 4 Contoh Foto KTP

Kunci yang digunakan adalah kunci publik (127, 589) dan kunci privat (523, 589). Hasil enkripsi foto KTP tersebut adalah sebagai berikut (tidak semua hasil enkripsi ditampilkan karena terlalu panjang).

```
171 d8 171 152 0 e1 37 176 a8 176 0 1 1 1 0 ac 0 ac 0 0 171 fe
0 23c 43 240 b7 8d 14b 88 240 4d 20 242 d6 241 ef 83 195 242
20 76 1f9 1aa 69 20 17e 3 16a 56 a7 242 20 a8 37 19d 20 37
218 b7 19d 20 76 153 45 204 b1 20 12 3 4e 1e8 56 191 2d 20
3d 20 38 69 1e5 171 13a 0 43 0 6 23e 5 6 5 23e 6 6 5 6 79 79 6
138 1e5 e1 1e5 1e5 15f 15f 1e5 202 222 18b 246 e1 134 202
1f3 1f3 134 202 10d 10d 1a 20b 25 1f 1a 2e 10 21d 10d 10d 20
b1 20 10 98 157 204 89 204 19 1f 241 69 241 17e 69 25 17e
204 17e 171 13a 0 43 1 79 79 79 1e5 138 1e5 26 1e5 1e5 26
17e 1a 10d 1a 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e
```

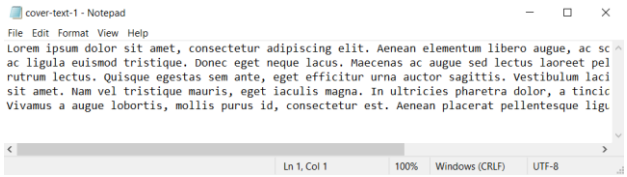
17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 171 1f2 0 4a 138 61 20 4f 19b 4f 1 6e 0 61 4a 1 4f 4a 1 171 52 0 2e 0 0 61 4f 1 1 1 0 0 0 0 0 0 0 0 0 0 0 61 1 4f 23e 5 6 79 171 52 0 1f3 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 61 4f 23e 171 da 0 246 4f 1 0 61 e1 4f e1 0 0 1 f9 17 20b 39 ca b3 ff 7f 41 a0 246 17 0 0 …..

….

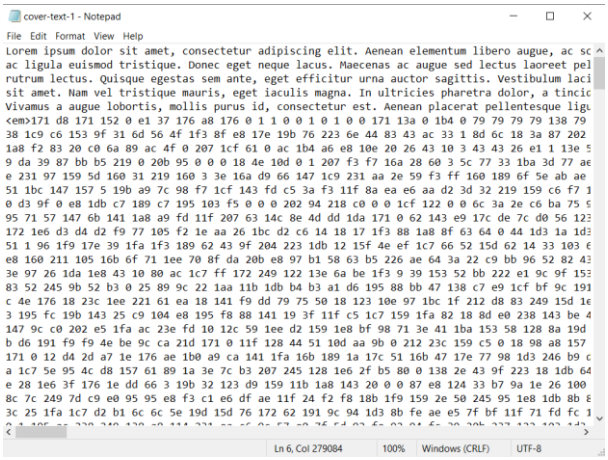
….

a 2e ab f3 25 8b ae 5 2e 58 1a8 141 1 ac 31 1da 1c9 1da 59 14c 1da 189 23c 43 1ee 221 a9 1c7 df e9 c7 dd 4c 95 ac f2 124 df 12 6d 7c 1ee e9 17c 221 147 e5 13e f7 6c fd 19 fd d4 3f 4d 14c 81 ae 242 f7 c5 147 c7 1f 19b 8c d9 43 f2 246 8b dd 14b cb 3e 8c ca 88 160 94 97 d3 147 70 fd 202 fe 226 cb 3 8c ca 242 97 c5 159 204 fd 8a cb 13e 1f e0 a0 4d f9 64 e9 66 3f ba 8c e8 62 bf 89 cb 3 1f9 147 70 fd b4 fe 159 cb 7d 39 5e 191 fe 9a 1f 1b0 88 123 41 147 15d 18b 143 b7 a9 f2 1f9 e9 41 3f 226 141 160 41 1b0 d2 37 df 3 3f 105 a2 143 d3 159 17e fe 172 3f e5 141 160 1a8 211 f9 218 e9 da cb 3 1f 47 94 a9 6c fd 14b 133 1e6 94 97 c5 147 c7 20b 7c 12d 6b e0 4f f9 6c fd d4 fe 9a cb 13e 3f 105 8c e8 88 1e6 221 147 e5 57 d2 242 a9 94 97 6c fd 8a cb 6c 8c e8 242 a9 58 fe 3 3f 226 8c e8 88 160 41 3f ba 96 e9 70 fd 10d 1aa 195 bb c0 b4 1aa f3 159 e5 fe f5 18b 19b 1f9 19d 241 de 202 7b d3 159 64 1c9 249 76 1 f3 176 50 1b 241 f3 9f a2 1b ef 104 47 18b 88 24 14c 160 43 ea f3 1f 104 13 d8 5e 1bc 3f 1b cb e cb a7 64 f3 43 143 37 d4 a9 7d 191 212 171 0 1a be 105 10 171 0 b3 50 fe 20b cb 124 be f3 159 100 b9 a9 1c7 df 1ba 171 d9

Berikut merupakan tampilan sebelum dan sesudah *string hex* disembunyikan di file teks.



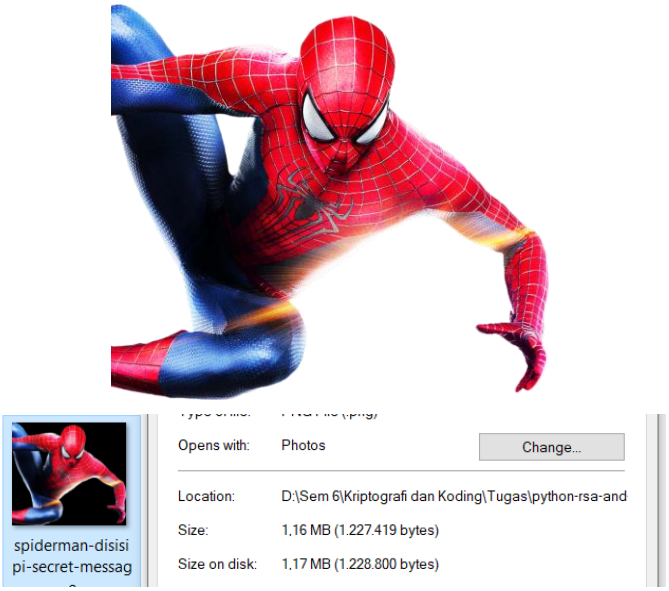
Gambar 5 *Cover-Text*



Gambar 6 *Stego-Object* (Teks)

Berikut merupakan tampilan sebelum dan sesudah *string hex* disembunyikan di file PNG.

Gambar 7 *Cover-Image*



Gambar 8 *Stego-Object* (Gambar)

Berikut merupakan hasil pengambilan *hex* baik dari file teks maupun file gambar yang telah disisipi.

171 d8 171 152 0 e1 37 176 a8 176 0 1 1 1 0 ac 0 ac 0 0 171 fe 0 23c 43 240 b7 8d 14b 88 240 4d 20 242 d6 241 ef 83 195 242 20 76 1f9 1aa 69 20 17e 3 16a 56 a7 242 20 a8 37 19d 20 37 218 b7 19d 20 76 153 45 204 b1 20 12 3 4e 1e8 56 191 2d 20 3d 20 38 69 1e5 171 13a 0 43 0 6 23e 5 6 5 23e 6 6 5 6 79 79 6 138 1e5 e1 1e5 1e5 15f 15f 1e5 202 222 18b 246 e1 134 202 1f3 1f3 134 202 10d 10d 1a 20b 25 1f 1a 2e 10 21d 10d 10d 20 b1 20 10 98 157 204 89 204 19 1f 241 69 241 17e 69 25 17e 204 17e 171 13a 0 43 1 79 79 79 1e5 138 1e5 26 1e5 1e5 26

17e 1a 10d 1a 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 17e 171 1f2 0 4a 138 61 20 4f 19b 4f 1 6e 0 61 4a 1 4f 4a 1 171 52 0 2e 0 0 61 4f 1 1 1 0 0 0 0 0 0 0 0 0 0 0 61 1 4f 23e 5 6 79 171 52 0 1f3 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 61 4f 23e 171 da 0 246 4f 1 0 61 e1 4f e1 0 0 1 f9 17 20b 39 ca b3 ff 7f 41 a0 246 17 0 0 …..

….

….

….

a 2e ab f3 25 8b ae 5 2e 58 1a8 141 1 ac 31 1da 1c9 1da 59 14c 1da 189 23c 43 1ee 221 a9 1c7 df e9 c7 dd 4c 95 ac f2 124 df 12 6d 7c 1ee e9 17c 221 147 e5 13e f7 6c fd 19 fd d4 3f 4d 14c 81 ae 242 f7 c5 147 c7 1f 19b 8c d9 43 f2 246 8b dd 14b cb 3e 8c ca 88 160 94 97 d3 147 70 fd 202 fe 226 cb 3 8c ca 242 97 c5 159 204 fd 8a cb 13e 1f e0 a0 4d f9 64 e9 66 3f ba 8c e8 62 bf 89 cb 3 1f9 147 70 fd b4 fe 159 cb 7d 39 5e 191 fe 9a 1f 1b0 88 123 41 147 15d 18b 143 b7 a9 f2 1f9 e9 41 3f 226 141 160 41 1b0 d2 37 df 3 3f 105 a2 143 d3 159 17e fe 172 3f e5 141 160 1a8 211 f9 218 e9 da cb 3 1f 47 94 a9 6c fd 14b 133 1e6 94 97 c5 147 c7 20b 7c 12d 6b e0 4f f9 6c fd d4 fe 9a cb 13e 3f 105 8c e8 88 1e6 221 147 e5 57 d2 242 a9 94 97 6c fd 8a cb 6c 8c e8 242 a9 58 fe 3 3f 226 8c e8 88 160 41 3f ba 96 e9 70 fd 10d 1aa 195 bb c0 b4 1aa f3 159 e5 fe f5 18b 19b 1f9 19d 241 de 202 7b d3 159 64 1c9 249 76 1 f3 176 50 1b 241 f3 9f a2 1b ef 104 47 18b 88 24 14c 160 43 ea f3 1f 104 13 d8 5e 1bc 3f 1b cb e cb a7 64 f3 43 143 37 d4 a9 7d 191 212 171 0 1a be 105 10 171 0 b3 50 fe 20b cb 124 be f3 159 100 b9 a9 1c7 df 1ba 171 d9

Dari *string hex* tersebut, dapat dilakukan *write* sehingga menghasilkan kembali gambar foto KTP awal.

Kesimpulan dan Saran

Pada makalah ini, diajukan solusi berupa penggabungan algoritma RSA dan steganografi untuk menjaga keamanan foto KTP. Kelebihan dari penggabungan algoritma RSA dan steganografi adalah peningkatan keamanan yang tinggi, karena sulit untuk memecahkan algoritma RSA untuk mendapatkan pesan awal, juga adanya tambahan steganografi sehingga menyembunyikan pesan enkripsi tersebut. Akan tetapi, terdapat kekurangan yaitu hasil enkripsi algoritma RSA yang dapat memakan *storage* cukup besar dan diperlukannya *cover-object* (gambar) yang cukup besar untuk menyembunyikan hasil enkripsi RSA tersebut dengan baik. Ke depannya, diharapkan solusi ini dapat dikembangkan dan diperbaiki lagi, seperti pemilihan ekstensi *cover-object* berupa gambar yang lebih variatif (bukan hanya PNG) dan/atau tampilan GUI yang lebih interaktif dan menarik.

References

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

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

[3] 2022, "Stegano Documentation". URL: https://stegano.readthedocs.io/_/downloads/en/latest/pdf/ . Diakses pada 25 Mei 2022.

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, 25 Mei 2022

Fernaldi Fauzie, 18219099