Perbandingan Citra Digital Sebelum dan Sesudah Melakukan Kombinasi Proses Enkripsi Menggunakan Algoritma RC4 dengan Metode Steganografi *Least Significant Bit* (LSB)

Clarisa Gita Aprillia[1], Aditya Prapanca[2]

[1,2] Jurusan Teknik Informatika/Teknik Informatika, Universitas Negeri Surabaya
[1]clarisa.18078@mhs.unesa.ac.id
[3]adityaprapanca@unesa.ac.id

***Abstrak*— Citra digital sebagai salah satu format data digital yang digunakan untuk menyimpan gambar atau hasil karya dalam format digital. Namun, ada kemungkinan orang-orang yang tidak berwenang juga bisa melakukan kejahatan atau hal-hal yang tidak diinginkan. Oleh karena itu, dibutuhkan teknik kombinasi menggunakan algoritma simetris *Rivest Code 4* (RC4) pada proses enkripsi dan metode Steganografi *Least Significant Bit* (LSB) pada proses penyisipan pesan untuk menjaga keaslian suatu citra. Algoritma *Rivest Code 4* (RC4) adalah algoritma yang memiliki kunci simetris dan mengenkripsi *plaintext* secara *digit* per *digit* atau *byte* per *byte* dengan cara mengkombinasikan dengan operasi biner (biasanya XOR) dengan sebuah angka semi acak. Adapun Steganografi *Least Significant Bit* (LSB) adalah Metode Steganografi pada domain spasial dimana setiap bit yang mempunyai nilai minimum pada file citra diganti dengan urutan biner dari pesan rahasia. Pada penelitian ini, proses perbandingan dilakukan dengan cara memasukkan data citra dengan format (*JPG, *PNG, *BMP) pada proses enkripsi dan penyisipan pesan kemudian diproses menggunakan 6 variabel (PRNG *Password*, *Convert* teks menjadi *ciphertext*, *ciphertext* menjadi biner, *convert* gambar ke RGB, *convert* RGB ke nilai biner, menyisipkan biner *ciphertext* ke biner RGB sesuai posisi PRNG) agar dapat direalisasikan kedalam sistem. Adapun hasil pengujian perbandingan yang didapat yaitu semakin banyak karakter teks yang dimasukkan dalam proses *encode*, maka semakin banyak juga nilai biner yang muncul dan dari ketiga format citra yang diuji. Hasil pengujian juga menunjukkan bahwa tidak merubah kualitas citra (perubahan piksel tidak terlihat) akan tetapi merubah kualitas ukuran dengan nilai peningkatan 56.2% pada *JPG dan 1% pada *PNG serta penurunan sebesar 33.3% pada *BMP.**

***Kata Kunci*— Citra Digital, Kriptografi, Steganografi, Algoritma RC4, LSB**

I. PENDAHULUAN

Citra merupakan suatu representasi yang terjadi di dunia. Pada saat ini, citra memiliki nilai guna yang sangat luas penggunaannya sebagai data digital. Penyampaian informasi menggunakan media internet tentu akan memberikan kemudahan bagi sebagian besar pengguna dalam berbagai hal. Namun, sisi lain dari perkembangan teknologi bukan hanya memberikan dampak yang baik saja, tetapi memberikan dampak buruk juga seperti beberapa ancaman yang dilakukan oleh para peretas untuk melakukan penyadapan informasi, manipulasi informasi dengan tujuan tertentu sehingga menyebabkan berbagai masalah seperti kebocoran data atau hal-hal yang tidak diinginkan lainnya. Oleh karena itu, upaya untuk menjaga keamanan infomasi atau keamanan pesan merupakan aspek terpenting untuk citra agar tetap menjadi aset berharga yang tidak dapat dilihat atau bahkan dimiliki selain orang yang berhak. Salah satu teknik keamanan data yang sangat membantu mengatasi masalah tersebut adalah teknik kriptografi dan teknik steganografi sehingga pesan atau data dapat ditransformasi (makna tersembunyi) dan mengubahnya menjadi sebuah pesan lain yang mempunyai makna atau pesan yang dapat dimengerti kemudian disisipkan dan dimasukkan ke dalam file citra [1].

Kriptografi dan Steganografi merupakan 2 teknik yang masih berhubungan dengan *spycraft*. Perbedaan paling dasar antara keduanya terdapat di proses penyembunyian informasi dan hasil akhir. Dalam teknik kriptografi, terdapat 2 proses utama, yaitu enkripsi dan dekripsi. Enkripsi adalah proses penyandian pesan asli atau *plaintext* menjadi *ciphertext* (teks tersandi). Sedangkan dekripsi adalah proses penyandian kembali *ciphertext* menjadi *plaintext*. *Rivest Code 4* (RC4) merupakan algoritma simetris yang terdapat dalam Teknik kriptografi. *Rivest Code 4* (RC4) dapat disebut *synchrone stream cipher*, yaitu *cipher* yang memiliki kunci simetris dan mengenkripsi *plaintext* secara *digit* per *digit* atau *byte* per *byte* dengan cara mengkombinasikan dengan operasi biner (XOR) dengan sebuah angka semi acak. [2].

Sedangkan teknik steganografi digunakan untuk penyembunyian pesan rahasia supaya orang awam tidak menyadari keberadaan dari pesan yang disembunyikan [3]. Steganografi menyisipkan pesan kedalam citra sampel agar perbedaannya tidak terlihat secara visual antara citra asli dengan citra yang sudah disisipi pesan sehingga tidak akan diketahui oleh orang yang mampu memecahkan *stego image* tanpa mengetahui kunci [4] [5]. Salah satu metode steganografi citra digital adalah *Least Significant Bit* (LSB) dimana tiap bit yang mempunyai nilai paling kecil pada deret file citra diganti dengan deret biner pesan rahasia [6]. Metode *Least Significant Bit* (LSB) telah banyak dipakai dalam berbagai penelitian karena perhitungannya tidak terlalu rumit dan pesan yang disembunyikan cukup aman sehingga tidak seorang pun kecuali pengirim dan penerima pesan dapat mendeteksi keberadaan pesan tersebut.

Penelitian yang dilakukan oleh Dian Novianto (Novianto, Dian 2018) dengan judul "IMPLEMENTASI KEAMANAN BERKAS MENGGUNAKAN TEKNIK STEGANOGRAFI

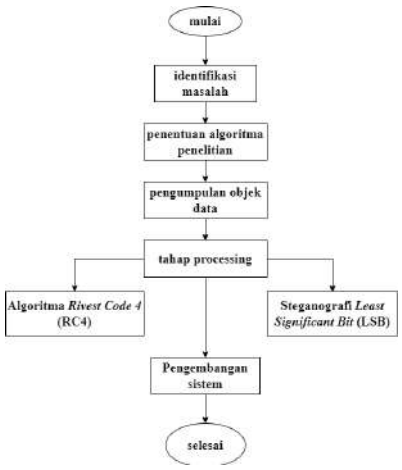
DAN ALGORITMA KRIPTOGRAFI MENGGUNAKAN METODE LEAST SIGNIFICANT BIT (LSB) DAN ALGORITMA RIVEST CODE 4 (RC4).” Dimana penelitian tersebut menggabungkan algoritma RC4 dengan Metode LSB dalam proses pengamanan pada berkas dengan cara menggabungkan kedua proses tersebut kedalam form dan memanfaatkan fungsi class pada bahasa pemrograman visual basic 6.0. untuk memproses perhitungan matematis pada metode LSB dan kriptografi RC4 dapat berjalan dengan baik. Hasil dari penelitian tersebut menunjukkan bahwa proses penyisipan di metode LSB dilakukan dengan menyisipkan pesan pada file objek tidak mengalami perubahan ukuran dan tidak merubah kualitas gambar secara kasat mata. Hal ini dapat dilihat dari hasil pengujian data [7]. Perbedaan mendasar antara referensi penelitian diatas dengan penelitian ini terletak pada file yang diuji dimana penelitian diatas menggunakan file dokumen dan penelitian ini menggunakan file citra serta menggunakan Bahasa pemrograman *Phyton*.

Berdasarkan studi pendahuluan tentang beberapa cara mengenai enkripsi dan dekripsi, Algoritma RC4 dinilai sangat cepat pada prosesnya kurang lebih 10 kali lebih cepat dari DES [8]. Selain itu kombinasi antara Kriptografi dan Steganografi terbilang efisien untuk memperkuat keamanan data dimana citra telah terenkripsi dan tidak terbaca tetapi tidak menutup fakta bahwa mereka mengirim rahasia.

Tujuan dari penelitian ini adalah menggabungkan kedua teknik tersebut dalam proses analisa beberapa format file citra asli dengan file citra yang telah di enkripsi algoritma RC4 dan disisipi pesan LSB untuk mengetahui perbandingannya dari segi ukuran file, RGB, dan waktu yang diperlukan selama proses tersebut berjalan.

II. METODE PENELITIAN

Dalam penelitian ini, penulis menggunakan metode kuantitatif dengan cara penulis mengumpulkan objek data dalam melakukan perhitungan dengan tujuan pengujian kualitas hasil perbandingan dari citra sampel atau citra asli dengan citra yang telah dienkripsi menggunakan Algoritma *Rivest Code 4* (RC4) dan disisipi dengan pesan Steganografi *Least Significant Bit* (LSB).



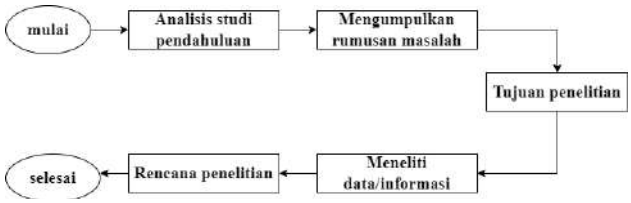
Gambar 1. Tahap Penelitian

Penelitian ini memiliki beberapa tahapan untuk mencapai hasil yang maksimal sesuai dengan uraian yang ditunjukkan pada Gambar 1.

Adapun beberapa tahapan tersebut sebagai berikut:

A. Identifikasi Masalah

Pada tahapan identifikasi masalah, penulis menganalisis masalah apa saja yang akan terjadi serta mencari, mengumpulkan, meneliti informasi yang diperlukan selama proses pengenkripsian dan penyisipan pesan pada file citra yang ditunjukkan pada Gambar 2.



Gambar 2. Identifikasi Masalah

B. Penentuan Algoritma Penelitian

Pada tahapan penentuan algoritma penelitian, penulis akan mengkombinasikan 2 teknik yaitu enkripsi menggunakan Algoritma *Rivest Code 4* (RC4) dan penyisipan pesan menggunakan metode Steganografi *Least Significant Bit* (LSB)

C. Pengumpulan Objek Data

Dalam tahap pengumpulan objek data, penulis mendapatkan berbagai sumber referensi seperti jurnal dan artikel yang masih berhubungan dengan metode yang diambil. Untuk sumber objek data penelitian ini mengambil dari situs resmi www.petitcolas.net. dengan menggunakan 3 format citra yang berbeda yaitu

1) *JPG *(Joint Photographic Group)*
   Merupakan format yang sama dengan *JPEG (*Joint Photographic Expert Group*). Yang membedakan keduanya hanya terletak pada jumlah hurufnya saja. *JPG juga sangat serimg kita jumpai dikarenakan ukurannya yang kecil sehingga *portable* untuk proses enkripsi pada citra digital.
2) *BMP *(Bitmap)*
   Merupakan gambar yang terdiri dari kumpulan titik–titik dan memiliki warna sendiri sehingga *BMP memiliki ukuran cukup besar dikarenakan kualitas gambar tergantung pada resolusi atau piksel yang terdaapat pada gambar tersebut.
3) *PNG *(Portable Network Graphics)*
   Pada *PNG, kompresi yang dilakukan tidak akan menghilangkan data atau *lossless compression.*

Pada pengujian 3 format citra diatas menggunakan ukuran yang sama yaitu resolusi 200x200 piksel. Variabel yang digunakan pada penelitian sebanyak 6 variabel yang akan dimunculkan sebagai penilaian dari objek data yang terkumpul seperti pada Tabel 1.

TABEL 1

ATRIBUT OBJEK DATA

Atribut	Deskripsi	Tipe Data
PRNG *password*	PRNG (*Pseduo Random Number Generator*) sebagai pembangkit bilangan acak yang berlaku sebagai kunci.	*Integer*
Convert text ke c*iphertext* dengan algoritma RC4	Mengubah *plaintext* ke *ciphertext* menggunakan Algoritma RC4	*Integer*
Convert ciphertext ke biner	Mengubah *ciphertext* menjadi biner	*Integer*
Convert gambar ke RGB	Mengubah gambar ke RGB	*Integer*
Convert RGB gambar ke biner	Mengubah RGB ke biner	*Integer*
Menyisipkan biner *ciphertext* ke biner RGB gambar sesuai hasil posisi PRNG	Penyisipan pesan dengan LSB	*Integer*

D. Citra Digital

Galuh Adjeng Sekarsari, dkk melakukan penelitian pada tahun 2015 dengan judul "ANALISIS ALGORITMA KRIPTOGRAFI RC4 PADA ENKRIPSI CITRA DIGITAL". Mereka mengatakan bahwa citra digital merupakan baris dan kolom yang terdapat pada suatu matriks, dimana setiap pasangan indeks baris dan kolom tersebut dapat menyatakan titik pada citra. Nilai dari matriks tersebut menyatakan nilai kecerahan titik dari suatu citra. Sekumpulan titik yang terdapat pada citra disebut sebagai elemen citra atau *pixel* (*picture element*) yang memiliki koordinat (x,y) sehingga menunjukan intensitas dari citra tersebut [8].

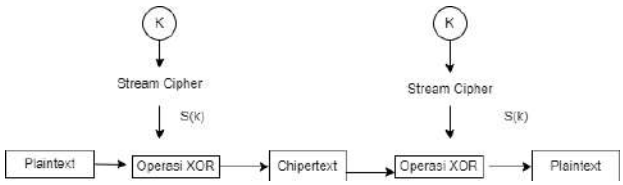
Citra digital terbagi menjadi tiga jenis warna yang berbeda yaitu RGB (*Red, Green, Blue*), *Grayscale* (abu-abu) dan Biner (hitam dan putih). Citra digital tidak hanya dibuat untuk merekam data dari sistem, namun dapat juga rekaman data yang sifatnya berkelanjutan seperti gambar TV, foto dan lain sebagainya. Adapun salah satu contoh citra digital berupa foto seperti yang ditunjukkan oleh Gambar D. dibawah ini:

Gambar D. *Wildflowers*

Access from: www.petitcolas.net.

E. Algoritma *Rivest Code 4* (RC4)

Algoritma RC4 adalah Algoritma Simetris yang masuk kedalam *stream cipher*. Oleh karena itu, dalam proses pengenkripsi serta pendekripsian menggunakan kunci yang sama atau bisaa disebut dengan *single key*. RC4 memproses data dalam ukuran byte (1*byte* = 8 bit) yang ditunjukkan pada Gambar 3.



Gambar 3. Cara Kerja Algoritma RC4

Cara proses enkripsi dari Algoritma RC4 ditunjukkan pada Gambar 4. dimulai dengan:

1. Input *plaintext* dan kunci
2. Inisialisasi *array* S (dengan indeks 0 sampai 255) sehingga $S[0] = 0, S[1] = 1 \ldots S[255] = 255$
3. Inisialisasi *secret key* atau *array* T. dibutuhkan Panjang yang sama antara *array* S dan *array* T agar algoritma dapat berjalan. Jika Panjang kunci < 256, lakukan *padding* atau pengulangan *byte* sampai Panjang 256.
4. *Key Scheduling Algorithm* (KSA). Pada perhitungan KSA dimulai dengan menentukan kata, indeks, S[i], dan kunci. Selanjutnya inisialisasi i dan j dengan 0, lakukan KSA agar menghasilkan *state array* secara acak.

$for\ (i = 0\ ; i \leq 255; i++)\{$
$S[i] = i$
$T[i] = kunci\ [i\ mod\ panjang\ kunci]$
$\}$

$permutasi\ array\ S - Box$
$\mathrm{j} = 0\ \mathrm{for}\ (\mathrm{i} = 0; \mathrm{i} \leq 255; \mathrm{i}++)\{$
$j = (j + S[i] + T[i])\ mod\ 256$
$Swap\ (S[i], S[j])$
$\}$

Perhitungan manual *Key Scheduling Algorithm* (KSA) diawali dengan menentuka S-Box seperti pada Tabel 2.

TABEL 2

MENENTUKAN S-BOX

Kata	U	N	E	S	A
Indeks	1	2	3	4	5
S [i]	0	1	2	3	4
Key	4	3	2	1	0

Pada Tabel 3. Inisialisasi i dan j dengan 0, sehingga menghasilkan *state array* acak

- iterasi 1

$i = 0$
$j = (j + S[0] + K[0]) mod\ 5$
$= (0 + 0 + 4) mod\ 5 = 4$
$Swap(S[0], S[4])$

TABEL 3

HASIL ITERASI 1

S[i]	4	1	2	3	0
Key	4	3	2	1	0

- iterasi 2

$i = 0$
$j = (4 + 1 + 3) mod\ 5 = 3$
$Swap(S[1], S[3])$

Pada Tabel 4. Menunjukkan hasil iterasi kedua, *state array* yang berbeda dengan sebelumnya.

TABEL 4

HASIL ITERASI 2

S[i]	4	3	2	1	0
Key	4	3	2	1	0

Setelah itu lakukan iterasi selanjutnya sampai memenuhi jumlah *array* S (S-Box) dan *array* kunci (T) yang diperlukan.

5. Tahapan selanjutnya adalah (*Pseudo-Random Generator Algorithm*) PRGA untuk menentukan *key stream*. Pada tahap PRGA dilakukan perulangan dengan cara mengambil jumlahnya sama dengan karakter plainteks yang tersedia

Untuk mencari *key stream*, cari nilai i dan j

$i = 0$
$j = i\ for\ (i = 0; i \leq jumlah\ karakter; i++)$
{
$i = (i + 1) mod\ 256$
$j = (j + S[i]) mod\ 256$
$Swap\ (S[i], S[j])$
$t = (S[i] + S[j]) mod\ 256$
$Kunci[i] = S[t]$
}

Pada Tabel 5. Perhitungan PRGA dilakukan sebanyak 5 kali karena *plaintext* yang akan dienkripsi sebanyak 5 karakter menggunakan hasil *array* dari perhitungan sebelumnya.

TABEL 5

HASIL S ARRAY

Hasil S array					
S[i]	0	3	2	0	4
Key	4	3	2	1	0

Pada Tabel 6. Mencari nilai i dan j

- Iterasi 1

$i = (0 + 1) mod\ 5 = 1$
$j = (0 + 3) mod\ 5 = 3$
$Swap\ (S[1], S[3])$
$K_1 = (S[1] + S[3]) = (0 + 3) mod\ 5 = 3$
$K_1 = 0000\ 0011$

TABEL 6

HASIL ITERASI 1

S[i]	0	0	2	3	4
Key	4	3	2	1	0

Pada Tabel 7. Mecari nilai i dan j

- Iterasi 2

$i = (1 + 1) mod 5 = 2$
$j = (3 + 0) mod 5 = 3$
$Swap(S[2], S[3])$
$K_2 = (S[2] + S[3]) = (3 + 2) mod 5 = 0$
K_2=0000 0000

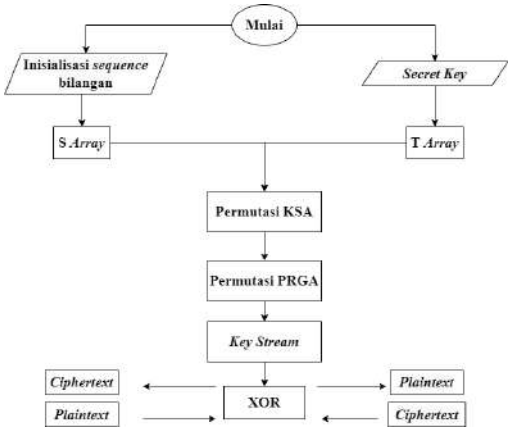
TABEL 7

HASIL ITERASI 2

S[i]	0	0	3	2	4
Key	4	3	2	1	0

6. Tahap yang terakhir dilakukan adalah proses operasi XOR antara *key stream* dan *plaintext* sehingga terjadi proses enkripsi.
7. Menentukan *array* S membutuhkan 2 kunci yang sama kemudian dimasukkan ke *stream cipher* atau algoritma yang sama sehingga menghasilkan *key stream* S(k). *stream key* akan membentuk suatu

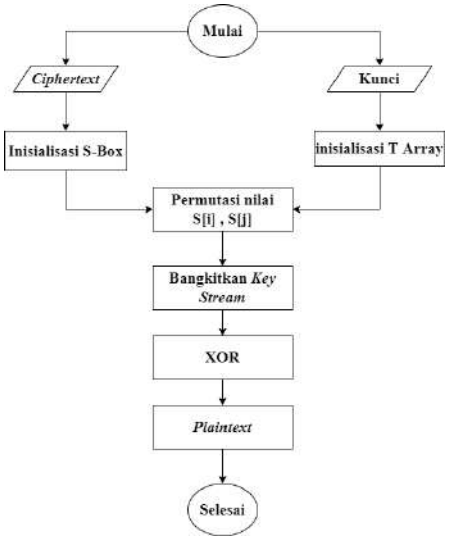
sequence yang dipakai untuk operasi XOR. XOR melakukan operasi dari *plaintext* dan *key stream* sehingga menghasilkan *ciphertext* (proses enkripsi).



Gambar 4. Proses Enkripsi Algoritma RC4

Adapun proses pengembalian atau (proses dekripsi) ditunjukkan pada Gambar 5. yang dimulai dengan:

1. Input *ciphertext* dan kunci
2. Inisialisasi Sbox $(S[0], S[1], \dots, S[255])$, maka format Sbox adalah $S[0] = 0, S[1] = 1, \dots, S[255] = 2555$.
3. Inisialisasi *array* kunci (panjang 256). Jika panjang kunci < 256 maka dilakukan *padding*, Sehingga *array* kunci K berbentuk $K[0], K[1], \dots, K[255]$.
4. Permutasi terhadap nilai -nilai yang ada dengan melakukan penukaran pada isi *array* S[i] , S[j]
5. Membangkitkan *key stream* (k) dari nilai S[i] dan S[j] selanjutnya jumlah dan mod 256
6. Lakukan operasi XOR antara *ciphertext* dan *key stream* dan ubah operasi XOR menjadi karakter ASCII untuk pengembalian *plaintext*.



Gambar 5. Proses Dekripsi Algoritma RC4

F. *Least Significant Bit* (LSB)

Least Significant Bit merupakan teknik steganografi substitusi yang digunakan dalam proses steganografi berbasis media (*media-based steganography*) [8]. Teknik Steganografi dengan mengubah LSB dilakukan dengan mengubah bit LSB pada setiap *byte* piksel citra. LSB piksel tersebut diganti dengan bit informasi pesan yang akan disisipkan. Metode LSB dilakukan dengan cara mengubah nilai bit terkecil yang terletak pada barisan paling kanan dari bit data. Sebagai contoh, bit pada gambar dengan ukuran 3 piksel ditunjukkan pada Tabel 8 sebagai berikut:

TABEL 8

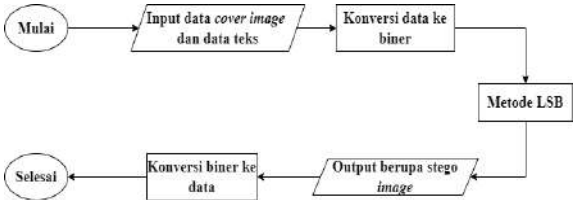
UKURAN PIKSEL PADA GAMBAR

01000111	10000011	10000011
00110010	01100110	01100110
00100100	01011101	01011010

Pesan yang akan disisipkan adalah karakter “C” yang memiliki biner 00001111, *stego image* yang akan dihasilkan adalah:

(0100011**0** 1000001**0** 1000001**0**)
(0011001**0** 0110011**1** 0110011**1**)
(0010010**1** 0101110**1** 0101101**0**)

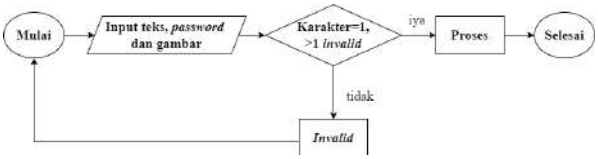
Bilangan yang terdiri dari bit biasanya ditandai dengan huruf "b" diakhir bilangan misalnya 01000111 menjadi 0100011**b**. Huruf huruf b yang terletak diakhir bilangan memiliki arti biner atau bit yang ditunjukkan pada Gambar 6.



Gambar 6. Penyisipan Pesan dengan LSB

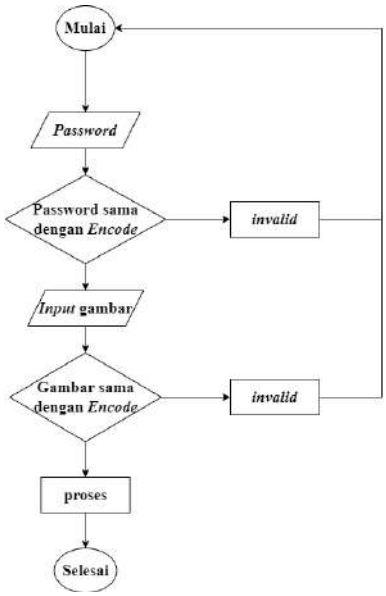
G. *Encode* dan *Decode*

1. *Encode* atau dalam istilah lain yaitu pembuatan pesan dimana *encode* berperan mengubah cara data yang ingin kita munculkan kepada publik dapat dipahami dengan baik. Proses *encode* pada penelitian ini yaitu dimulai dengan memasukkan teks (karakter=1), *password* dan file citra yang diinginkan. Setelah itu sistem akan memproses data-data tersebut dan menampilkan beberapa variabel yang dibutuhkan pada proses perbandingan file citra yang ditunjukkan pada Gambar 7.



Gambar 7. *Encode*

2. *Decode* merupakan kebalikan dari *encode* yaitu proses mengubah kode yang tidak dimengerti maknanya menjadi ke teks awal. Proses *decode* yang terjadi pada penelitian ini yaitu dengan memasukkan *password* dan file citra yang sama saat melakukan proses *encode* sebelumnya. Setelah itu sistem menampilkan file citra serta waktu eksekusi pada proses *decode* sebagai bukti bahwa file citra tersebut memang ada. Ditunjukkan pada Gambar 8.



Gambar 8. *Decode*

H. Pengembangan Sistem

Pada tahapan pengembangan sistem, penulis membuat rancangan yang memiliki tujuan dari pembuatan artikel ilmiah ini sehingga dapat menjadikan sistem perbandingan file citra. Adapun 2 macam dalam pengembangan sistem yaitu analisis sistem dan desain sistem:

1) Analisis Sistem

a. Identifikasi masalah

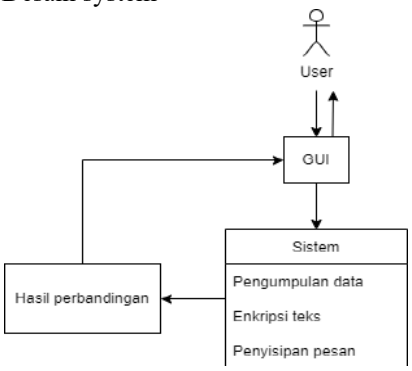
Berdasarkan permasalahan tersebut, maka diketahui bagaimana merealisasikan sistem perbandingan file citra sebelum dan sesudah melakukan enkripsi Algoritma *Rivest Code 4* (RC4) dan disisipi pesan menggunakan metode *Least Significant Bit* (LSB).

b. Analisis kebutuhan fungsional

Sistem perbandingan file citra dalam penelitian ini dapat melakukan beberapa hal antara lain:

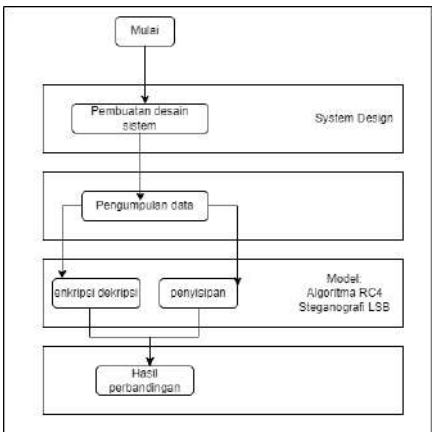
- Pengguna dapat memasukkan teks sebelum proses *encode* dilakukan
- Pengguna dapat *password* sebelum proses *encode* dilakukan
- Pengguna dapat memilih file citra apa saja yang diinginkan sebelum proses *encode* dilakukan
- Pengguna dapat memasukkan password sebelum proses *decode* dilakukan (*password* harus sama dengan proses *encode*)
- Pengguna dapat memasukkan file citra sebelum proses *decode* dilakukan (file citra harus sama dengan proses *encode*)

2) Desain system



Gambar 9. Desain sistem

Berdasarkan Gambar 9. Desain Sistem diatas dijelaskan bahwa proses awal penelitian ini adalah pengguna diarah ke program untuk menampilkan sistem bagian pengumpulan data yang berisi *input* teks, *password* dan file citra atau gambar. Kemudian sstem akan memproses data tersebut menggunakan enkripsi Algoritma RC4 dan penyisipan pesan LSB. Setelah input data dan proses dalam system telah selesai, sistem tersebut akan menampilkan hasil perbandingan dari file citra sebelum dan sesudah dilakukan enkripsi dan penyisipan pesan.



Gambar 10. Alur sistem

Pada Gambar 10. Menunjukkan alur sistem diatas, menggambarkan sistem yang dijalankan oleh pengguna dimana proses tersebut dimulai dengan pengumpulan data berupa teks, *password*, file citra. setelah itu dilanjutkan dengan proses enkripsi Algoritma *Rivest Code 4* (RC4) dan penyisipan pesan *Least Significant Bit* (LSB), proses pengujian file citra asli dengan citra hasil *stego* (*stego image*), dan sistem akan menampilkan hasil perbanding tersebut.

III. HASIL DAN PEMBAHASAN

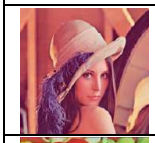
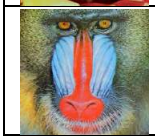
Pada penelitian ini, penulis menghasilkan sistem yang melakukan perbandingan citra asli dengan citra yang telah terenkripsi Algoritma RC4 dan dikombinasikan dengan metode LSB. Adapun 5 proses yang harus dijalankan yaitu data gambar penelitian, perencanaan enkripsi pada file citra, penyisipan pesan, proses *encode decode* dan hasil pengujian perbandingan dari citra tersebut.

A. Data Gambar Penelitian

Pada tahap pertama, yang menjadi penentuan adalah data gambar dimana gambar tersebut akan di uji coba pada proses enkripsi kemudian disisipi pesan dengan metode LSB dan dikembalikan melalui proses dekripsi. Data gambar yang digunakan dalam penelitian ini memiliki 3 format yang berbeda tetapi dengan resolusi yang sama. Berikut 3 data gambar uji coba yang ditunjukkan pada Tabel 9.

TABEL 9

DATA GAMBAR UJI COBA

Gambar	Nama File	Resolusi	Ukuran	Format
	Lena	200x200	35.3 KB	JPG
	Peppers	200x200	117 KB	BMP
	Baboon	200x200	91.8 KB	PNG

B. Proses perencanaan enkripsi pada file citra

Pada proses enkripsi file citra, sistem menggunakan Bahasa pemrograman *phyton* sehingga pada *code* yaitu app.py
Untuk proses pemanggilan fungsi, ditunjukkan pada Gambar 11.

```
PS D:\rc4-lsb baru> py app.py
 * Serving Flask app 'app' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
```

Gambar 11. menjalankan fungsi

Pada Gambar 12 dan Gambar 13. Proses awal yang dijalankan adalah *encode* (pra proses 1) dengan mengambil gambar dan disimpan pada folder *assets*

```
def encode():
    # pra proses 1
    time_execute = time.time()
    picture = request.files['picture']
    savepath = os.path.join("assets/img/"+picture.filename)
    picture.save(savepath)
```

Gambar 12. *Script* menyimpan citra

Gambar 13. penyimpanan gambar

Pada pra proses 2, dari gambar yang disimpan akan dibaca ulang. Jadi pada pra proses 2 memiliki variabel gambar, teks dan *password* dengan masing-masing rumus yang ditunjukkan pada Gambar 14.

```
# pra proses 2
image = cv2.imread("assets/img/" + picture.filename)
message  = request.form['hidden_text']
password = request.form['password']
```

Gambar 14. fungsi pengulangan

Pada Gambar 15 dan Gambar 16. *convert* PRNG untuk menentukan posisi *password* pada gambar dan RC4 untuk menentukan *ciphertext*.

```
# convert prng and rc4
position = prng(password, image.shape[0])
chipper  = rc4(password, message)
```

Gambar 15. *script* enkripsi dan rumus

```
# convert key and data to chippertext with ksa and prga method
def rc4(key, data):
    x = 0
    box = list(range(256))
    for i in range(256): #ksa
        x = (x + box[i] + ord(key[i % len(key)])) % 256
        box[i], box[x] = box[x], box[i]
    x = y = 0
    out = []
    for char in data: #prga
        x = (x + 1) % 256
        y = (y + box[x]) % 256
        box[x], box[y] = box[y], box[x]
        out.append(chr(ord(char) ^ box[(box[x] + box[y]) % 256]))

    return ''.join(out)
```

Gambar 16. *script* enkripsi dan rumus

C. Proses penyisipan pesan

Pada Gambar 17 sampai dengan Gambar 20. Terdapat beberapa tahapan dalam proses penyisipan pesan yaitu pesan yang menjadi *ciphertext* akan dimasukkan ke gambar, *input secret* data dan *convert* ke biner.

```
# encode to lsb and save to img_lsb as image
encode  = lsb(image, chipper, position)
cv2.imwrite("assets/img_lsb/"+ picture.filename.split('.')[0] + '.png' ,encode)
```

Gambar 17. *script* penyisipan dan rumus

```
# insert lsb in image
def lsb(image, secret_data, prn):
    secret_data += "#####"

    binary_data = data2binary(secret_data)

    index     = 0
    for binary in binary_data:
        position1 = prn[index]

        pixel = image[0][position1]
        r,g,b = data2binary(pixel)

        image[0][position1][0] = int(r[:-1] + binary, 2)
        image[0][position1][1] = int(g[:-1] + binary, 2)
        image[0][position1][2] = int(b[:-1] + binary, 2)

        index += 1
```

Gambar 18. *script* penyisipan dan rumus

```
# convert data to binary
def data2binary(data):
    if type(data) == str:
        return ''.join([format(ord(i),"08b") for i in data])
    elif type(data) == bytes or type(data) == np.ndarray:
        return [format(i,"08b") for i in data]
```

Gambar 19. *script* penyisipan dan rumus

```
# check max text
maxText = int(image.shape[0] / 8) - 5 #rumus
if len(message) > maxText:
    error = "Gagal, maksimal karakter " + str(maxText) + ", jumlah karakter " + str(len(message))
    return render_template("home.html", error = error)
```

Gambar 20. *script* batas maks karakter

Pada Gambar 21 dan Gambar 22. Proses memanggil fungsi untuk mengambil data RGB file citra asli dalam bentuk angka dan diubah kembali ke RGB bentuk biner

```
        pixel = image[0][position1]
        r,g,b = data2binary(pixel)
```

Gambar 21. *script* RGB *image*

```
        image[0][position1][0] = int(r[:-1] + binary, 2)
        image[0][position1][1] = int(g[:-1] + binary, 2)
        image[0][position1][2] = int(b[:-1] + binary, 2)

        index += 1
```

Gambar 22. *script* RGB *image*

Pada Gambar 23 dan Gambar 24. File citra yang sudah melakukan *encode*, disimpan dalam folder

```
# encode to lsb and save to img_lsb as image
encode  = lsb(image, chipper, position)
cv2.imwrite("assets/img_lsb/"+ picture.filename.split('.')[0] + '.png' ,encode)
```

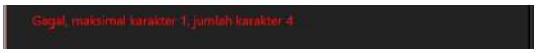
Gambar 23. *script* penyimpanan *stegoimage*

Gambar 24. penyimpanan *stegoimage*

D. Proses *Encode Decode*

1. Adapun beberapa tahapan yang dapat diuraikan pada proses *encode*, antara lain:
   a. Beranda (*Home Page*)

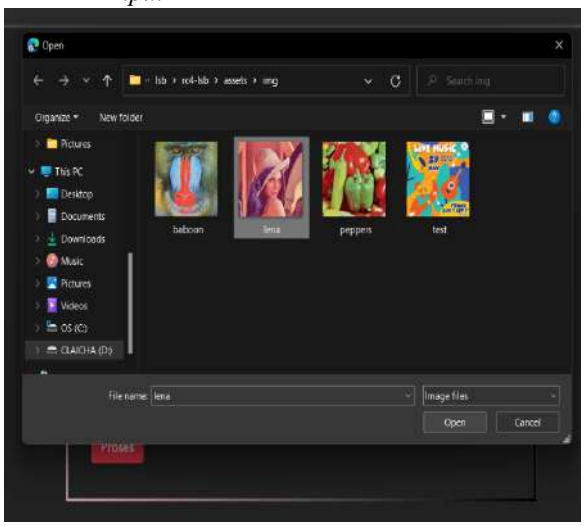
Gambar 25. *Home Page*



Gambar 26. *Home Page*

Gambar diatas merupakan tampilan beranda atau *home* dimana users dapat mengisi teks dan *password*. *Users* hanya dapat memasukkan 1 karakter saja. Sebagai contoh dimasukkan teks "L" *password* 123 maka akan muncul tampilan seperti pada Gambar 25. Ketika dimasukkan kata "Lena" dengan *password* 123 maka hasilnya *invalid* seperti Gambar 26.

b. *Input* file citra



Gambar 27. Input Citra

Gambar 27. menunjukkan proses pemilihan dan input file citra yang diinginkan. Sebagai contoh penulis memasukkan foto Lena kedalam sistem.

c. Hasil dari proses *encode*

Gambar 28. Hasil *Encode*

Gambar 29. Hasil *Encode*

Gambar diatas menampilkan hasil dari proses *endoce* dengan beberapa variabel yang dibutuhkan yaitu PRNG Password menentukan posisi biner, hasil *ciphertext*, pengubahan kode *ciphertext* ke bentuk biner, ubah RGB gambar ke bentuk biner, dan yang terakhir penyisipan biner *ciphertext* kedalam biner RGB gambar. yang ditunjukkan pada Gambar 28.

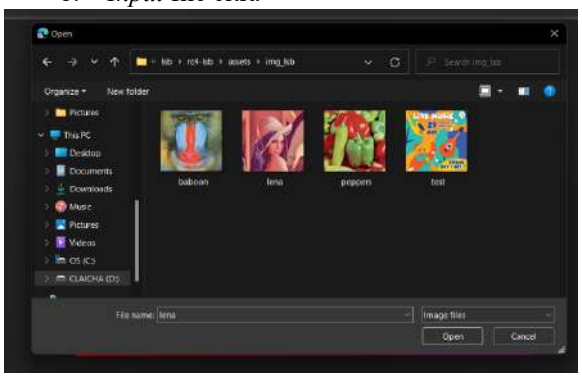
Selanjutnya hasil yang didapat dari beberapa variabel diatas adalah hasil perbandingan gambar sebelum dan setelah pengenkripsian dan penyisipan pesan, perubahan nilai RGB, perubahan biner RGB gambar dan waktu eksekusi proses *encode* yang ditunjukkan pada Gambar 29.

2. Adapun beberapa tahapan yang dapat diuraikan pada proses decode, antara lain:
   a. Beranda (*home page*)

Gambar 30. *Home Page*

Gambar 30. merupakan tampilan beranda atau home decode dimana users dapat mengisi ulang password dengan catatan *password* yang dimasukkan harus sama pada saat users melakukan proses *encode*.

   b. *Input* file citra



Gambar 31. Input citra

Gambar 31. menunjukkan proses pemilihan dan *input* file citra dengan catatan file citra harus sama pada saat *users* melakukan proses *encode* sebagai pembuktian bahwa data yang dimasukkan adalah benar.

Gambar 32. Hasil *Decode*

Gambar 32. menampilkan hasil dari proses *decode* dimana sistem akan menampilkan hasil *decode*, file citra asli dan waktu yang dibutuhkan.

E. Hasil pengujian perbandingan file citra

Perbandingan file citra pada saat sebelum dan sesudah dilakukan kombinasi proses enkripsi Algoritma RC4 dengan Metode Steganografi LSB merupakan proses terakhir pada penelitian ini, maka terdapat beberapa aspek penilaian yang perlu diuraikan yaitu penampilan citra asli dengan *stego image*, ukuran file citra, perubahan RGB, dan waktu eksekusi yang ditunjukkan pada Tabel 10 sampai dengan Tabel 12.

TABEL 10

PENGUJIAN FILE CITRA FORMAT *JPG

Hasil perbandingan	Sebelum proses kombinasi	Setelah proses kombinasi
Penampilan citra		
Ukuran file citra	35 KB	80 KB
RGB	Red Green Blue 111 121 222 104 108 210 94 100 205 113 131 222	Red Green Blue 110 120 222 104 108 210 95 101 205 113 131 223
Waktu eksekusi	0.05998945236206055 detik	

Pada Tabel 10. Dijelaskan bahwa pengujian file citra dengan format *JPG dari segi penampilan citra secara visual menghasilkan perbandingan yang tidak terlihat, namun dari segi ukuran file mengalami perubahan dari 35 KB menjadi 80 KB sehingga diperoleh peningkatan kualitas ukuran sebesar 56.2%. Kemudian pada nilai RGB juga mengalami perubahan yaitu RGB baris pertama citra asli yang masih dalam bentuk angka memiliki hasil dengan 3 piksel *Red* bernilai 100, *Green* bernilai 121 dan *Blue* bernilai 222 berbeda dengan nilai RGB baris pertama yang sudah terenkripsi dan melakukan penyisipan pesan dengan nilai piksel *Red* bernilai 110, *Green* 120 dan *Blue* 222. Dan untuk pengujian file citra format *JPG membutuhkan waktu eksekusi 0.05998945236206055 detik.

TABEL 11

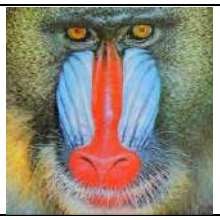
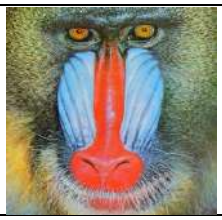
PENGUJIAN FILE CITRA FORMAT *BMP

Hasil perbandingan	Sebelum proses kombinasi	Setelah proses kombinasi
Penampilan citra		
Ukuran file citra	117 KB	78 KB
RGB	Red Green Blue 77 55 184 68 78 165 81 119 191 66 54 166	Red Green Blue 76 54 184 68 78 164 81 119 191 66 54 166
Waktu eksekusi	0.08344650268554688 detik	

Pada Tabel 11. Dijelaskan bahwa pengujian file citra dengan format *BMP dari segi penampilan citra secara visual menghasilkan perbandingan yang tidak terlihat, namun dari segi ukuran file mengalami perubahan dari 117 KB menjadi 78 KB sehingga diperoleh penurunan kualitas ukuran sebesar 33.3%. Kemudian pada nilai RGB juga mengalami perubahan yaitu RGB citra asli yang masih dalam bentuk angka memiliki hasil dengan 3 piksel *Red* bernilai 77, *Green* bernilai 55 dan *Blue* bernilai 184 berbeda dengan nilai RGB yang sudah terenkripsi dan melakukan penyisipan pesan dengan nilai piksel *Red* bernilai 76, *Green* 54 dan *Blue* 184. Dan untuk pengujian file citra format *BMP membutuhkan waktu eksekusi 0.08344650268554688 detik.

TABEL 12

PENGUJIAN FILE CITRA FORMAT *PNG

Hasil perbandingan	Sebelum proses kombinasi	Setelah proses kombinasi
Penampilan citra		
Ukuran file citra	91 KB	92 KB
RGB	Red Green Blue 69 139 133 90 97 70 156 155 127 137 144 123	Red Green Blue 68 138 132 90 96 70 157 155 127 137 145 123
Waktu eksekusi	0.07376933097839355 detik	

Pada Tabel 12. Dijelaskan bahwa pengujian file citra dengan format *PNG dari segi penampilan citra secara visual menghasilkan perbandingan yang tidak terlihat, namun dari segi ukuran file mengalami perubahan dari 117 KB menjadi 78 KB sehingga diperoleh peningkatan kualitas ukuran sebesar 1%. Kemudian pada nilai RGB juga mengalami perubahan yaitu RGB citra asli yang masih dalam bentuk angka memiliki hasil dengan 3 piksel *Red* bernilai 69, *Green* bernilai 139 dan *Blue* bernilai 133

berbeda dengan nilai RGB yang sudah terenkripsi dan melakukan penyisipan pesan dengan nilai piksel *Red* bernilai 68, *Green* 138 dan *Blue* 132. Dan untuk pengujian file citra format *PNG membutuhkan waktu eksekusi 0.07376933097839355 detik.

IV. KESIMPULAN

Berdasarkan hasil dan pembahasan sebelumnya, maka penelitian artikel ilmiah ini memperoleh beberapa kesimpulan sebagai berikut:

1. Sistem perbandingan file citra sebelum dan sesudah melakukan enkripsi menggunakan Algoritma RC4 dan disisipi pesan steganografi *Least Significant Bit* (LSB) berjalan baik dengan hasil tidak merubah kualitas antara citra asli dengan citra LSB atau perubahan piksel pada citra tidak terlihat.
2. Semakin banyak karakter teks yang dimasukkan dalam proses *encode*, maka semakin banyak juga nilai biner yang yang muncul.
3. Dari ketiga file citra yang sudah diuji, format *JPG dan *PNG mengalami perubahan yaitu ukuran file lebih besar dengan nilai akurasi 56.2% dan 1%. Berbeda dengan format *BMP yang mengalami perubahan ukuran menjadi lebih kecil dengan nilai akurasi 33.3% serta waktu eksekusi yang dibutuhkan lebih banyak 0.08344650268554688 detik.

V. SARAN

Berdasarkan penelitian yang telah dilakukan, tentu penulis maasih banyak mengalami kesulitan dan kekurangan dalam pengerjaan artikel ilmiah ini. Oleh karena itu, terdapat saran untuk menampilkan lebih banyak perbandingan file citra seperti perbedaan histogram dan lain-lain guna menunjang sistem.

UCAPAN TERIMA KASIH

Terimakasih kepada Tuhan Yang Maha Segala-Nya, karena telah memberikan kesempatan terlaksananya penelitian ini. Kepada orang tua atas kasih sayang dan doa yang selalu dipanjatkan kepada Tuhan. Kepada para Dosen Jurusan Teknik Informatika Unesa yang telah memberikan bimbingan dan ilmu yang bermanfaat. Tidak lupa juga kepada teman-teman Jurusan Teknik Informatika yang selalu memberikan dukungan dalam segala kesempatan.

REFERENSI

[1] Hozeng, Suryadi dan Sitti Aisa. 2015. “Implementasi Algoritma Rc4 Untuk Enkripsi Dan Deskripsi Citra”. PROSIDING SEMINAR ILMIAH SISTEM INFORMASI DAN TEKNOLOGI INFORMASI Pusat Penelitian dan Pengabdian Pada Masyarakat (P4M) STMIK Dipanegara Makassar. IV (02). 172-181

[2] Simbolon, Buha Johannes. 2021. “Steganografi Penyisipan Pesan Pada File Citra Menggunakan Metode LSB (Least Significant Bit)”. *Jurnal Nasional Komputasi dan Teknologi Informasi*. 4(1). 1-6

[3] Manurung, Modesty Sri Pebriyani. 2019. "Penerapan Algoritma Advanced Encryption Standard dalam Mengamankan File pada Citra dengan Metode Least Significant Bit". Jurnal Teknik Informatika Unika St. Thomas (JTIUST). 04(01). 62-69

[4] Hendrata, Agitiya Dwi dan Agus Prihanto. 2021. "Analisis Kualitas Suara Stego Audio Penyisipan Informasi Tersembunyi dengan Metode Least Significant Bit". *JINACS (Journal of Informatics and Computer Science)*. 02(03). 178-184

[5] Mido, Agus Rakhmadi dan Erik Iman Heri Ujianto. 2022. "ANALISIS PENGARUH CITRA TERHADAP KOMBINASI KRIPTOGRAFI RSA DAN STEGANOGRAFI LSB". *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*. 9(2): 279-286.

[6] Hameed, M. Sarab., Sa'adoon, A. Hiba., & Al-Ani, Mayyadah. 2018. "Image Encryption Using DNA Encoding and RC4 Algorithm". Iraqi Journal of Science. 59(1B). 434-446.

[7] Novianto, Dian. 2018. “IMPLEMENTASI KEAMANAN BERKAS MENGGUNAKAN TEKNIK STEGANOGRAFI DAN ALGORITMA KRIPTOGRAFI MENGGUNAKAN METODE LEAST SIGNIFICANT BIT (LSB) DAN ALGORITMA RIVEST CODE 4 (RC4)”. JUTIM STMIK MUSIRAWAS Lubuklinggau. 03(02). 92-98

[8] Adjeng, Galuh, dkk. 2015. “ANALISIS ALGORITMA KRIPTOGRAFI RC4 PADA ENKRIPSI CITRA DIGITAL”. Techno.COM. 14(04). 250-254

[9] Anas, Irfan, dkk. 2021. “Perancangan Aplikasi Keamanan Data Dengan Kombinasi Algoritma Kriptografi RC4 dan One Time Pad”. *JURIKOM (Jurnal Riset Komputer)*. 8(1): 20-27.

[10] Zebua, Taronishoki dan Eferoni Ndruru. 2017. “PENGAMANAN CITRA DIGITAL BERDASARKAN MODIFIKASI ALGORITMA RC4” Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK). 4(4). 275-282

[11] Taha, K. Zahraa., Alturfi, Mohammed Naser., Albaker, M. Baraa., Abduljalel, Y. W., Jasim, H. A., Majeed, I., . . . Abbas, M. G. 2019. "Robust and Secured Image Steganography using Improved LSB and RC4 Cryptography with Preprocessing Operation". Internation Journal of Recent Technology and Engineering (IJRTE). 8(2S9)

[12] Hakim, E. L., Khairil, K., & Utami, F. H. 2014. APLIKASI ENKRIPSI DAN DESKRIPSI DATA MENGGUNAKAN ALGORITMA RC4 DENGAN MENGGUNAKAN BAHASA PEMROGRAMAN PHP. JURNAL MEDIA INFOTAMA.10(1).1–7

[13] Febriyani, Fauziyah Suwarsita dan Arief Afriandi. 2021. Implementasi Algoritma RC4 pada Sistem Pengamanan Dokumen Digital Soal Ujian. JISKa. 06(03). 171-177