IMPLEMENTASI KRIPTOGRAFI ALGORITMA RIJNDAEL DAN STEGANOGRAFI METODE *END OF FILE* UNTUK KEAMANAN DATA

Dewi Kusumaningsih[1], Ahmad Pudoli[2], Iqbal Rahmadan[3]

[1, 2,3)]Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Budi Luhur
Jl. Raya Ciledug, Petukangan Utara, Kebayoran Lama, Jakarta Selatan 12260
Telp. (021) 5853753, Fax. (021) 5866369
[1]dewi.kusumaningsih@budiluhur.ac.id, [2]ahmad.pudoli45@gmail.com,[3]1311500803@budiluhur.ac.id

ABSTRAK

Teknologi dan informasi saat ini berkembang semakin pesat proses pertukaran data dan informasi dapat dilakukan dengan sangat mudah melalui berbagai macam media. Tentunya keamanan dalam pertukaran data dan informasi sangat diperlukan demi menjaga kerahasian data tersebut. Teknologi dan informasi saat ini berkembang semakin pesat proses pertukaran data dan informasi dapat dilakukan dengan sangat mudah melalui berbagai macam media. Tentunya keamanan dalam pertukaran data dan informasi sangat diperlukan demi menjaga kerahasian data tersebut. Dari permasalahan diatas didapat sebuah solusi dengan membuat sebuah aplikasi yang dapat menjaga kerahasiaan dari informasi ataupun data pada PT. Kothis Prima Mitra. Salah satu cara untuk menyembunyikan data tersebut dengan menggunakan teknik kriptografi dan steganografi. Teknik kriptografi yang digunakan adalah kriptografi algoritma Rijndael dengan melakukan enkripsi menjadi chiperteks, sehingga tidak dapat dibaca lagi. Untuk membacanya lagi perlu dilakukan proses dekripsi. Selain itu dilakukan pengamanan ganda dengan menggunakan teknik steganografi End Of File (EOF).Aplikasi ini menggunakan metode algoritma kriptografi Rijndael dengan kombinasi steganografi EOF (End of file) sehingga sangat optimal untuk mengamankan data. Dengan adanya Aplikasi ini diharapakan dapat mengamankan dan menjaga kerahasiaan data pada PT. Khotis Prima Mitra sehingga dapat mencegah terjadinya pencurian dan manipulasi data.

Kata Kunci : *Algoritma, Kriptografi, Rijndael, steganografi,* End Of File

I. PENDAHULUAN

1.1 Latar Belakang

Teknologi dan informasi saat ini berkembang semakin pesat proses pertukaran data dan informasi dapat dilakukan dengan sangat mudah melalui berbagai macam media. Tentunya keamanan dalam pertukaran data dan informasi sangat diperlukan demi menjaga kerahasian data tersebut. Bahkan masih ada saja pihak yang kurang sadar untuk menjaga kerahasiaan datanya, salah satu dampak negatif dalam perkembangan teknologi adalah adanya pencurian data. Bila hal itu sampai terjadi maka kemungkinan akan terjadi pemalsuan data, penyalahgunaan, penyadapan dan lain sebagainya.

PT. Kothis Prima Mitra adalah perusahaan yang bergerak di bidang Jasa *printing* emblem baju, sepatu, dan tas. Perusahaan besar seperti Nike, Adidas dan Mizuno pun memakai jasa PT. Kothis Prima mitra untuk mencetak logo atau emblem perusahaan tersebut. Dokumen transaksi serta pendapatan perusahaan harus terjaga kerahasiaannya dari pihak yang tidak berhak untuk mengaksesnya. Tetapi selama ini sistem PT. Kohtis Prima Mitra hanya melakukan pengamanan data secara sederhana, begitu juga saat mengirim dan menerima data yang penting, tanpa adanya keamanan untuk menjaga kerahasiaan data atau *file* yang telah dikirim atau disimpan oleh PT. Kothis Prima Mitra. Hal ini dapat menyebabkan terjadinya resiko dari pencurian data serta penyalahgunaan data tersebut. Dengan demikian sistem keamanan terhadap data atau *file* sangat diperlukan untuk menghindari tindakan-tindakan tersebut yang dapat merugikan.

Berdasarkan uraian diatas maka dalam penelitian tugas akhir ini penulis memilih algoritma kriptografi Rijndael dikombinasikan dengan steganografi dengan metode EOF dikarenakan teknik menyembunyikan atau menyisipkan file rahasia pada suatu media penampung, penyandian pada algoritma Rijndael ini memiliki mekanisme mengacak informasi dan menggunakan operasi XOR serta transformasi linier, sehingga sangat optimal untuk mengamankan data.

1.2 Tujuan Penulisan

Berdasarkan permasalahan yang telah didapat penulis maka Tujuan dari penulisan penelitian ini adalah membangun aplikasi untuk mengamankan data rahasia perusahaan dengan cara mengenkripsi dokumen dengan algoritma Rijndael kemudian menyisipkan dokumen/*file* tersebut kedalam sebuah gambar dengan menggunakan metode steganografi *End Of File* (EOF).

1.3 Batasan Masalah

Agar tidak keluar dari materi pembahasan maka akan diberikan beberapa batasan masalah sebagai berikut :

a. Penelitian ini hanya membahas kriptografi algoritma Rijndael dan steganografi EOF.
b. *File* gambar yang digunakan sebagai penampung berformat *.jpg dan *.png

c. *File* yang akan disisipkan pada aplikasi steganografi ini berekstensi .docx, .doc, .xlsx, .xls, dan .pdf.
d. *File* dokumen tidak lebih dari 1 Mb dan *file* gambar tidak lebih dari 1,5 MB.
e. Bahasa pemrograman yang digunakan adalah java.
f. Aplikasi ini hanya bisa berjalan pada dekstop.

II. LANDASAN TEORI

2.1. Steganografi

A. Sejarah Steganografi

Steganografi adalah seni komunikasi rahasia atau ilmu yang digunakan untuk menyembunyikan pesan rahasia sehingga selain orang yang dituju, tidak akan menyadari keberadaan dari pesan rahasia tersebut. Steganografi membutuhkan dua bagian yang sangat penting yaitu berkas atau media penampung dan data rahasia yang akan disembunyikan [1]. Steganografi berasal dari bahasa yunani yaitu *stegos* yang berarti penyamaran dan *graphia* yang berarti tulisan. pada saat itu penguasa Yunani, Histiaeus sedang ditawan oleh Raja Darius di Susa. Histiaues ingin mengirim pesan rahasia kepada menantunya, Aritagoras, di Miletus[2]. Untuk itu Histiaeus mencukur habis rambut budaknya dan mematok pesan rahasia yang ingin dikirim melalui kepala budak-budaknya tersebut.

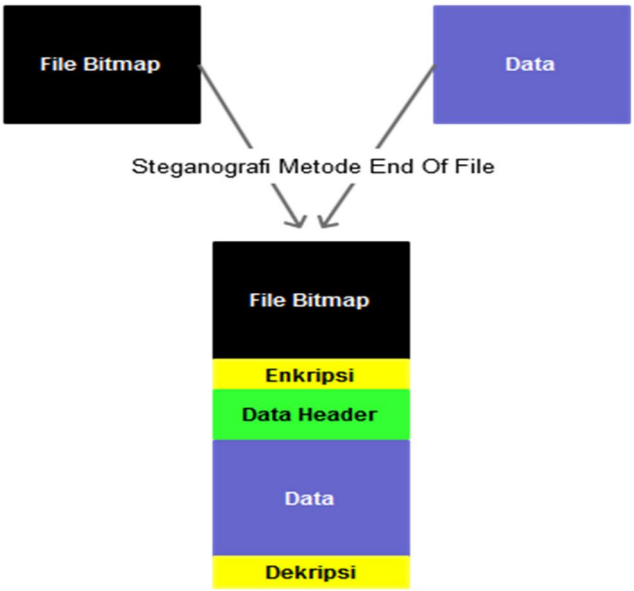
B. Media Steganografi

Teknik steganografi Dari algoritma yang digunakan hingga media apa yang digunakan. Beberapa contoh media penyisipan pesan rahasia yang digunakan yaitu [3] :

1) Steganografi pada teks
   Pada algoritma steganografi yang menggunakan teks sebagai media penyisipannya biasa digunakan teknik NLP sehingga teks yang telah disisipkan pesan rahasia akan mencurigai orang yang melihatnya.
2) Steganografi pada citra
   Pada media citra merupakan format yang sering digunakan, karena format ini merupakan salah satu format *file* yang sering dipertukarkan dalam dunia internet. Alasannya adalah banyak tersedia algoritma steganografi untuk media penampung yang berupa citra.
3) Steganografi pada audio
   Format ini pun sering dipilih karena biasanya berkas dengan format ini berukuran relatif besar. Sehingga dapat menampung pesan rahasia dalam jumlah yang besar pula.
4) Steganografi pada vidio
   Format ini memang merupakan format dengan ukuran *file* yang relatif sangat besar namun jarang digunakan karena ukurannya yang terlalu besar sehingga mengurangi kepraktisannya dan juga kurangnya algoritma yang mendukung format ini.

C. Konsep Metode *End Of File*

Dalam metode EOF data yang disisipkan pada akhir *file* diberi tanda khusus sebagai pengenal *start* dari data tersebut dan pengenalan akhir data tersebut. Sehingga, tidak akan mengganggu kualitas data awal yang akan disisipkan pesan. Namun, ukuran *file* setelah disisipkan pesan rahasia akan bertambah. Sebab, ukuran *file* yang telah disisipkan pesan rahasia sama dengan ukuran *file* sebelum disisipkan pesan rahasia ditambah dengan ukuran pesan rahasia yang disisipkan[4].



Gambar 1: Konsep Metode End Of File

2.2. Kriptografi

A. Sejarah Kriptografi

Pada dasarnya kriptografi sudah dikenal sejak lama, kriptografi sudah digunakan oleh bangsa Mesir sejak 4000 tahun yang lalu oleh raja-raja Mesir pada saat perang untuk mengirimkan pesan rahasia kepada panglima perangnya melalui kurir-kurirnya rumit [5].

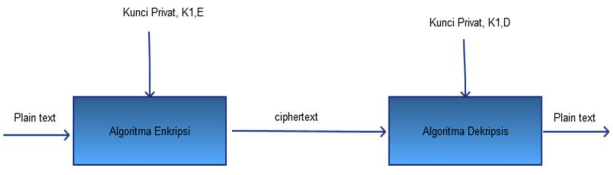
B. Pengertian Kriptografi

Kriptografi (*cryptography*) adalah ilmu atau seni yang menggunakan matematika untuk mengamankan suatu informasi. Algoritma kriptografi merupakan aturan untuk *enciphering* dan *deciphering* atau fungsi matematika yang digunakan untuk melakukan proses enkripsi dan dekripsi. Algoritma kriptografi terdiri dari tiga fungsi dasar yaitu enkripsi, dekripsi dan kunci[6].

C. Jenis Kriptografi

1) Kriptografi Kunci Simetri

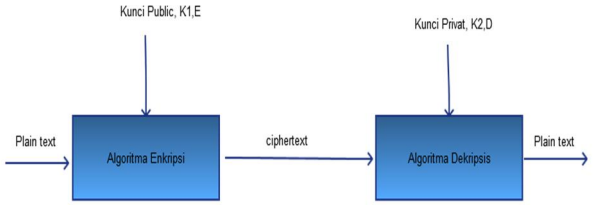
Kunci simetri berarti menggunakan kunci yang sama untuk proses enkripsi maupun dekripsi pada prosesnya pengirim pesan harus berbagi kunci rahasia tersebut. Keamanan sistem kriptografi kunci simetri terletak pada kerahasiaan kuncinya[7].



Gambar 2: Kriptografi Kunci simetri

2) Kriptografi Kunci Asimetris

Teknik kriptografi kunci asimetris berarti mengugunakan kunci *public* dan kunci *privat.* pada proses enkripsi , dekripsi dan pembuatan kunci teknik kriptografi asimetris memerlukan komputerasi yang lebih intensif dibandingkan kriptografi simetri, karena enkripsi asimetris menggunakan bilangan-bilangan yang sangat besar. Naskah yang telah dienkripsi menggunakan kunci privat hanya dapat didekripsi menggunakan kunci publik dan naskah yang dapat didekripsi menggunakan kunci publik dapat dipastikan telah dienkripsi menggunakan kunci privat. Sebaliknya, naskah yang telah dienkripsi menggunakan kunci publik hanya dapat didekripsi menggunakan kunci privat[8].



Gambar 3 : Kriptografi Kunci Asimetris

D. Algoritma Kriptografi Rijndael

Kriptografi Rijndael termasuk kedalam jenis kriptografi yang sifatnya simetri dan cipher block. Dengan demikian algoritma ini mempergunakan kunci yang sama saat enkripsi dan dekripsi serta masukan dan bit keluarannya berupa blok dengan jumlah bit tertentu[9]. Algortima Rijndael mendukung berbagai variasi ukuran kunci yang tetap sebesar 128, 192, dan 256bit. Pemilihan ukuran blok data dan kunci akan menentukan jumlah proses enkripsi dan dekripsi. Tabel berikut adalah perbandingan jumblah proses yang harus dilalui untuk masing-masing masukan [10].

Tabel 1 Jumlah Proses Berdasarkan Bit Blok dan Kunci.

Panjang Kunci (Nk) Dalam word	Ukuran blok data (Nb) Dalam Word	Jumlah proses (Nr)
4	4	10
6	4	12
8	4	14

1) Proses Pembuatan Kunci (*Roundkey*)

Proses pembuatan kunci terdiri dari empat tahap yaitu RotWord, SubWord, proses XOR dengan nilai Rcon dan proses XOR dengan word sebelumnya.

a) Tahap *RotWord*

Untuk menghasilkan kolom pertama pada sub-kunci pertama(W_i), maka yang akan dioperasikan pertama adalah kolom ke empat dari *ciphert key*. Prosesnya hanya menggeser kolom secara sekali ke atas

b) Tahap *SubWord*

Word hasil proses *RotWord* akan disubtitusi dengan nilai dalam table Sbox.

c) Tahap XOR dengan Nilai Rcon

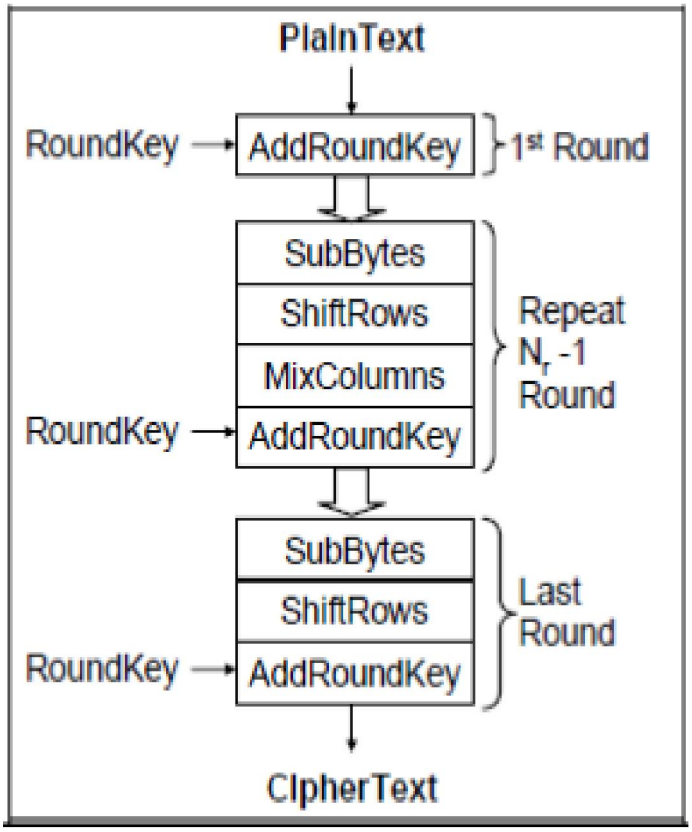
Tahap ini untuk mendapatkan kolom ke W_i adalah proses XOR yang dilakukan antara hasil *SubWord* dengan nilai Rcon yang bersesuaian, lalu proses XOR dengan kolom W_{i-4}. Prosesnya adalah dengan mengubah bilangan heksadesimal pada matriks sebelumnya menjadi bilangan biner, kemudian dilakukan proses XOR.

d) Tahap XOR dengan *Word* Sebelumnya

Untuk mendapatkan kolom kedua pada sub-kunci pertama,cukup dengan melakukan operasi XOR antara W_i - dengan kolom W_{i-3}. Cara yang sama untuk kolom ketiga dan keempat.

2) Proses Enkripsi Rijndael

Pada proses enkripsi Rijndael terdiri dari empat transformasi *byte* yaitu *SubByte, ShiftRows, Mixcolumns,* dan *AddRoundkey*. pada awal proses enkripsi, masukan yang telah berbentuk *array state* akan mengalami transformasi *addRoundKey*. kemudian *array state* akan mengalami transformasi *subByte, ShiftRows, Mixcolumns,* dan *AddRoundkey* secara berulang-ulang sebanyak Nr, *round* terakhir tidak mengalami transformsi *MixColumns*. Proses ini dalam algoritma Rijndael disebut sebagai *round function*[11].



Gambar 4: Alur Proses Enkripsi

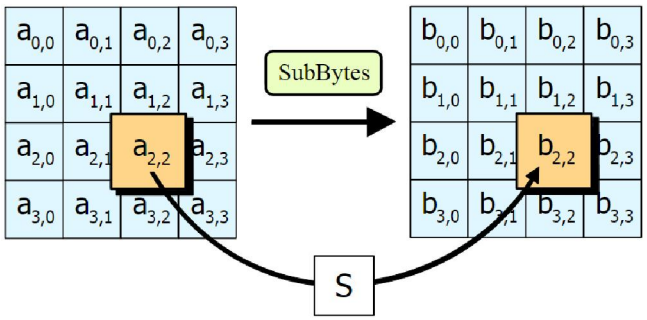
a) Proses *SubByte*

Proses *SubByte* memetakan setiap *byte* dari *array state* dengan menggunakan table subtitusi *S-box* [2]. Cara pensubstitusian adalah sebagai berikut : untuk setiap *byte* pada *array state*, misalkan S [*r*, *c*] = *xy*, yang dalam hal ini *xy* adalah

digit heksadesimal dari nilai *S*[*r*,*c*], maka nilai subtitusinya dinyatakan dengan S'[r,c], adalah elemen didalam *S-box* yang merupakan Perpotongan baris x dengan kolom y.

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

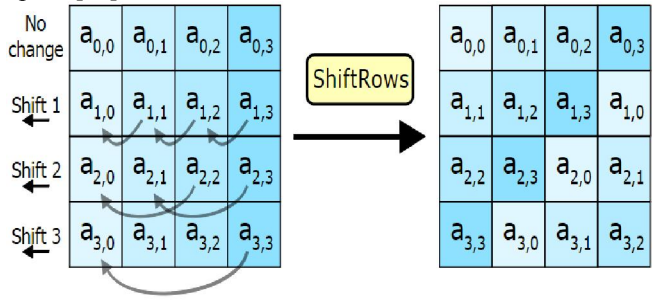
Gambar 5 : S-BOX Rijndael



Gambar 6 : Skema SubByte

b) Proses *ShiftRows*

Proses *ShiftRows* akan beroperasi pada tiap baris dengan tabel *state*. Proses ini akan bekerja dengan cara memutar *byte-byte* pada 3 baris terakhir (baris 1, 2 dan 3) dengan jumlah perputaran yang berbeda- beda. Baris 1 akan diputar sebanyak 1 kali, baris 2 akan diputar sebanyak 2 kali, dan baris 3 akan diputar sebanyak 3 kali. Sedangkan baris 0 tidak akan diputar[12].

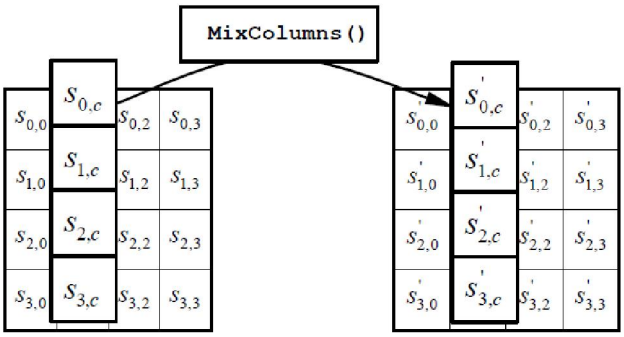


Gambar 7 : Skema ShiftRows

Proses *ShiftRows* dilakukan dengan melakukan pergeseran secara siklik pada 3 baris terakhir dari *array state*. Jumlah pergeseran bergantung pada urutan baris *r*. Baris *r* =1 digeser sejauh 1 byte, baris r = 2 digeser sejauh 2 byte, dan baris r = 3 digeser sejauh 3 byte. Baris *r*=0 tidak digeser[13].

c) Proses *Mixcolumns*

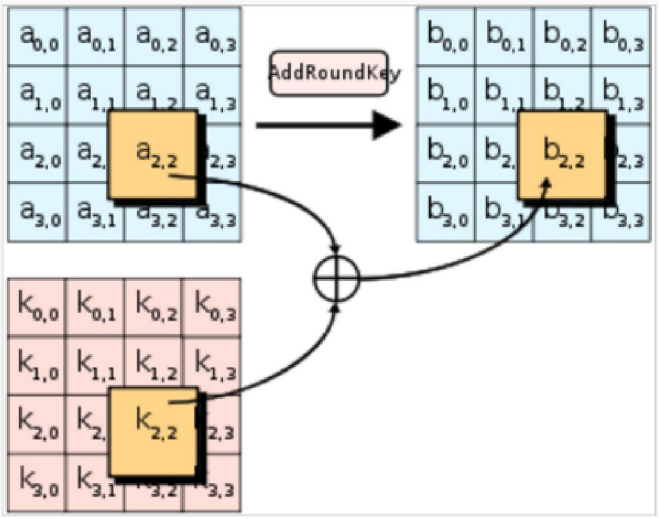
Proses *Mixcolumns* beroperasi pada *state* kolom perkolom dengan memperlakukan setiap kolom sebagai 4 buah polinomial. Kolom tersebut dianggap sebagai polinomial pada GF(28) dan dikalikan modul x4+ 1 dengan polinomial tetap a(x)[10].



Gambar 8: Skema Proses Mixcolumns

d) Trasnformasi *AddRoundkey*

Pada proses ini *subkey* digabungkan dengan *state*. Proses penggabungan ini menggunakan operasi XOR untuk setiap *byte* dari *subkey* dengan *byte* yang bersangkutan dari *state*. Untuk setiap tahap, *subkey* dibangkitkan dari kunci utama dengan menggunakan proses *key schedule*. Setiap *subkey* berukuran sama dengan state yang bersangkutan [12].



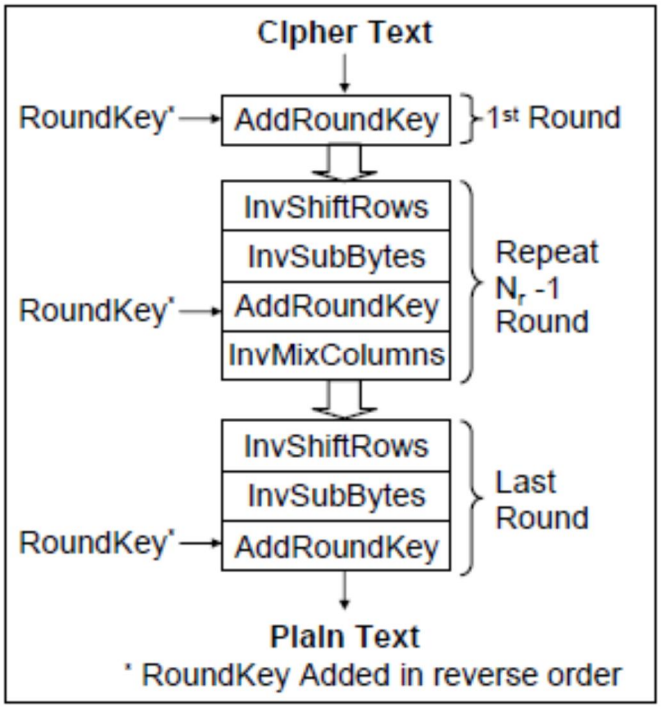
Gambar 9 : Skema Proses Proses Addroundkey

3) Proses Dekripsi Rijndael

Struktur proses dekripsi secara umum sama dengan proses enkripsi, tetapi pada proses dekripsi Rijndael memiliki urutan proses transformasi penyusun tiap iterasi yang berbeda. Tidak hanya itu transformasi yang digunakan pun merupakan transformasi kebalikan atau *invers* dari proses transformasi penyusun setiap pada proses enkripsi [4].

Transformasi *cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma Rijndael. Transformasi byte yang digunakan pada

invers *cipher* adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*[6].



Gambar 10: Alur Proses Dekripsi Rijndael

III. ANALISIS PERANCANGAN PROGRAM

3.1. Analisa Masalah

PT Kothis Prima Mitra mempunyai masalah tentang tingkat keamanan data. Data yang sering digunakan berupa *file* dokumen, dimana *file* tersebut tidak semua pihak berhak melihat dan mengaksesnya. PT. Kothis Prima Mitra hanya melakukan pengamanan data secara sederhana dan belum memperhatikan seberapa amankah tingkat keamanannya. Hal ini memungkinkan terjadinya pencurian data dan manipulasi data oleh orang yang tidak bertanggung jawab. Oleh sebab itu PT. Kothis Prima Mitra membutuhkan suatu aplikasi pengamanan data agar data yang tersimpan terjamin keamanannya.

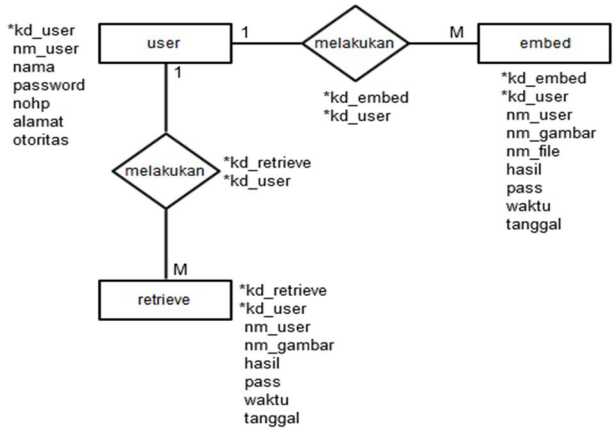
3.2. Penyelesaian Masalah

Dari permasalahan diatas didapat sebuah solusi dengan membuat sebuah aplikasi yang dapat menjaga kerahasiaan dari informasi ataupun data pada PT. Kothis Prima Mitra. Salah satu cara untuk menyembunyikan data tersebut dengan menggunakan teknik kriptografi dan steganografi. Teknik kriptografi yang digunakan adalah kriptografi algoritma Rijndael dengan melakukan enkripsi menjadi chiperteks, sehingga tidak dapat dibaca lagi. Untuk membacanya lagi perlu dilakukan proses dekripsi. Selain itu dilakukan pengamanan ganda dengan menggunakan teknik steganografi *End Of File* (EOF).

3.3. Rancangan Basis Data

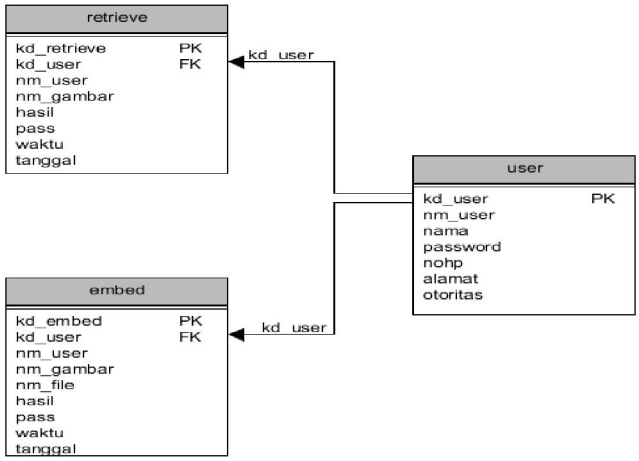
Dalam proses pembuatan aplikasi ini, dibutuhkan basis data yang berisikan semua data untuk menjalankan aplikasi sebagai berikut :

A. *Entity Relationship Diagram*



Gambar 11 : ERD (Entity Relationship Diagram)

B. *Logical Record Structure*

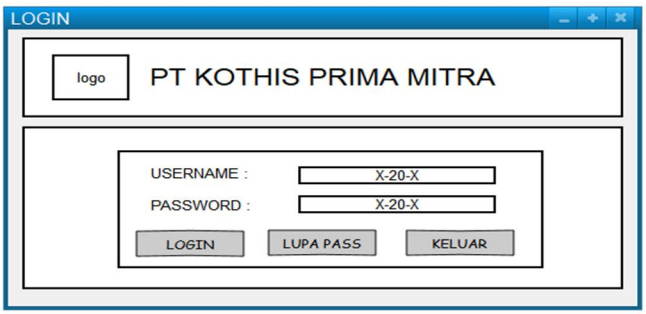


Gambar 12 : LRS (Logical Record Structure)

3.4. Rancangan Layar

A. Rancangan Layar *Form Login*

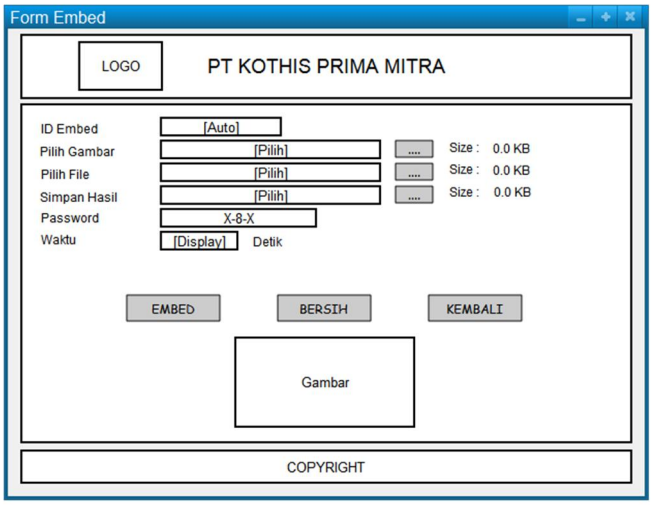
Form Login merupakan form pertama yang ditampilkan pada *user* saat menggunakan aplikasi ini.



Gambar 13 : Rancangan Layar Form Login

B. Rancangan Layar *Form Embed*

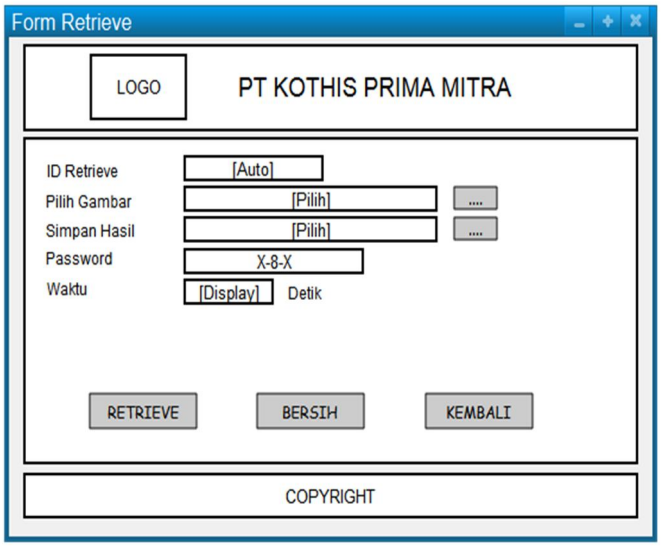
From Embed berfungsi Untuk meng-*embed* dan mengenkripsi *file, form* ini merupakan fungsi utama dari aplikasi ini.



Gambar 14 :Rancangan Layar Form Embed

C. Rancangan Layar *From Retrieve*

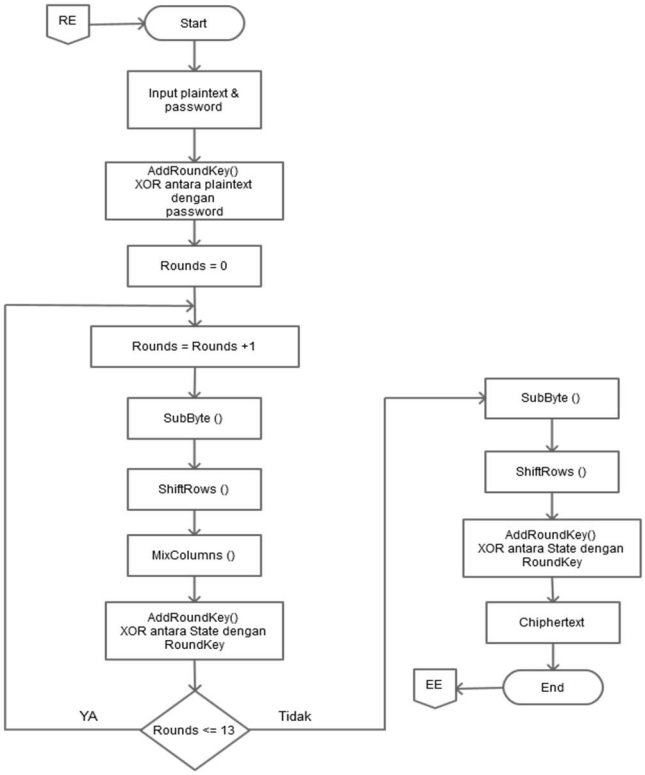
Form retrieve adalah *form* yang berfungsi untuk melakukan *retrieve* dan dekripsi *file* atau mengubah *file* dari yang tidak terbaca dan tersembunyi menjadi ke bentuk semula.



Gambar 15 : Rancangan Layar Form Retrieve

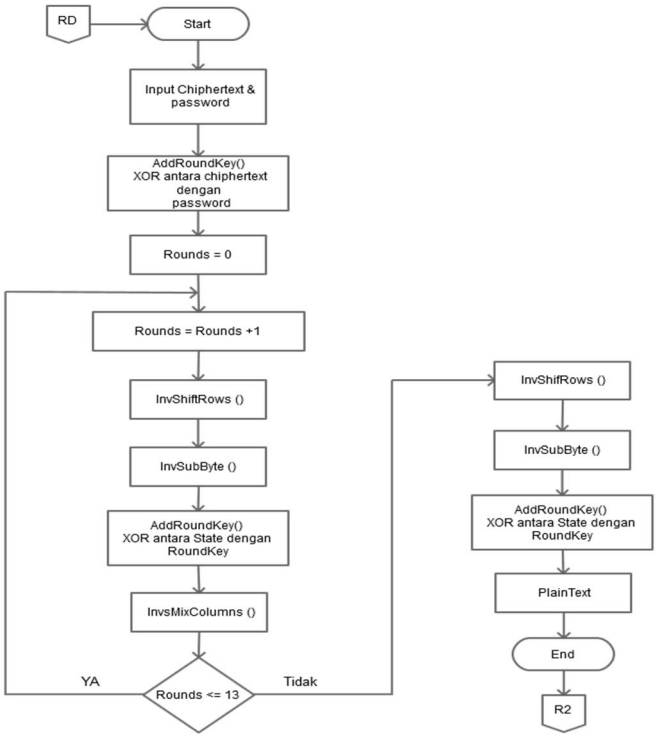
3.5. *Flowchart*

A. *Flowchart Enkripsi Rijndael*



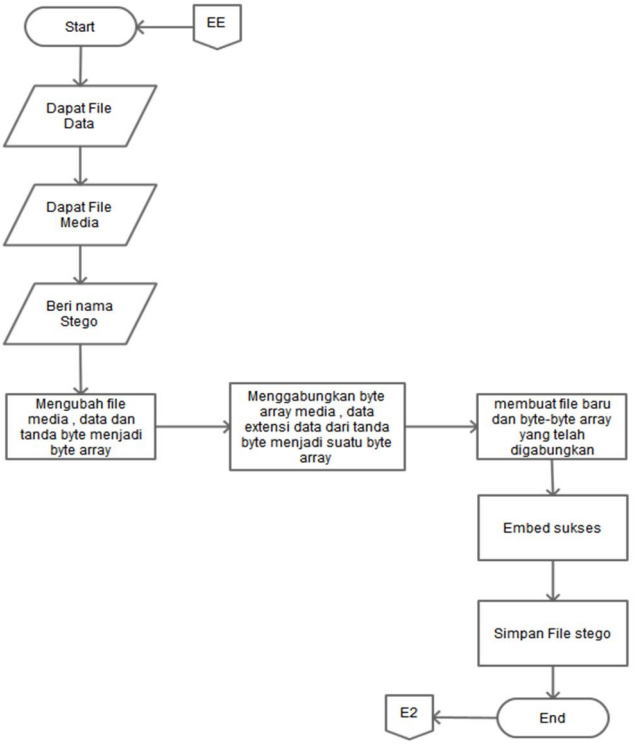
Gambar 16 : Flowchart Enkripsi Rijndael

B. *Flowchart Dekripsi Rijndael*



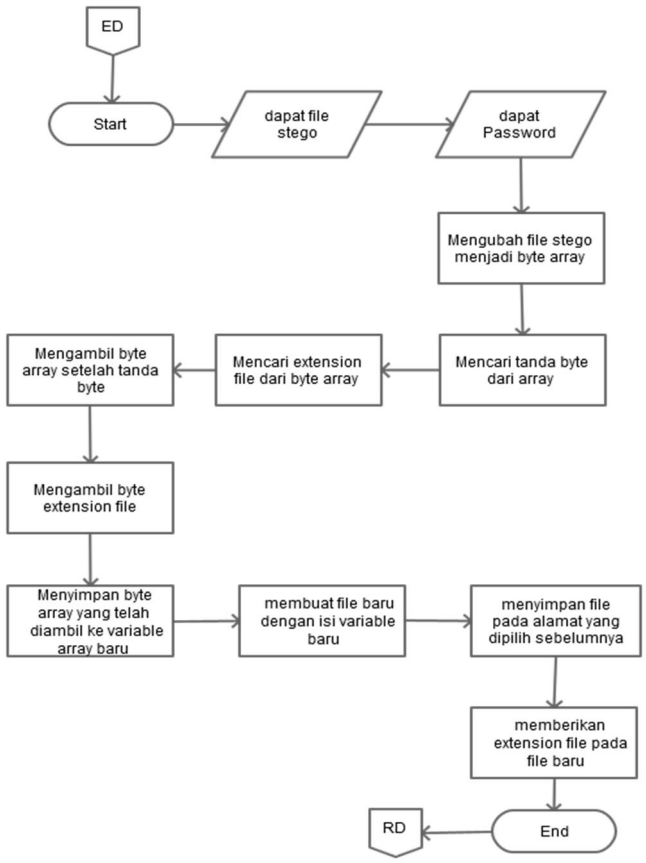
Gambar 17 : Flowchart Dekripsi Rijndael

C. *Flowchart Embed EOF*



Gambar 18 : Flowchart Embed EOF

D. *Flowchart Retrieve EOF*



Gambar 19 : Flowchart Retrieve EOF

3.6. Algoritma Program

A. Algoritma Enkripsi Rijndael

1. Proses Rijndael enkrip
2. *Start*
3. *Input plaintext & password*
4. addRoundkey = *plaintext* XOR *Password*
5. Rounds = 0
6. Rounds = rounds +1
7. Proses SubByte() : subtitusi byte dengan tabel S-Box
8. Proses ShiftRows() : pergeseran baris array
9. Proses MixColumns() : mengoprasikan setiap elemen
10. Proses AddRoundKey() : *current* state XOR
11. *if* rounds <=13 *Then*
12. Kembali kebaris 5
13. *Else*
14. Proses SubByte() : subtitusi byte dengan tabel S-Box
15. Proses ShiftRows() : pergeseran baris-baris array secara wrapping
16. Proses AddRoundKey() : *current* state XOR
17. Output *Chiphertext*
18. *End if*
19. *End*
20. Menuju ke EOF enkrip

B. Algoritma Dekripsi Rijndael

1. Proses Rijndael dekrip
2. *Start*
3. *Input chiphertext & password*
4. addRoundkey = *chiphertext* XOR *Password*
5. Rounds = 0
6. Rounds = rounds +1
7. Proses InvShiftRows() : pergeseran baris-baris array secara wrapping
8. Proses InvSubByte() : subtitusi byte dengan tabel S-
9. Proses AddRoundKey() : *current* state XOR
10. Proses invMixColumns() : mengoprasikan setiap elemen yang berada dalam satu kolom menggnakan
11. *if* rounds <=13 *Then*
12. Kembali kebaris 5
13. *Else*
14. Proses InvShiftRows() : pergeseran baris-baris array
15. Proses InvSubByte() : subtitusi byte dengan tabel S-
16. Proses AddRoundKey() : *current* state XOR
17. Output *Plaintext*
18. *End if*
19. *End*
20. Menuju ke *Form* Retrieve R2

C. Algoritma Embed Rijndael

1. Proses EOF enkrip
2. *Start*
3. dapat*file* data
4. dapat *file* media
5. Beri nama stego
6. Mengubah *file* media, data dan tanda byte menjadi byte array
7. Menggabungkan byte array media, data extensi data dari tanda btye menjadi suatu byte array
8. Membuat *file* baru dan byte-byte array yang telah digabungkan
9. *Embed* sukses
10. Simpan *file* stego
11. *End*
12. Menuju ke *form* enkrip E2

D. Algoritma Retrieve Rijndael

1. Proses EOF dekrip
2. *Start*
3. dapat file stego
4. dapat *password*
5. Mengubah file stego menjadi byte array
6. Mencari tanda byte dari array
7. Mencari extension file dari byte array
8. Mengambil byte array setelah tanda byte
9. Mengambil byte extension file
10. Meyimpan byte array yang telah diambil ke variable array baru
11. Membuat file baru dangen isi variable baru
12. Menyimpan file pada alamat yang dipilih sebelumnya
13. Memberikan extension file pada file baru
14. *End*
15. Menuju Proses Rijndael dekrip

IV. HASIL DAN PEMBAHASAN

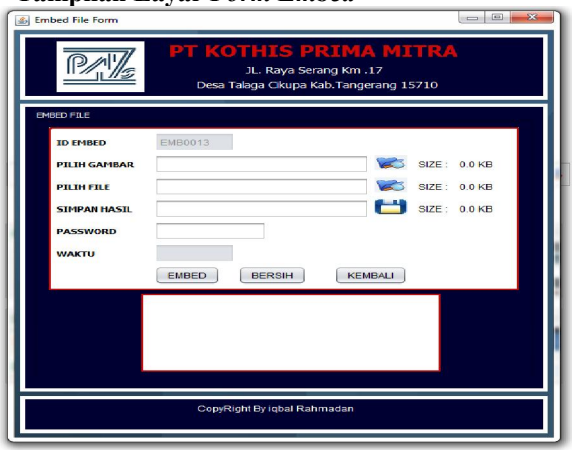
4.1. Tampilan Layar

Tampilan layar program berguna untuk mengetahui bahwa dapat berjalan secara baik atau terjadi kesalahan yang tidak diinginkan.

A. Tampilan Layar Menu Utama

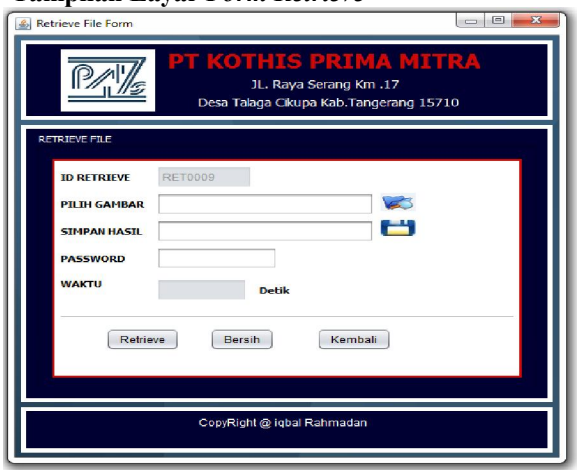
Gambar 20 : Tampilan Layar Menu Utama

B. Tampilan Layar *Form Embed*



Gambar 21 : Tampilan Layar Form Embed

C. Tampilan Layar *Form Retrieve*



Gambar 22 : Tampilan Layar Form Retrieve

4.2. Uji Coba Aplikasi

Setelah Kebutuhan terpenuhi baik *software* maupun *hardware,* maka proses selanjutnya adalah menguji coba aplikasi yang telah dibuat. Berikut ini adalah tabel pengujian aplikasi yang bertujuan untuk mengetahui apakah aplikasi telah berjalan dengan baik.

A. Pengujian *Embed File*

Tabel 2: Tabel Pengujian embed file

Nama File	Ukuran File Asli (KB)	Ukuran File Hasil Enkripsi (KB)	Nama Gambar	Ukuran Gambar Asli (KB)	Ukuran Gambar Hasil Embed (KB)	Waktu Embed (Detik)	Keterangan
Surat Keterangan.doc	30	92	250kb.jpg	251	329	3.618	berhasil
Test166kb.docx	166	410	38kb.png	38	447	3.717	berhasil
Report oktober.xls	130	348	8kb.jpg	8	354	7.94	berhasil
Data Karyawan.xlsx	39	105	PT.png	47	148	24.083	berhasil
Surat keterangan.pdf	336	369	PT720.png	56.2	421	164.87	berhasil
Spk3.pdf	1.430	-	1,5mb.jpg	1.480	-	-	Gagal program
Text1000kb	1.159	-	PT.png	47	-	-	Gagal program

B. Pengijuan *Retrieve File*

Tabel 3 :Tabel Pengujian retrieve file

Nama Gambar	Ukuran Gambar Hasil Embed (KB)	Waktu Retrieve (Detik)	Ukuran File Hasil Retrieve (KB)
Test1.jgp	329	3.246	30
Test2.png	447	31.397	166
Test3.jpg	354	20.085	130
Test4.png	148	3.219	39
Test5.png	421	22.794	369

V. PENUTUP

Berdasarkan penelitian yang telah dilakukan terhadap permasalahan dari aplikasi yang dibuat, maka dapat ditarik kesimpulan dan saran yang mungkin diperlukan untuk pengembangan aplikasi ketahap yang lebih baik lagi.

5.1 Kesimpulan

Melalui proses perancangan, pembuatan, analisa program dan serangkaian uji coba dari aplikasi ini, maka dapat diambil suatu kesimpulan antara lain :

a. Dengan dibuatnya aplikasi ini PT. Kothis Prima Mitra dapat mengamankan data-data penting seperti data pendapatan,data transaksi, dan data karyawan.
b. Dengan menggunakan metode kriptografi algoritma Rijndael dan metode steganografi *End Of File* (EOF) mampu mengurangi tingkat pencurian data rahasia perusahaan sehingga sulit dilakukan oleh pihak yang tidak memiliki wewenang.
c. Kecepatan Proses *embed* dan *retrieve* pada aplikasi ini sangat bergantung dengan ukuran *file,* semakin besar ukuran *file* maka waktu yang digunakan semakin lama, sedangkan semakin kecil ukuran *file* maka waktu yang digunakan semakin cepat.

5.2 Saran

Adapun saran yang mungkin diperlukan untuk membuat aplikasi ini dapat berjalan lebih baik lagi antara lain :

a. Ukuran *file* rahasia sebaiknya bisa lebih besar dari 1 Mb.
b. Proses *encode* dan *decode* diharapkan dapat berjalan dengan cepat bila ukuran *file* melebihi 1 Mb.
c. Algoritma enkripsi yang dibuat sebaiknya selalu ditingkatkan, karena dengan semakin berkembangnya ilmu pengetahuan kriptografi, maka tidak dapat dipastikan apakah algoritma ini masih bisa diandalkan.
d. Aplikasi ini hanya dapat mengamankan *file* docx, .doc, .xlsx, .xls, dan .pdf.
e. Diharapkan dapat mengamankan lebih banyak *file* dokumen ataupun video.
f. Format *file* penampung sebaiknya ditambahkan dengan format seperti *.gif, *.bmp.
g. Diharapkan aplikasi ini dapat terintegrasi dengan internet, sehingga pengguna dapat mengakses aplikasi ini dimana saja.

DAFTAR PUSTAKA

[1] Anggraini, Y. & Shaka, D., 2014. Penerapan Steganografi Metode End of File (EOF) dan Enkripsi Metode Data Encryption Standard (DES) pada Aplikasi Pengamanan Data Gambar. *konferensi Nasional Sistem Informasi*. Makassar, pp. 1743–1753.

[2] Sembiring, S., 2013. Perancangan Aplikasi Steganografi untuk Menyisipkan Pesan pada Gambar dengan Metode end of file. *Jurnal Pelita Informatika Budi Darma*, 4(2), pp.45–51.

[3] Saefullah, A. & Agani, N., 2012. Aplikasi Steganografi untuk Menyembunyikan Teks Dalam Media Image Dengan Menggunakan Metode LSB. *Seminar Nasional Teknologi Informasi & Komunikasi Terapan*. Semarang, pp. 151–157.

[4] Wandani, H., 2012. Tugas Akhir : Implementasi Sistem Keamanan Data Dengan Menggunakan Teknik Steganografi End Of File (EOF) dan Rabin Public Key Cryptosystem. *Universitas Sumatera Utara*

[5] Meidina, 2013. Tugas akhir :Visualisasi Algoritma RSA dengan Menggunakan bahasa Pemrograman Java. *Universitas gunadarma*.

[6] Hanifah, F., 2012. Tugas akhir : Aplikasi Algoritma Rijndael dalam Pengamanan citra digital. *Univeritas Indonesia*.

[7] Ratih, 2010. Tugas Akhir : Studi dan Perbandingan Penggunaan Kriptografi Kunci Simetri dan Asimetri pada Telepon Selular, *Institute Teknologi Bandung*.

[8] Kromodimoeljo, S., 2009. *Teori & Aplikasi Kriptografi*, Jakarta: SPK IT Consulting.

[9] Indriyono, B.V., 2016a. Implementasi Sistem Keamanan File dengan Metode Steganografi EOF dan Enkripsi Caesar Cipher. *jurnal Sistem informasi*, 06(01), pp.1–16.

[10] Setiawan, P.A., 2015. Tugas Akhir : Aplikasi Keamanan Data Menggunakan Algoritma Kriptografi AES 128 dan Steganografi LSB. *Universitas Budiluhur*.

[11] Bendi, R.K. jawa & Bp, S.A., 2016. Implementasi Algoritma Rijndael untuk Enkripsi dan Deskripsi Pada Citra Digital. *Seminar Nasional Informatika*. yogyakarta.

[12] Maulana, I.R., 2014. Tugas akhir : Apikasi Pengamanan Data Pada Ponsel Android Dengan Menggunakan Algoritma Rijndael dan LZW. *Universitas Budiluhur*.

[13] Fadli, hadyan ghaziani, 2010. Tugas Akhir : Studi dan Implementasi Algoritma Rijndael Untuk Enkripsi Halaman Web HTML. *Institut Teknologi Bandung*.