

Pengamanan Citra Digital Menggunakan Kombinasi Antara Algoritma Aes Dan Metode Lsb

Muhammad Zunaidi, Suharsil

STMIK Triguna Dharma

info@trigunadharma.ac.id

Abstrak

Pesatnya perkembangan teknologi sekarang ini membuat proses komunikasi menjadi lebih mudah dan memiliki jangkauan yang sangat luas untuk mengirimkan suatu pesan atau informasi. Penyampaian pesan melalui media internet merupakan sarana komunikasi yang mudah dan efisien. Akan tetapi, banyak orang yang kini telah meragukan keamanan pada sarana komunikasi yang ada. Hal ini tidak terlepas dari terjadinya berbagai tindakan penyadapan dan pemantauan oleh pihak-pihak yang tidak berkepentingan atau tidak bertanggung jawab sehingga kerahasiaannya kurang terjaga dalam mengamankan suatu pesan. Banyak cara untuk mengamankan data salah satunya dengan menggunakan teknik Kriptografi dan Steganografi. Kriptografi, secara umum adalah ilmu dan seni untuk menjaga kerahasiaan pesan. Sedangkan Steganografi adalah seni untuk menyembunyikan pesan di dalam media citra digital sedemikian rupa sehingga orang lain tidak menyadari ada sesuatu pesan di dalam media tersebut. Adapun salah satu metode yang dapat digunakan pada teknik Kriptografi adalah algoritma AES sedangkan pada teknik Steganografi dengan menggunakan metode LSB. Dengan adanya aplikasi sistem pengamanan pesan rahasia menggunakan teknik kriptografi dengan algoritma AES dan steganografi dengan metode LSB, diharapkan dapat mengamankan suatu pesan sehingga tidak dapat dimengerti oleh pihak-pihak yang tidak berkepentingan.

Kata kunci : Kriptografi, Steganografi, Algoritma AES, Metode LSB, Citra Digital, pesan

1. Pendahuluan

Pesatnya perkembangan teknologi sekarang ini membuat proses komunikasi menjadi lebih mudah dan memiliki jangkauan yang sangat luas untuk mengirimkan suatu pesan atau informasi. Penyampaian pesan melalui media *internet* merupakan sarana komunikasi yang mudah dan efisien. Akan tetapi, banyak orang yang kini telah meragukan keamanan pada sarana komunikasi yang ada. Hal ini tidak terlepas dari terjadinya berbagai tindakan penyadapan dan pemantauan oleh pihak-pihak yang tidak berkepentingan atau tidak bertanggung jawab sehingga kerahasiaannya kurang terjaga dalam mengamankan suatu pesan.

Banyak cara untuk mengamankan data salah satunya dengan menggunakan metode Kriptografi dan Steganografi. Kriptografi, secara umum adalah ilmu dan seni untuk menjaga kerahasiaan pesan. Sedangkan Steganografi adalah seni untuk menyembunyikan pesan di dalam media digital sedemikian rupa sehingga orang lain tidak menyadari ada sesuatu pesan di dalam media tersebut.

Dalam Kriptografi dan Steganografi maka data yang dianggap rahasia akan disamarkan dengan sedemikian rupa sehingga jika data itu bisa didapatkan maka tidak akan bisa dimengerti oleh pihak yang tidak berhak. Adapun salah satu metode yang dapat digunakan pada teknik Kriptografi adalah algoritma *Advanced Encryption Standard* sedangkan pada teknik Steganografi dengan menggunakan metode *Least Significant Bit*.

AES merupakan sistem penyandian blok yang bersifat non-Feistel karena AES menggunakan komponen yang selalu memiliki invers dengan panjang blok 128 bit. Kunci AES dapat memiliki panjang kunci bit 128, 192 dan 256 bit. Penyandian AES menggunakan proses yang berulang yang disebut dengan ronde. Jumlah ronde yang digunakan oleh AES tergantung dengan panjang kunci yang digunakan. Setiap ronde membutuhkan kunci dan masukan dari ronde berikutnya. Kunci ronde dibangkitkan berdasarkan kunci yang diberikan. Sedangkan LSB merupakan teknik substitusi pada steganografi, biasanya arsip 24 bit atau 8 bit digunakan untuk menyimpan citra digital. Representasi warna dari *pixel-pixel* bisa diperoleh dari warna-warna primer, yaitu merah, hijau, dan biru. Citra 24 bit menggunakan 3 *byte* untuk masing-masing *pixel*, dimana setiap warna primer direpresentasikan dengan ukuran 1 *byte*. Penggunaan citra 24 bit memungkinkan setiap pixel dipresentasikan dengan nilai warna sebanyak 16.777.216 macam. Dua bit dari saluran warna tersebut bisa digunakan untuk menyembunyikan data, yang akan mengubah jenis warna *pixel* nya menjadi 64 warna.

Dengan kombinasi Kriptografi dan Steganografi secara bersamaan dimaksudkan untuk memberikan keamanan berlapis dalam pengamanan suatu pesan. Berdasarkan uraian diatas maka pembahasan ini mengangkat judul "Pengamanan Citra Digital Menggunakan Kombinasi Antara Algoritma AES Dan Metode LSB".

2. Metode Penelitian

Dalam melakukan penulisan skripsi ini ada beberapa hal untuk mendapatkan atau mengumpulkan data yang diperlukan, antara lain adalah:

1. Pengumpulan Data yaitu :
 - a. *Surfing (Field Research)*
Yaitu pencarian atau pengumpulan data yang diperlukan melalui media internet serta sumber lainnya.
 - b. *Studi Pustaka (Library Research)*
Yaitu dengan membaca literatur-literatur dan referensi yang berhubungan dengan permasalahan yang dibahas.
2. Analisa Data.
Yaitu melakukan analisa terhadap data yang diperoleh untuk digunakan dalam menyelesaikan masalah yang diangkat.
3. Perancangan
Yaitu melakukan pengkodean terhadap rancangan-rancangan yang telah didefinisikan.
4. Implementasi
Yaitu mengevaluasi kemampuan program dengan menggunakan aplikasi yang telah di instalasi sebagai pengujian, untuk menemukan hasil enkripsi, dekripsi, penyisipan pesan dan pengestrakan.

3. ANALISA DAN HASIL

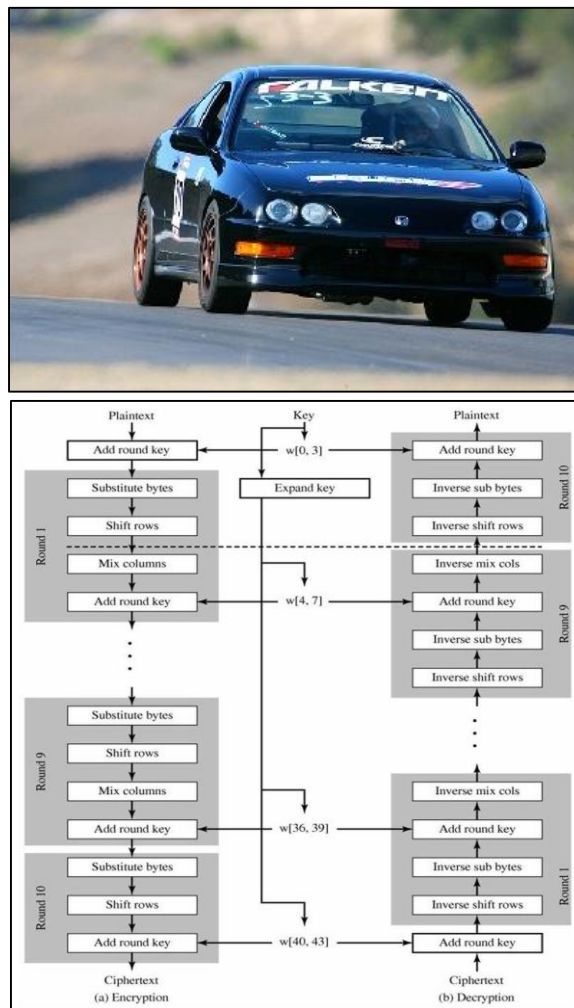
A. Algoritma Enkripsi Aes-128

Sebelum pesan rahasia disisipkan kedalam media citra digital, pesan akan dilakukan proses enkripsi menggunakan algoritma AES. Proses enkripsi ini membutuhkan kunci yang sama panjangnya dengan teks pesan, apabila panjang kunci yang dimasukkan kurang dari panjang teks pesan maka kunci akan dilakukan proses perulangan sepanjang jumlah teks pesan. Proses enkripsi ini akan menghasilkan *ciphertext* yang siap untuk disisipkan kedalam media citra digital. Sesuai dengan batasan masalah, file citra yang digunakan adalah berformat bitmap 24 bit. Pesan rahasia yang dapat ditampung di dalam

Pengamanan Citra Digital Menggunakan Kombinasi Antara Algoritma AES Dan Metode LSB (M Zunaidi)

media citra digital tergantung kapasitas citra yang dimiliki, misalkan pada berkas citra digital 24 bit berukuran $400 \times 267 \text{ pixel}$ terdapat 40050 pixel , setiap pixel (titik) pada citra tersebut terdiri dari susunan tiga warna dasar yaitu, merah, hijau dan biru (RGB), dimana setiap warna dasar dipresentasikan dengan ukuran 8 bit (1 *byte*). Sehingga jumlah seluruh kapasitas media penampung adalah $40050 \times 3 = 120150 \text{ byte}$. Karena setiap *byte* hanya bisa menyisipkan 1 bit LSB-nya, maka ukuran data yang disisipkan di dalam citra tersebut maksimum $40050 : 8 = 5006,25 \text{ byte}$ (1 *byte* = 1 karakter). Ukuran data ini harus dikurangi dengan panjang nama *file*-nya karena penyembunyian pesan rahasia tidak hanya menyembunyikan isi pesan tersebut, tetapi juga nama *file*-nya.

Misalkan pesan rahasia yang akan disisipkan adalah "MuhammadAjiPraye" dengan menggunakan kunci "A5r1S4m5UARrT1l1". Pesan tersebut akan disisipkan kedalam media citra digital "mobil.bmp" dengan ukuran $400 \times 267 \text{ pixel}$.



Dalam hal ini sebelum proses penyisipan pesan dilakukan, terlebih dahulu pesan dienkripsikan dengan algoritma AES. Berikut skema algoritma AES ditunjukkan pada gambar diatas. Dalam algoritma AES untuk mengenkripsi sebuah teks, teks terlebih dahulu diorganisir sebagai *state*. *State* adalah

blok-blok *array* dengan berukuran tetap yaitu 128 bit dimana tiap blok menyimpan bilangan biner 8 bit yang dikonversi dalam bentuk heksadesimal. Berikut ini akan dijelaskan tahapan-tahapan dalam proses enkripsi algoritma AES dengan sebuah *plaintext* dan kunci :

***Plaintext* (P) = MuhammadAjiPraye**
Kunci (K) = A5r1S4m5UARrT1I1

Langkah pertama yang dilakukan adalah mencari bilangan heksadesimal dari P dan K dengan mengkonversi P dan K dalam bentuk bilangan biner. Berikut adalah bilangan biner 8 bit dari P dan K :

P = 01001101 01110101 01101000 01100001 01101101 01101101 01100001 01100100 01000001
 01000001 01101010 01101001 01110010 01100001 01111001 01100101
 K = 01000001 00110101 01110010 00110001 01010011 00110100 01101101 00110101 01010101
 01000001 01010010 01110010 01010100 00110001 01001001 00110001

Pada AES *state* dibentuk menjadi blok *array* 4x4 (matriks 4x4) yang berarti terdapat 16 blok *array*. Berdasarkan bilangan biner yang didapat dari P dan K maka pembentukan *state* dapat dilakukan dengan membagi bilangan biner menjadi 16 bagian. Pembagian tersebut dikarenakan untuk memudahkan langkah selanjutnya yaitu mencari bilangan heksadesimal dari masing-masing bilangan biner hasil pembagian, bilangan heksadesimal untuk P dan K adalah bilangan heksadesimal dua dimensi. Setelah bilangan heksadesimal didapat maka selanjutnya penyusunan *state* blok *array* dilakukan secara vertikal. Untuk bilangan heksadesimal dari P dan K dapat dilihat pada gambar berikut:

4D	75	68	61	⊕	41	35	72	31
6D	6D	61	64		53	34	6D	35
41	6A	69	50		55	41	52	72
72	61	79	65		54	31	49	31
P					K			

Proses enkripsi algoritma AES menggunakan 4 jenis transformasi, yaitu *SubBytes*, *ShiftRows*, *MixColumns* dan *AddRoundKey*. Berdasarkan *plaintext* pada *state* akan dijelaskan untuk setiap transformasi. Berikut transformasi lagoritma enkripsi AES.

1. Transformasi *SubBytes*

Pada proses transformasi *SubBytes*, *plaintext* yang telah disusun berbentuk matriks 4x4 pada *state* melakukan pencarian bilangan heksadesimal pada *S-Box SubBytes*. Namun sebelum melakukan transformasi *SubBytes*, *plaintext* pada *state* terlebih dahulu melakukan operasi XOR dengan kunci sebagai proses pencampuran *plaintext* dan kunci (*initial Round*). Berikut adalah gambar untuk proses operasi XOR dengan plainteks pada *state* dan kunci sebagai nilai masukan.

State					Key					Output				
4D	75	68	61		41	35	72	31		0C	40	1A	50	
6D	6D	61	64		53	34	6D	35		3E	59	0C	51	
41	6A	69	50		55	41	52	72		14	2B	3B	22	
72	61	79	65		54	31	49	31		16	50	30	54	

Pada gambar diatas, *output* adalah hasil dari operasi XOR yang kemudian dianggap sebagai *state* yang baru setelah pencampuran *plaintext* dengan kunci. Proses tersebut dikatakan *initial round*.

Setelah proses XOR *plainteks* dan key, *state* keluaran kemudian di transformasikan dengan *S-Box SubBytes*. Pada transformasi ini nilai pertama bilangan heksadesimal *state* mengarah pada kolom *S-Box SubBytes*. Bilangan heksadesimal kedua *state* mengarah pada kolom *S-Box SubBytes*. Bilangan heksadesimal yang didapat antara baris dan kolom adalah hasil dari transformasi. Setelah proses transformasi, keluaran proses transformasi selanjutnya, yaitu transformasi *ShiftRows*. Untuk transformasi *SubBytes* pada *state* terhadap *S-Box* dapat dilihat pada gambar berikut:

0C	=	FE
40	1A	50
59	0C	51
2B	3B	22
50	30	54

→ S-Box

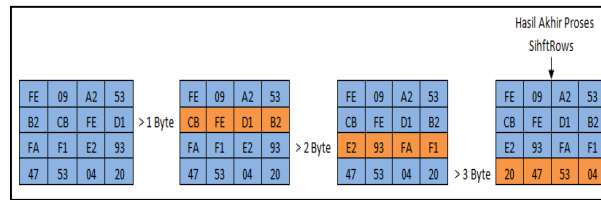
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	BB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	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	C6	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	08	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

Dari gambar di atas proses transformasi *SubBytes* maka hasil keseluruhannya ditunjukkan pada gambar berikut:

FE	09	A2	53
B2	CB	FE	D1
FA	F1	E2	93
47	53	04	20

2. Transformasi ShiftRows

Setelah melakukan transformasi *SubBytes*, selanjutnya adalah transformasi permutasi atau dikenal dengan transformasi *ShiftRows*. *ShiftRows* dilakukan dengan menjalankan operasi *circular left*. Transformasi ini mengubah posisi blok pada *state* tanpa merubah nilai blok. Perubahan posisi dilakukan sebanyak satu kali pada baris kedua, sebanyak dua kali pada baris ketiga dan sebanyak tiga kali pada baris keempat. Berdasarkan hasil transformasi *SubBytes*, maka gambar berikut adalah dari transformasi *ShiftRows*.



3. Transformasi MixColumns

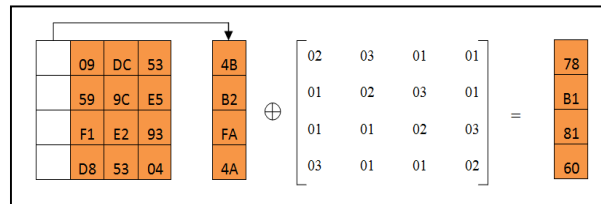
Transformasi *MixColumns* adalah proses ketiga dalam satu ronde enkripsi. Transformasi *MixColumns* bertujuan untuk mencampur blok pada kolom-kolom *state* terhadap blok keluaran. Agar dapat melakukan perhitungan matriks dengan melakukan perkalian dan penjumlahan. Disini, blok-blok pada kolom *state* akan diperlukan sebagai suatu *polynomial* yang berada dalam $GF(2^8)$ dan akan dikalikan dengan modulo $x^4 + 1$. Berikut adalah rumus perhitungan transformasi *MixColumns*.

$$S'_{0,0} = (S_{0,0} * 02) \oplus (S_{1,0} * 03) \oplus (S_{2,0} * 01) \oplus (S_{3,0} * 01)$$

$$S'_{0,1} = (S_{0,1} * 01) \oplus (S_{1,1} * 02) \oplus (S_{2,1} * 03) \oplus (S_{3,1} * 01)$$

$$S'_{0,2} = (S_{0,2} * 01) \oplus (S_{1,2} * 01) \oplus (S_{2,2} * 02) \oplus (S_{3,2} * 03)$$

$$S'_{0,3} = (S_{0,3} * 03) \oplus (S_{1,3} * 01) \oplus (S_{2,3} * 01) \oplus (S_{3,3} * 02)$$

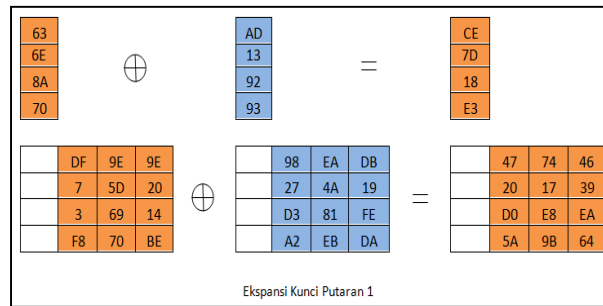


Dari gambar diatas skema rumus transformasi *MixColumns* maka hasil keseluruhannya ditunjukkan pada gambar berikut:

63	DF	9E	9E
6E	07	5D	20
8A	03	69	14
70	F8	70	BE

4. Transformasi AddRoundKey

Transformasi *AddRoundKey* pada satu ronde enkripsi adalah proses pencampuran *state* keluaran pada proses transformasi *MixColumns* dengan kunci. Dalam ronde pertama dan seterusnya yang menjadi kunci adalah hasil dari proses ekspansi kunci awal (*Key Schedule*). Berikut adalah gambar yang menunjukkan proses *AddRoundKey* antara keluaran *MixColumns* dengan hasil kunci ekspansi.



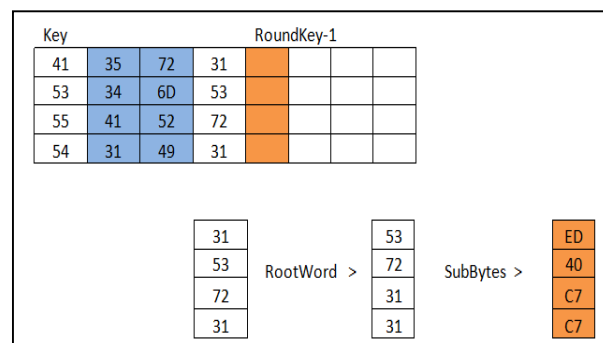
Berikut hasil transformasi *AddRoundKey* :

CE	47	74	46
7D	20	17	39
18	D0	E8	EA
E3	5A	9B	64

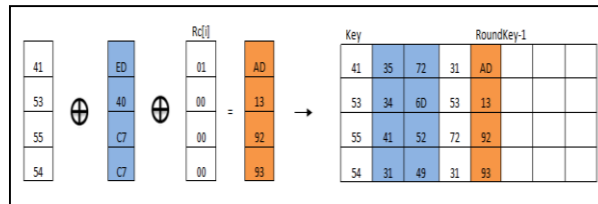
Pada gambar diatas hasil transformasi *AddRoundKey* ronde-1 yang juga merupakan *ciphertext* yang dihasilkan untuk ronde-1 (*state* keluaran ronde-1). Untuk selanjutnya *state* keluaran ronde-1 akan menjadi *state* masukan untuk ronde-2, proses tersebut akan berlangsung hingga *state* keluaran ronde-9 menjadi *state* masukan untuk ronde ke-10 dan *state* keluaran ronde-10 merupakan *ciphertext* yang dihasilkan. Untuk menghasilkan *ciphertext* dalam penyandian algoritma AES membutuhkan kunci ronde untuk setiap ronde transformasi. Kunci ronde ini dibangkitkan (diekspansi) dari kunci algoritma AES yang disebut dengan *Key Schedule*. Berikut adalah dimana proses *Key Schedule* berlangsung.

5. Key Schedule

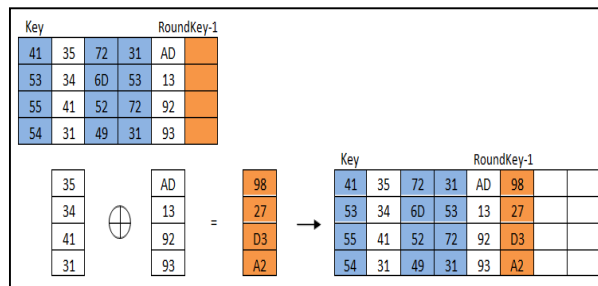
Dalam ekspansi atau pembangkitan kunci dikenal dengan istilah *Key Schedule*. Pada setiap ronde kunci yang digunakan berbeda satu dengan yang lainnya. Kunci ronde pertama adalah hasil dari ekspansi dari kunci awal, kunci ronde kedua adalah hasil dari ekspansi dari kunci ronde pertama dan seterusnya. Dalam hal ini, pembahasan *Key Schedule* akan memperlihatkan bagaimana kunci dibangkitkan pada setiap ronde hingga ronde terakhir (sesuai dengan tipe AES). Berikut ini adalah gambar dari ekspansi kunci.



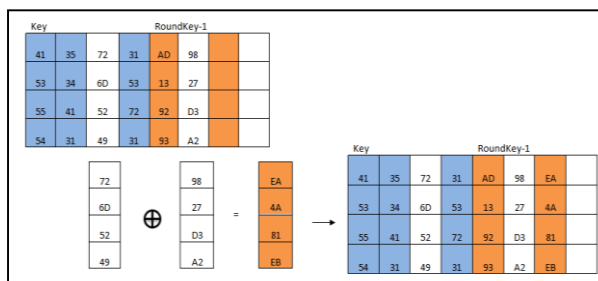
Pada gambar diatas langkah yang pertama untuk mencari kunci pada kolom pertama *round key-1* adalah dengan mencari nilai kunci kolom keempat setelah *RootWord*. *RootWord* merupakan proses pergeseran posisi nilai kunci teratas bergeser kebawah. Setelah pergeseran, maka melakukan transformasi *SubBytes*. Untuk langkah kedua dapat dilihat pada gambar berikut:



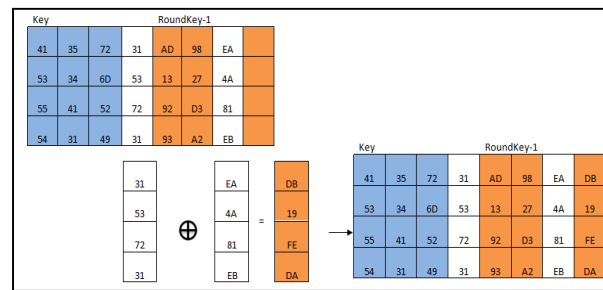
Pada langkah kedua ekspansi kunci, yaitu melakukan perhitungan XOR antara kolom pertama *key*, kolom ketiga *key* yang terlebih dahulu melalui proses *RootWord* dan *SubBytes* dengan *Rc[i]*. langkah ketiga dapat dilihat pada gambar berikut:



Pada proses langkah ketiga ekspansi kunci, maka melakukan perhitungan XOR antara kolom kedua *key* dengan kolom pertama *Round Key-1* tanpa harus ada proses *RootWord*, *SubWord* atau perhitungan XOR dengan *Rc[i]*. untuk langkah selanjutnya dapat dilihat pada gambar berikut:



Proses pada langkah keempat sama dengan proses langkah ketiga, namun pada langkah keempat kolom yang dihitung adalah kolom ketiga pada kunci dan kolom kedua pada *Round Key-1*. Untuk langkah kelima dapat dilihat pada gambar berikut:



Berikut ini gambar yang menampilkan hasil enkripsinya:

RoundKey-1			
AD	99	EA	DB
13	27	4A	19
92	D3	81	FE
93	A2	EB	DA

Proses pada langkah kelima juga sama dengan langkah ketiga dan keempat, namun pada langkah keempat kolom yang dihitung adalah kolom keempat pada *key* dengan kolom ketiga *RoundKey-1*.

Untuk mencari nilai kunci pada *RoundKey-2*, lakukan langkah yang sama, namun perhitungannya dimulai dari *Round Key-1*, dengan langkah pertama yaitu melakukan proses *RootWord* dan *SubBytes* pada kolom ketiga *Round Key-1*, setelah hasil didapat maka lakukan perhitungan XOR dengan kolom pertama *Round Key-1* dan *Rc[i]*. untuk *Rc[i]* dapat dilihat pada tabel 2.6 pada bab II. Berikut ini hasil ekspansi kunci putaran 10.

key	RoundKey-1				RoundKey-2				RoundKey-3				RoundKey-10			
41	35	72	31	AD	98	EA	DB	7B	E3	09	D2	DC	3F	36	E4	C1
53	34	6D	53	13	27	4A	19	05	22	68	71	FC	DE	B6	C7	8D
55	41	52	72	92	D3	81	FE	C5	16	97	69	93	85	12	7B	51
54	31	49	31	93	A2	EB	DA	2A	88	63	B9	9F	17	74	CD	68

Untuk melihat gambaran enkripsi Ronde 1-5 dapat dilihat pada tampilan berikut:

	Round 1				Round 2				Round 3				Round 4				Round 5			
After SubBytes	FE	9	A2	53	8B	AD	92	5A	50	3F	7D	D6	6A	FE	FC	93	F0	2E	69	FE
	B2	CB	FE	D1	FF	87	F0	12	AB	2D	64	AD	1A	8A	E1	51	74	E5	19	08
	FA	F1	E2	93	AD	70	9B	87	82	D0	73	8E	45	45	9E	92	79	9D	21	96
	47	53	4	20	11	BE	46	43	6F	F7	47	20	A3	AA	A5	CB	3E	03	3E	41
After ShiftRows	FE	09	A2	53	8B	AD	92	5A	50	3F	7D	D6	6A	FE	FC	93	F0	2E	69	FE
	B2	CB	FE	D1	FF	87	F0	12	AB	2D	64	AD	1A	8A	E1	51	74	E5	19	08
	E2	93	FA	F1	9B	87	AD	70	73	BE	82	D0	9E	92	45	45	21	96	79	9D
	20	47	53	04	43	11	BE	46	20	6F	F7	47	CB	A3	AA	A5	41	3E	03	3E
After MixColumns	63	DF	9E	9E	17	C6	1A	98	84	33	63	C6	04	EE	FF	F3	AF	DF	B0	D8
	6E	07	5D	20	0B	D8	E4	69	BF	11	56	87	17	29	3B	CD	03	83	F1	94
	8A	03	69	14	D4	76	18	8F	F8	ED	CD	0F	81	DE	C2	F7	94	42	96	E9
	70	F8	70	BE	2C	AE	75	ED	EE	75	5D	94	27	37	44	A0	4D	81	CC	8C
RoundKey	AD	98	EA	D8	7B	E3	09	D2	DC	3F	36	E4	13	2D	1B	FF	43	6E	75	8A
	13	27	4A	19	05	22	68	71	FC	DE	B6	C7	DD	03	B5	72	F8	95	20	52
	92	D3	81	FE	C5	16	97	69	93	85	12	7B	2E	AB	B9	C2	44	EF	56	94
	93	A2	EB	DA	2A	88	63	89	9F	17	74	CD	F6	E1	95	58	83	62	7F	AF
After RoundKey	CE	47	74	46	6C	25	13	4A	58	DC	55	22	17	C3	E4	0C	EC	B1	C5	52
	7D	20	17	39	0E	FA	8C	18	43	CF	E0	70	CA	2A	8E	BF	FB	16	D1	C6
	18	D0	E9	CA	11	60	9F	66	68	68	DF	74	4F	75	7B	35	D0	AD	D0	7D
	E3	5A	9B	66	06	26	16	54	71	62	29	58	D1	D5	D1	F8	CE	E3	8B	23

Sedangkan berikut ini adalah gambaran enkripsi ronde 6-10.

	Round 6				Round 7				Round 8				Round 9				Round 10			
After SubBytes	CE	C8	A6	00	C8	E7	A7	22	BF	68	A1	06	74	C6	01	E1	72	83	F1	89
	0F	47	3E	B4	F9	02	E2	4D	FF	3D	5F	DF	83	1E	25	B3	61	A7	C5	0E
	70	95	BA	FF	C8	A0	6E	7D	61	7F	A7	53	7D	2C	BC	7A	20	3D	F7	7F
	8B	11	E2	26	6B	CA	D2	13	FE	60	21	1C	E1	93	38	10	1B	8B	A5	7E
After ShiftRows	CE	C8	A6	00	C8	E7	A7	22	BF	68	A1	06	74	C6	01	E1	72	83	F1	89
	47	3E	B4	0F	02	E2	4D	F9	3D	5F	DF	FF	1E	25	B3	83	A7	C5	0E	61
	BA	FF	70	95	6E	7D	C8	A0	A7	53	61	7F	BC	7A	7D	2C	F7	7F	20	3D
	26	8B	11	E2	13	6B	CA	D2	1C	FF	60	21	10	E1	93	38	7E	1B	8B	A5
After MixColumns	D2	BD	F1	56	F0	FE	80	26	99	9D	22	48	66	63	22	53				
	B3	25	54	58	6D	D4	B4	E2	2B	DC	C7	43	87	E3	68	80				
	8C	95	C1	03	23	42	24	ED	F3	8B	1C	64	39	2F	E6	72				
	F8	8F	17	45	09	7B	F8	80	78	51	86	C8	1E	D7	F0	E7				
RoundKey	63	0D	78	F1	04	09	71	83	53	5A	2B	A8	7B	22	09	A1	C1	E3	EA	4B
	DA	4F	6F	3D	10	5F	30	0D	6A	35	05	08	5F	6A	6F	67	8D	E7	88	EF
	3D	D2	84	10	F8	29	AD	BD	E0	C9	64	D9	6D	A4	C0	19	51	F5	35	2C
	FD	9F	68	C7	74	EB	83	44	98	73	F0	B4	5A	29	D9	6D	68	41	98	F5
After RoundKey	B1	80	89	94	F4	F7	F1	A5	CA	C7	09	E0	1E	41	2B	F2	83	60	1B	C2
	69	6A	3B	65	7D	8B	84	EF	41	E9	C2	4B	D8	89	07	D7	2A	22	86	8E
	B1	47	45	13	D8	6B	89	50	13	42	78	BD	54	8B	26	6B	A6	8A	15	11
	05	10	7F	82	7D	90	7B	C4	E0	22	76	7C	44	FE	29	8A	16	5A	23	50

Berikut ini adalah tampilan *ciphertext* yang dihasilkan :

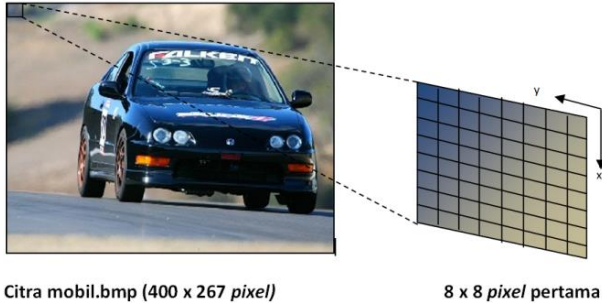
B3	60	1B	C2
2A	22	86	8E
A6	8A	15	11
16	5A	23	50
CIPHERTEXT			

6. Analisis Proses Steganografi Atau Penyisipan Pesan

Setelah proses pengenkripsian selesai maka tahap selanjutnya melakukan proses penyisipan pesan (*chipertext*) kedalam media citra "contrast.bmp" berukuran 640 x 246 pixel dengan metode LSB. Karena ukuran pesan (*chipertext*) adalah 16 byte maka (16 byte x 8 = 128 bit) dimana 1 karakter = 1 byte, maka

Pengamanan Citra Digital Menggunakan Kombinasi Antara Algoritma AES Dan Metode LSB (M Zunaidi)

jumlah *pixel* pada media citra tersebut yang dibutuhkan adalah $128 \text{ bit}/3 \text{ bit} = 8 \text{ pixel}$ pertama. Sebagai contoh dari citra $640 \times 246 \text{ pixel}$ ini akan diambil $8 \times 8 \text{ pixel}$ pertama, seperti yang terlihat pada gambar berikut ini:



Dari pengambilan $8 \times 8 \text{ pixel}$ pertama pada citra *contrast.bmp* ($400 \times 267 \text{ pixel}$) tersebut, diketahui nilai warnanya sebagai berikut:

The diagram shows a zoomed-in view of the top-left corner of the car image, represented as an 8x8 grid of pixels. The grid is labeled 'Citra mobil.bmp'.

X\Y	0	1	2	3	4	5	6	7
0	R=6 G=9 B=5	R=10 G=9 B=12	R=13 G=9 B=8	R=14 G=6 B=3	R=4 G=3 B=28	R=44 G=55 B=38	R=129 G=16 B=107	R=137 G=33 B=33
1	R=40 G=133 B=138	R=10 G=138 B=133	R=12 G=138 B=133	R=45 G=143 B=128	R=42 G=143 B=134	R=42 G=146 B=133	R=19 G=161 B=133	R=44 G=154 B=138
2	R=132 G=141 B=146	R=140 G=141 B=136	R=134 G=145 B=143	R=143 G=141 B=140	R=135 G=146 B=142	R=142 G=143 B=141	R=140 G=134 B=137	R=144 G=142 B=139
3	R=148 G=133 B=146	R=148 G=132 B=141	R=140 G=155 B=148	R=148 G=139 B=149	R=144 G=151 B=139	R=144 G=144 B=145	R=145 G=138 B=144	R=147 G=150 B=147
4	R=139 G=145 B=142	R=141 G=145 B=145	R=143 G=146 B=139	R=144 G=144 B=141	R=139 G=146 B=140	R=148 G=145 B=144	R=139 G=149 B=143	R=148 G=138 B=147
5	R=128 G=123 B=126	R=130 G=123 B=118	R=125 G=134 B=124	R=127 G=131 B=129	R=139 G=127 B=123	R=141 G=128 B=126	R=140 G=127 B=123	R=151 G=134 B=123
6	R=138 G=144 B=142	R=137 G=142 B=140	R=137 G=141 B=144	R=133 G=139 B=142	R=135 G=142 B=147	R=132 G=144 B=145	R=139 G=138 B=143	R=130 G=143 B=141
7	R=144 G=144 B=136	R=149 G=140 B=146	R=140 G=144 B=140	R=144 G=156 B=145	R=157 G=139 B=144	R=147 G=145 B=147	R=146 G=146 B=146	

Setelah itu konversikan nilai dari representasi warna pada gambar 3.23 kedalam biner, sehingga akan terbentuk nilai citra biner sebagai berikut.

Y / X	0	1	2	3	4	5	6	7
0	00000110	00001010	00001101	00001110	00000100	00101100	10000001	10001001
	00000110	00001001	00001001	00000110	00000011	00110111	00111000	00110101
	00000011	00001100	00001000	00000011	00100100	00100110	00011110	00100001
1	00101000	00011110	00100000	00101101	00111110	00111110	00111011	01000000
	10000101	10001010	10001010	10010001	10001110	10010010	10100001	10011010
	10000000	10000101	10000101	10000000	10000110	10000101	10000101	10001010
2	10000100	10001100	10000110	10001111	10000111	10001110	10001100	10010000
	10010011	10001101	10010001	10001101	10010010	10001111	10000110	10001110
	10010010	10001000	10001111	10001100	10001110	10001101	10001001	10001011
3	10010100	10010100	10001100	10010100	10010000	10010000	10010001	10010011
	10000111	10011000	10011011	10001011	10010111	10010000	10001010	10010110
	10010010	10001101	10010100	10010101	10001011	10010001	10010000	10010011
4	10001011	10001101	10001111	10010000	10001011	10010100	10001011	10010100
	10010001	10010001	10010010	10010000	10010010	10010001	10010101	10001010
	10001110	10010001	10001011	10001101	10001100	10010000	10001111	10010011
5	10000000	10000010	01111101	01111111	10001011	10001101	10001100	10010111
	01111011	01111011	10000110	10000011	01111111	10000000	01111111	10000110
	01111110	01110100	01111100	10000001	01111011	01111110	01111101	01111011
6	01111110	01111000	01111110	01110110	10000101	01111100	01111101	01111010
	10001010	10001001	10001001	10000100	10000100	10000100	10000001	10000010
	10010000	10001110	10001101	10001011	10001110	10010000	10001010	10001111
7	10001110	10001100	10010000	10001110	10010011	10010011	10010001	10001101
	10010000	10010101	10001100	10010000	10001000	10001001	10001011	10010010
	10001000	10010010	10001100	10010001	10010000	10010101	10010011	10010010

Lalu sisipkan nilai biner *chipertext* dari “B3601BC22A22868EA68A1511165A 2350” yaitu “10110011 01100000 00011011 11000010 00101010 00100010 10000110 10001110 10100110 10001010 00010101 00010001 00010110 01011010 00100011 01010000” kedalam *pixel* biner pada media citra contrast.bmp dengan menggunakan metode LSB, yaitu setiap 1 nilai bit *chipertext* dimasukkan untuk menggantikan posisi akhir dari citra RGB. Seperti yang terlihat pada tabel 3.2, yang merupakan hasil dari penyisipan nilai biner *chipertext* ke *pixel* biner pada citra contrast.bmp, dimana pada bit terakhir merupakan rangkaian bit-bit *ciphertext* yang telah disisipkan dan terdapat beberapa bit yang mengalami perubahan dengan bit-bit aslinya. Berikut tabel hasil penyisipan nilai biner *Ciphertext* ke pixel biner pada citra Mobil.bmp:

X/Y	0	1	2	3	4	5	6	7
0	00000111	00001011	00001101	00001111	00000100	00101100	10000000	10001000
	00000110	00001000	00001001	00000111	00000010	00110110	00111001	00110101
	00000011	00001100	00001000	00000010	00100100	00100110	00011111	00100001
1	00101001	00011110	00100001	00101100	00111111	00111110	00111011	01000000
	10000101	10001010	10001010	10010001	10001110	10010010	10100000	10011011
	10000000	10000100	10000100	10000000	10000111	10000100	10000100	10001010
2	10000101	10001100	10000111	10001110	10000111	10001110	10001101	10010001
	10010010	10001100	10010000	10001101	10010011	10001111	10000110	10001111
	10010010	10001001	10001111	10001100	10001111	10001100	10001000	10001010
3	10010101	10010100	10001101	10010100	10010000	10010001	10010000	10010010
	10000110	10011001	10011010	10001010	10010111	10010000	10001011	10010110
	10010010	10001100	10010100	10010101	10001010	10010000	10010000	10010011
4	10001010	10001101	10001111	10010001	10001011	10010100	10001011	10010100
	10010000	10010000	10010010	10010000	10010010	10010000	10010100	10001011
	10001110	10010001	10001010	10001101	10001101	10010000	10001110	10010011
5	10000000	10000011	01111100	01111111	10001011	10001101	10001100	10010111
	01111011	01111010	10000110	10000011	01111111	10000000	01111111	10000110
	01111110	01111010	01111100	10000001	01111011	01111110	01111101	01111011
6	01111110	01111000	01111110	01110110	10000101	01111100	01111101	01111010
	10001010	10001001	10001001	10000100	10000100	10000100	10000001	10000010
	10010000	10001110	10001101	10001011	10001110	10010000	10001010	10001111
7	10001110	10001100	10010000	10001110	10010011	10010011	10010001	10001101
	10010000	10010101	10001100	10010000	10001000	10001001	10001011	10010010
	10001000	10010010	10001100	10010001	10010000	10010101	10010011	10010010

Setelah penyisipan selesai dilakukan, konversikan kembali nilai biner citra tersebut ke nilai RGB pada citra digital. Dari proses penyisipan ini, maka akan terbentuk susunan nilai raster baru. Berikut adalah tabel raster baru yang sudah disisipkan pesan.

X/Y	0	1	2	3	4	5	6	7
0	R = 7	R = 11	R = 13	R = 15	R = 4	R = 44	R = 128	R = 136
	G = 6	G = 8	G = 9	G = 7	G = 2	G = 54	G = 57	G = 53
	B = 3	B = 12	B = 8	B = 2	B = 36	B = 38	B = 31	B = 33
1	R = 41	R = 30	R = 33	R = 44	R = 63	R = 62	R = 59	R = 64
	G = 133	G = 138	G = 138	G = 145	G = 142	G = 146	G = 160	G = 155
	B = 128	B = 132	B = 132	B = 128	B = 135	B = 132	B = 132	B = 138
2	R = 133	R = 140	R = 135	R = 142	R = 135	R = 142	R = 141	R = 145
	G = 146	G = 140	G = 144	G = 140	G = 147	G = 143	G = 134	G = 143
	B = 146	B = 137	B = 143	B = 140	B = 143	B = 140	B = 136	B = 138
3	R = 149	R = 148	R = 141	R = 148	R = 144	R = 145	R = 144	R = 146
	G = 134	G = 153	G = 154	G = 138	G = 151	G = 144	G = 139	G = 150
	B = 146	B = 140	B = 148	B = 149	B = 138	B = 144	B = 144	B = 147
4	R = 138	R = 141	R = 143	R = 145	R = 139	R = 148	R = 139	R = 148
	G = 144	G = 144	G = 146	G = 144	G = 146	G = 144	G = 148	G = 139
	B = 142	B = 145	B = 138	B = 141	B = 141	B = 144	B = 142	B = 147
5	R = 128	R = 131	R = 124	R = 127	R = 139	R = 141	R = 140	R = 151
	G = 123	G = 122	G = 134	G = 131	G = 127	G = 128	G = 127	G = 134
	B = 126	B = 116	B = 124	B = 129	B = 123	B = 126	B = 125	B = 123
6	R = 126	R = 120	R = 126	R = 118	R = 133	R = 124	R = 125	R = 122
	G = 138	G = 137	G = 137	G = 132	G = 132	G = 132	G = 129	G = 130
	B = 144	B = 142	B = 141	B = 139	B = 142	B = 144	B = 138	B = 143
7	R = 142	R = 140	R = 144	R = 142	R = 147	R = 147	R = 145	R = 141
	G = 144	G = 149	G = 140	G = 144	G = 136	G = 137	G = 139	G = 146
	B = 136	B = 146	B = 140	B = 145	B = 144	B = 149	B = 147	B = 146

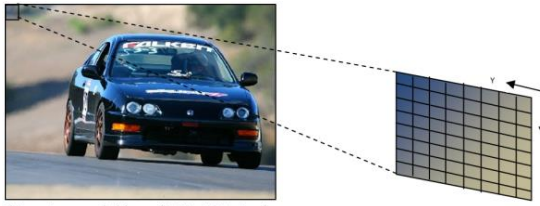
7. Proses Ekstraksi Pesan

Proses ekstraksi pesan merupakan proses pengungkapan atau pengambilan pesan rahasia yang terdapat di dalam citra stego. Proses pengambilan pesan rahasia di dalam citra stego ini merupakan kebalikan dari proses penyisipan pesan dengan LSB, dimana bit-bit LSB yang sekarang berada pada citra stego diambil satu persatu dan disatukan kembali sehingga menjadi sebuah informasi. Proses ini disebut dengan istilah *extraction* sehingga pesan akan dilakukan proses enkripsi dengan menggunakan kunci yang sama pada saat enkripsi pesan.



Misalkan gambar merupakan merupakan citra stego contrast.bmp dari hasil proses penyisipan pesan rahasia. Citra ini akan dilakukan proses ekstraksi pesan untuk mendapatkan kembali informasi yang disisipkan di dalamnya, dimana tahap-tahap proses ekstraksinya adalah sebagai berikut :

1. Citra stego tersebut akan dilakukan proses pengungkapan terhadap pesan rahasia di dalamnya. Sebagai contoh dari citra contrast.bmp ukuran 640 x 246 *pixel* ini, akan diambil 8 x 8 *pixel* pertama seperti yang terlihat pada gambar berikut:

Citra stego mobil.bmp (400 x 267 *pixel*)

2. Dari 8 x 8 *pixel* pertama pada citra stego mobil.bmp (400 x 267 *pixel*) tersebut, diketahui nilai representasi warnanya sebagai berikut ditunjukkan pada gambar citra mobil.bmp diatas.

X/Y	0	1	2	3	4	5	6	7
0	R=1 G=6 B=3	R=11 G=8 B=12	R=13 G=9 B=8	R=15 G=7 B=2	R=4 G=2 B=36	R=44 G=54 B=31	R=128 G=57 B=33	R=116 G=33 B=33
1	R=41 G=137 B=133	R=33 G=138 B=132	R=44 G=147 B=132	R=63 G=142 B=137	R=62 G=146 B=137	R=59 G=160 B=135	R=64 G=155 B=138	R=44 G=135 B=138
2	R=146 G=140 B=145	R=140 G=144 B=143	R=140 G=140 B=143	R=147 G=147 B=143	R=143 G=147 B=143	R=154 G=156 B=138	R=143 G=156 B=138	R=143 G=143 B=138
3	R=149 G=148 B=144	R=141 G=148 B=144	R=141 G=148 B=144	R=141 G=148 B=144	R=141 G=148 B=144	R=141 G=148 B=144	R=141 G=148 B=144	R=141 G=148 B=144
4	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144
5	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144
6	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144
7	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144	R=144 G=144 B=144

3. Konversi nilai dari representasi warna tersebut kedalam biner, sehingga akan terbentuk nilai yang ditunjukkan pada citra sebagai berikut :

X/Y	0	1	2	3	4	5	6	7
0	00000111 00000110 00000011	00001011 00001000 00001100	00001101 00001001 00001000	00001111 00000111 00000100	00000100 00110110 00100100	00101100 00111001 00010011	10000000 00111001 00011111	10001000 00110101 00100001
1	00101001 10000101 10000000	00011110 10001010 10000100	00100001 10001010 10000100	00101100 10010001 10000000	00111111 10001110 10000111	00111110 10100010 10000100	00111011 10100000 10000100	01000000 10011011 10001010
2	10000101 10010010 10010010	10001100 10001100 10001001	10000111 10001110 10001111	10001110 10001100 10001100	10000111 10010011 10001111	10001110 10001101 10001100	10001101 10001100 10001100	10010001 10001111 10001010
3	10010101 10000110 10010010	10010100 10011001 10001100	10001101 10001010 10001010	10010100 10001011 10001010	10010000 10010000 10010000	10010001 10001011 10010000	10010000 10001011 10010000	10010010 10010110 10010011
4	10001010 10010001 10001110	10001101 10010001 10001010	10001111 10001010 10001101	10001011 10001000 10001101	10001010 10010000 10010000	10001011 10001000 10001110	10001011 10001010 10001011	10010100 10001011 10001011
5	10000000 01111011 01111110	10000011 01111010 01111010	01111100 10000110 01111100	01111111 10000011 01111011	10001011 01111111 01111110	10001101 01111111 01111101	10001100 01111111 01111101	10010111 10000110 01111011
6	01111110 10001010 10010000	01111000 10001001 10001110	01111110 10000100 10001101	01111010 10000100 10001011	10000101 10000100 10010000	01111100 10000100 10001010	01111101 10000001 10001010	01111010 10000010 10001111
7	10001110 10010000 10010000	10001100 10010101 10001100	10010000 10001110 10010001	10001110 10010000 10010000	10010011 10010011 10010011	10010001 10010011 10010011	10010001 10010011 10010011	10001101 10010010 10010010

4. Dari data citra biner pada tabel 3.4, maka akan di ambil kembali bit-bit LSB yang disisipkan di dalamnya, nilai bit-bit LSB ini akan disatukan dan disusun kembali menjadi bentuk biner, sehingga terbentuk nilai-nilai LSB (*ciphertext*) yaitu "10110011 01100000 00011011 11000010 00101010 00100010 10000110 10001110 10100110 10001010 00010101 00010001 00010110 01011010 00100011 01010000"

5. Nilai biner yang terbentuk akan dikonversikan kedalam bilangan ASCII sehingga dapatlah *ciphertext* "B3601BC22A22868EA68A1511165A2350 "

B. Algoritma Dekripsi Aes-128

Proses dekripsi AES tidak jauh berbeda dengan proses enkripsi, untuk *state* yang dibentuk adalah hasil dari proses enkripsi dan melakukan proses dari kelebihan enkripsi. Untuk melakukan proses dekripsi empat transformasi yang dilakukan adalah *AddRoundKey*, *InvMixColumns*, *InvShiftRows* dan *InvSubBytes*. Berikut adalah hasil dari keseluruhan proses enkripsi.

B3	60	1B	C2
2A	22	86	8E
A6	8A	15	11
16	5A	23	50
CIPHERTEXT			

1. Transformasi AddRoundKey

Pada proses dekripsi *AddRoundKey*, *state* melakukan operasi XOR dengan kunci masukan yang sama pada saat melakukan proses enkripsi, dengan catatan untuk dekripsi kunci untuk pra ronde masukan adalah hasil ekspansi kunci ronde-10 pada proses enkripsi.

State		Key		Output																																																
<table><tr><td>B3</td><td>60</td><td>1B</td><td>C2</td></tr><tr><td>2A</td><td>22</td><td>86</td><td>8E</td></tr><tr><td>A6</td><td>8A</td><td>15</td><td>11</td></tr><tr><td>16</td><td>5A</td><td>23</td><td>50</td></tr></table>	B3	60	1B	C2	2A	22	86	8E	A6	8A	15	11	16	5A	23	50	$\oplus$	<table><tr><td>C1</td><td>E3</td><td>EA</td><td>4B</td></tr><tr><td>8D</td><td>E7</td><td>88</td><td>EF</td></tr><tr><td>51</td><td>F5</td><td>35</td><td>2C</td></tr><tr><td>68</td><td>41</td><td>98</td><td>F5</td></tr></table>	C1	E3	EA	4B	8D	E7	88	EF	51	F5	35	2C	68	41	98	F5	=	<table><tr><td>72</td><td>83</td><td>F1</td><td>89</td></tr><tr><td>A7</td><td>C5</td><td>0E</td><td>61</td></tr><tr><td>F7</td><td>7F</td><td>20</td><td>3D</td></tr><tr><td>7E</td><td>1B</td><td>BB</td><td>A5</td></tr></table>	72	83	F1	89	A7	C5	0E	61	F7	7F	20	3D	7E	1B	BB	A5
B3	60	1B	C2																																																	
2A	22	86	8E																																																	
A6	8A	15	11																																																	
16	5A	23	50																																																	
C1	E3	EA	4B																																																	
8D	E7	88	EF																																																	
51	F5	35	2C																																																	
68	41	98	F5																																																	
72	83	F1	89																																																	
A7	C5	0E	61																																																	
F7	7F	20	3D																																																	
7E	1B	BB	A5																																																	

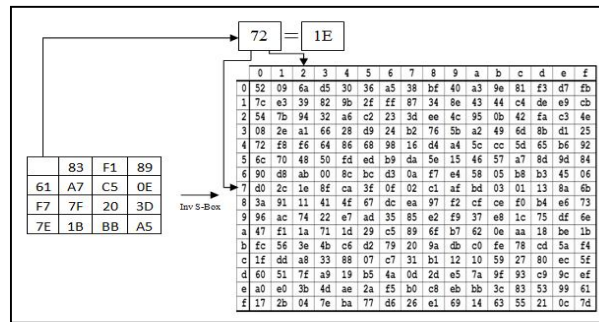
2. Transformasi InvShiftRows

Pada proses dekripsi *InvShiftRows*, *state* melakukan proses pergeseran posisi dari nilai *array* yang terdapat pada *state* seperti pada proses enkripsi *ShiftRows*, namun proses pergeseran *InvShiftRows* yaitu dimulai dari baris keempat dengan melakukan pergeseran satu blok nilai *array* dan baris kedua melakukan pergeseran sebanyak tiga blok *array*, baris pertama tidak ada pergeseran.

																Hasil Akhir Proses InvShiftRows																																																																											
<table border="1"> <tr><td>72</td><td>83</td><td>F1</td><td>89</td></tr> <tr><td>A7</td><td>C5</td><td>0E</td><td>61</td></tr> <tr><td>F7</td><td>7F</td><td>20</td><td>3D</td></tr> <tr><td>7E</td><td>1B</td><td>BB</td><td>A5</td></tr> </table>				72	83	F1	89	A7	C5	0E	61	F7	7F	20	3D	7E	1B	BB	A5	1 Byte >				<table border="1"> <tr><td>72</td><td>83</td><td>F1</td><td>89</td></tr> <tr><td>A7</td><td>C5</td><td>0E</td><td>61</td></tr> <tr><td>F7</td><td>7F</td><td>20</td><td>3D</td></tr> <tr><td>1B</td><td>BB</td><td>A5</td><td>7E</td></tr> </table>				72	83	F1	89	A7	C5	0E	61	F7	7F	20	3D	1B	BB	A5	7E	2 Byte >				<table border="1"> <tr><td>72</td><td>83</td><td>F1</td><td>89</td></tr> <tr><td>A7</td><td>C5</td><td>0E</td><td>61</td></tr> <tr><td>20</td><td>3D</td><td>F7</td><td>7F</td></tr> <tr><td>7E</td><td>1B</td><td>BB</td><td>A5</td></tr> </table>				72	83	F1	89	A7	C5	0E	61	20	3D	F7	7F	7E	1B	BB	A5	3 Byte >				<table border="1"> <tr><td>72</td><td>83</td><td>F1</td><td>89</td></tr> <tr><td>61</td><td>A7</td><td>C5</td><td>0E</td></tr> <tr><td>F7</td><td>7F</td><td>20</td><td>3D</td></tr> <tr><td>7E</td><td>1B</td><td>BB</td><td>A5</td></tr> </table>				72	83	F1	89	61	A7	C5	0E	F7	7F	20	3D	7E	1B	BB	A5
72	83	F1	89																																																																																								
A7	C5	0E	61																																																																																								
F7	7F	20	3D																																																																																								
7E	1B	BB	A5																																																																																								
72	83	F1	89																																																																																								
A7	C5	0E	61																																																																																								
F7	7F	20	3D																																																																																								
1B	BB	A5	7E																																																																																								
72	83	F1	89																																																																																								
A7	C5	0E	61																																																																																								
20	3D	F7	7F																																																																																								
7E	1B	BB	A5																																																																																								
72	83	F1	89																																																																																								
61	A7	C5	0E																																																																																								
F7	7F	20	3D																																																																																								
7E	1B	BB	A5																																																																																								

3. Transformasi InvSubBytes

Pada proses dekripsi, *InvSubBytes* sama dengan proses *SubBytes* pada enkripsi, akan tetapi proses *InvSubBytes* menggunakan tabel transformasi *InvSubBytes* (dapat dilihat pada pembahasan sebelumnya).



Berikut ini hasil Transformasi *InvSubBytes*:

1E	41	2B	F2
D8	89	07	D7
54	8B	26	6B
44	FE	29	8A

Selanjutnya hasil dari transformasi *InvSubBytes* di XOR kan dengan *AddRoundKey* lalu dilakukan proses transformasi *InvMixColumns*.

4. Transformasi *InvMixColumns*

Pada transformasi *InvMixColumns* dihitung dengan rumus penjumlahan dan perkalian serta menggunakan tabel E dan L *Galois Field Multiplication*. Transformasi *InvMixColumns* sama dengan *MixColumns*, dimana perbedaannya adalah $a(x)$ yang digunakan adalah inversnya (a^{-1}) yaitu :

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}$$

Sehingga :

$$s'(x) = a^{-1}(x) \oplus s(x) :$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix}$$

$$S'_{0,c} = (\{0e\} * S_{0,c}) \oplus (\{0b\} * S_{1,c}) \oplus (\{0d\} * S_{2,c}) \oplus (\{09\} * S_{3,c})$$

$$S'_{1,c} = (\{09\} * S_{0,c}) \oplus (\{0e\} * S_{1,c}) \oplus (\{0b\} * S_{2,c}) \oplus (\{0d\} * S_{3,c})$$

$$S'_{2,c} = (\{0d\} * S_{0,c}) \oplus (\{09\} * S_{1,c}) \oplus (\{0e\} * S_{2,c}) \oplus (\{0b\} * S_{3,c})$$

$$S'_{3,c} = (\{0b\} * S_{0,c}) \oplus (\{0d\} * S_{1,c}) \oplus (\{09\} * S_{2,c}) \oplus (\{0e\} * S_{3,c})$$

$$\begin{bmatrix} 66 \\ 87 \\ 39 \\ 1E \end{bmatrix} \cdot \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} = \begin{bmatrix} 74 \\ 1E \\ BC \\ 10 \end{bmatrix}$$

Berikut ini hasil transformasi InvMixColumns:

74	C6	01	E1
1E	25	B3	83
BC	7A	7D	2C
10	E1	93	38

Rumus di atas adalah rumus yang digunakan secara umum untuk mencari nilai invers dari *state* untuk proses *InvMixColumns*. Secara detail rumus dapat dijelaskan sebagai berikut dengan menggunakan tabel E dan L pada *Galois Field Multiplication*. Misal : Input = 66 87 39 1E
Output:

$$\begin{aligned}
 &= (66 * 0E) \oplus (87 * 0B) \oplus (39 * 0D) \oplus (1E * 09) \\
 &= E(L(66)+L(0E)) \oplus E(L(87)+L(0B)) \oplus E(L(39)+L(0D)) \oplus E(L(1E)+L(09)) \\
 &= E(1E+DF) \oplus E(74+68) \oplus E(F0+EE) \oplus E(1C+C7) \\
 &= E(FD) \oplus E(DC) \oplus E(1DE-FF) \oplus E(E3) \\
 &= E(FD) \oplus E(DC) \oplus E(DF) \oplus E(E3) \\
 &= 52 \oplus C6 \oplus 0E \oplus EE \\
 &= 74
 \end{aligned}$$

Setelah itu lakukan hal yang sama sampai iterasi ke-10 namun tidak dilakukan *MixColumns* lagi, melainkan langsung melakukan XOR hasil dari *InvShiftRows* dengan *RoundKey*.

4.Kesimpulan

1. Aplikasi yang digunakan untuk melakukan enkripsi dan dekripsi pesan yaitu menggunakan teknik kriptografi dengan algoritma AES.
2. Untuk melakukan proses penyisipan pesan, teknik steganografi yang digunakan dalam menyisipkan pesan rahasia yang sudah dienkripsi kedalam citra bitmap yaitu menggunakan teknik steganografi dengan metode LSB.
3. Pada proses ekstraksi Pesan, data yang sudah disisipkan kedalam media citra bitmap dapat diperoleh kembali dengan secara utuh.
4. Semakin besar ukuran citra digital yang digunakan maka semakin besar pula kapasitas pesan yang adapat disisipkan pada citra tersebut.

Daftar Pustaka

- [1] Rifki Sadikin, 2012, "*Kriptografi untuk Keamanan Jaringan*", Yogyakarta, Penerbit Andi
- [2] Dony Ariyus, 2009, "*Keamanan Multimedia*", Yogyakarta, Penerbit Andi
- [3] T. Sutoyo, S.Si., M.Kom., Edy Mulyanto, S.Si., M.Kom., Dr. Vincent Suhartono, Oky Dwi Nurhayati, [4]
- [4] M.T., Wijanarto, M.Kom, 2009, "*Teori Pengolahan Citra Digital*", Yogyakarta, Penerbit Andi
- [5] Wahana Komputer, 2009, "*Visual Basic 2008*", Yogyakarta, Penerbit Andi