Implementasi Algoritma AES 128 dan SHA – 256 Dalam Pengkodean pada Sebagian *Frame* Video CCTV MPEG-2

Anisha Yahdiani Mulyadi[1], Eddy Prasetyo Nugroho[2], Rizky Rahman J.P[3]

[#]*Departemen Pendidikan Ilmu Koputer, Universitas Pendidikan Indonesia*
Bandung, Indonesia

[1]anishayahdiani@gmail.com
[2,3]{eddypn,risky_rjp}@upi.edu

Abstrak - Perkembangan yang sangat pesat dalam pertukaran data pada teknologi internet dan informasi menyebabkan keamanan informasi menjadi masalah utama dalam penyimpanan data. Beberapa data memiliki informasi yang bersifat rahasia dan harus dilindungi, terutama dalam bentuk video yang mungkin mencakup beberapa informasi sensitif yang tidak diperuntukkan bagi konsumsi publik. Masalah muncul ketika informasi tersebut dapat dimanipulasi dan diubah keasliannya, salah satunya yaitu pada suatu file video *Closed Circuit Television* (CCTV). Oleh karena itu, tingkat keamanan dan privasi dalam suatu file video CCTV memiliki peranan yang vital. Penelitian ini membahas rancangan dan implementasi model pengamanan video CCTV MPEG-2 menggunakan teknik enkripsi. Proses enkripsi pada video CCTV akan menghasilkan video dengan sebagian *frame* yang acak sehingga dapat mencegah tindakan manipulasi. Metode enkripsi yang digunakan yaitu *Advanced Encryption Standard* (AES) 128 *bits* dan *Secure Hash Algorithm – 256* (SHA-256), dimana AES digunakan pada proses enkripsi dan dekripsi, sedangkan SHA-256 digunakan untuk meningkatkan kompleksitas pada kunci yang akan digunakan dan diuji oleh *Randomess Test*. Dengan menggunakan *Mean Squared Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR), model pengamanan video CCTV MPEG-2 yang dihasilkan menunjukkan kinerja yang cukup baik dengan nilai MSE dan PSNR maksimum secara berturut-turut yaitu 107,9 dan 27,8 dB pada hasil enkripsi. Sedangkan pada hasil dekripsi, nilai MSE dan PSNR maksimum secara berturut-turut yaitu 3,01 dan 43,43 dB.

Kata Kunci : Kriptografi, Metode AES128, SHA256, MPEG-2, enkripsi, dekripsi

I. PENDAHULUAN

Dengan semakin berkembangnya berkembangnya multimedia yang semakin pesat dan zaman yang semakin modern, maka suatu video memerlukan adanya keamanan dalam menjaga informasi yang ada pada video tersebut. Masalah keamanan merupakan salah satu aspek yang sangat penting dalam suatu data pada video. Selain ada pihak yang ingin menjaga agar data pada video tersebut tetap aman, namun ada juga pihak-pihak yang ingin agar dapat mengubah isi data pada video tersebut. Tidak semua video yang ada dibuat untuk konsumsi publik, banyak dari beberapa video yang bersifat pribadi yang hanya ditujukan untuk beberapa pihak tertentu saja. Beberapa contoh dari dari ancaman keamanan pada suatu informasi yang ada pada video seperti *modification* dan *fabrication*. Pada standar *National of Institute of Standards and Technology* (NIST) dalam SP 800-27 memiliki 33 prinsip tentang keamanan teknologi informasi, salah satu prinsipnya yaitu pada prinsip ke-26 untuk mengurangi ancaman serangan pada suatu informasi dengan menerapkan batasan hak akses [1]. Ini dapat diartikan menerapkan pembagian tugas yang tepat dan terbatas. Dengan melakukan pembatasan hak akses *(least privilege)* maka hanya pihak yang memegang kunci hak akses yang dapat mengakses suatu informasi dan dengan syarat hak akses itu secara aman diberikan kepada orang yang benar-benar memiliki hak tersebut.

Masalah yang dibahas pada penelitian ini yaitu merupakan tindakan manipulasi yang dapat mengubah keaslian data pada suatu *file* video CCTV. Sudah banyak kasus yang terjadi tentang data rekaman video CCTV yang dimanipulasi dan direkayasa untuk kepentingan pihak yang tidak berhak. CCTV *(Closed Circuit Television)* memainkan peran yang sangat signifikan dalam melindungi publik dan membantu kepolisian dalam menginvestigasi kasus kriminal [2]. Data yang dimanipulasi akan memiliki data yang berbeda, pada sebuah data yang dimanipulasi gambar atau *frame* yang didapatkan menjadi lebih kabur akibat pengurangan dari *frame* yang artinya file video tidak sama ukuran dengan yang asli di CCTV [3]. Data rekaman video yang telah dimanipulasi dapat menyebabkan kerugian yang sangat besar untuk beberapa pihak yang terkait. Karena sangat rentan dimanipulasi, keamanan pada data perlu ditingkatkan untuk melindungi data agar tetap terjaga keasliannya. Maka dalam penelitian ini akan diterapkan kriptografi video untuk meningkatkan keamanan pada data rekaman video, yaitu dengan menggabungkan algoritma AES dan fungsi *hash* SHA-256.

Kriptografi merupakan suatu ilmu mengenai teknik matematis yang ditujukan pada aspek pengamanan data yang meliputi tingkat kepercayaan terhadap data tersebut, integritas data, ontetikasi entitas data, otentifikasi terhadap keaslian data [4]. Terdapat empat tujuan mendasar pada ilmu kriptografi yang juga merupakan aspek dari keamanan informasi yaitu kerahasiaan, otentikasi, integritas data dan ketiadaan penyangkalan. Dalam penelitian ini akan mengacu pada kerahasiaan dan integritas data. Integritas data yaitu layanan yang menjamin bahwa data masih asli dan utuh atau belum pernah dimanipulasi [5].

Pada penelitian ini akan diimplementasikan algoritma AES dan fungsi *hash* SHA-256 pada proses enkripsi dan proses dekripsi sebagian *frame* video CCTV. Data video CCTV yang digunakan dalam format MPEG-2. AES memiliki komputasi yang ringan dan cocok digunakan dalam suatu proses enkripsi video dan dapat dikombinasikan dengan fungsi *hash* SHA-256 yang bertujuan meningkatkan kompleksitas kunci. Panjang kunci AES yang digunakan pada penelitian ini adalah kunci AES dengan panjang kunci 128 *bits*, 128 *bits* dipilih karena fleksibel dan dengan kunci 128 *bits*, AES tahan terhadap serangan *exhaustive key search*. Dengan penelitian yang akan dilakukan, diharapkan implementasi dari algoritma AES dan fungsi *hash* SHA-256 ini dapat menghasilkan sebuah aplikasi keamanan video yang memiliki tujuan untuk mencegah terjadinya proses manipulasi data pada video serta meningkatkan keamanan pada video.

II. PENELITIAN TERKAIT

Penelitian-penelitian berikut ini adalah penelitian yang berkaitan dengan kriptografi, MPEG-2, penggunaan algoritma AES-128 dan fungsi *hash* SHA-256 dalam pembuatan sistem keamanan video CCTV MPEG-2. Pada penelitian yang berjudul *"An Analysis and Comparison for Popular Video Encryption Algorithm"* mengenai perbandingan beberapa algoritma untuk enkripsi video yaitu, algoritma DES, RSA dan AES menyatakan bahwa sistem kriptografi kunci simetris memberi keamanan lebih tinggi dibandingkan sistem kriptografi kunci asimetris [6]. Dalam kasus ini algoritma DES telah terbukti sebagai algoritma yang aman. Tetapi algoritma AES memberikan keamanan yang lebih baik dan memiliki kebutuhan memori yang sangat rendah. Maka dari itu algoritma AES memiliki algoritma enkripsi yang paling efisien dan menjadi salah satu algoritma yang paling kuat dibandingkan algoritma lainnya.

Penelitian algoritma enkripsi lainnya yang berkaitan dengan algoritma AES yaitu pada implementasi algoritma AES-256 dan fungsi *hash* SHA-1 dalam pengamanan *file* [6]. Implementasi yang dilakukan pada penelitian ini dilakukan proses enkripsi yang dimulai dengan pemilihan *file input* dan kunci enkripsi. Setelah itu, proses dilanjutkan dengan menghitung nilai *hash* kunci dengan menggunakan metode SHA. Algoritma SHA-1 dapat digunakan untuk mengamankan kunci sebelum diterapkan dalam proses enkripsi dan dekripsi. Kombinasi dari algoritma AES-256 dan fungsi SHA-1 dapat meningkatkan keamanan dari pesan yang akan dikirimkan. Penelitian lainnya mengenai modifikasi algoritma AES untuk video MPEG-4, yaitu modifikasi pada Transformasi ShiftRow untuk enkripsi gambar. Tujuan dari modifikasi yang dilakukan dalam penelitian ini agar meningkatkan proses kinerja enkripsi [7].

Setelah mempelajari penelitian terdahulu maka dapat disimpulkan bahwa algoritma AES merupakan algoritma yang efisien dalam proses enkripsi dan dekripsi. Dan dengan menambahkan fungsi *hash* SHA-256 kunci pada algoritma AES dapat disamarkan terlebih dahulu sehingga dapat meningkatkan keamanan dari video. Maka implementasi yang dilakukan pada skripsi ini adalah implementasi algoritma enkripsi video dengan menggunakan algoritma kunci rahasia AES dan fungsi *hash* SHA-256.

III. *ADVANCED ENCRYPTION STANDARD*

Algoritma AES mendukung panjang kunci 128 *bits* sampai dengan 256 *bits* dengan *step* 32 *bits*. Panjang kunci dan ukuran blok dapat dipilih secara independen. Setiap blok dienkripsi dalam sejumlah putaran tertentu, sebagaimana halnya DES. Berikut pada Tabel I adalah banyaknya putaran kunci pada versi AES.

TABEL 1. PUTARAN KUNCI VERSI AES [5]

AES *(bits)*	**Panjang Kunci *(Nk Words)***	**Ukuran Blok *(Nb Words)***	**Jumlah Putaran *(Nr)***
128	4	4	10
192	6	4	12
256	8	4	14

Karena AES menetapkan panjang kunci adalah 128, 192 dan 256 maka dikenal sebagai AES-128, AES-192 dan AES-256. AES memiliki panjang kunci paling sedikit yaitu 128 *bits*, namun AES tetap tahan terhadap serangan *exhaustive key search* dengan teknologi saat ini. Dengan panjang kunci 128 *bits* maka terdapat sebanyak $2^{128} = 3{,}4 \times 10^{38}$ kemungkinan kunci. Seperti pada DES, Rijndael atau AES menggunakan substitusi dan permutasi dan sejumlah putaran atau *cipher* berulang. Setiap putaran menggunakan kunci internal yang berbeda. Empat proses utama algoritma AES yaitu sebagai berikut [8]:

a. SubBytes (Transformasi Substitusi Byte)
b. ShiftRow (Transfornasi Pergeseran Baris)
c. MixColumns (Transformasi Percampuran Kolom)
d. AddRoundKey (Transformasi Penambahan Kunci)

Algoritma AES memiliki tiga parameter yaitu sebagai berikut [5]:

a. Plainteks merupakan array yang berukuran 16-byte, yang berisi data masukan.
b. Cipherteks merupakan array yang berukuran 16-byte, yang berisi hasil dari enkripsi.
c. *Key* merupakan array berukuran 16-byte, yang berisi kunci *ciphering* (disebut juga *cipher key*).

IV. SHA – 256

SHA – 256 adalah salah satu algoritma *hash* yang relatif masih baru. Algoritma ini dirancang oleh *The National Institute of Standards and Technology* (NIST) pada tahun 2002. SHA – 256 menghasilkan *message digest* dengan panjang 256 *bits*. SHA – 256 tergolong aman karena didesain sedemikian rupa sehingga tidak memungkinkan mendapatkan pesan yang berhubungan dengan *message digest* yang sama. Proses untuk menghasilkan *message digest* pada algoritma ini meliputi lima tahapan [9].

1. *Message Padding*

   Input pesan pada algoritma SHA-256 akan dibagi menjadi blok-blok yang masing-masing panjangnya adalah 512 bit. Akibat dari pembagian ini maka jumlah blok terakhir akan lebih kecil atau sama dengan 512 bit. Blok terakhir tersebut akan mengalami *message padding.* Langkah-langkah *message padding* adalah sebagai berikut:

   a. Diawali dengan masuknya input pesan yang memiliki kode *American Standard Code for Information Interchange* (ASCII) dan kemudian diubah ke dalam bentuk biner rangkaian bit yang akan dihitung panjangnya.
   b. Rangkaian bit tersebut dibagi menjadi blok-blok yang masing-masing mempunyai panjang 512 bit. Hasil pembagian akan menyebabkan jumlah blok terakhir lebih kecil satu atau sama dengan 512 bit.
   c. Lakukan penambahan bit-bit isian *(padding)* pada blok terakhir pesan tersebut. Bit-bit yang digunakan sebagai bit isian adalah bit ‘1’ diikuti sejumlah bit ‘0’ sesuai dengan kebutuhan, dengan ketentuan sebagai berikut:
      - Jika panjang bit blok pesan terakhir lebih kecil dari 448 bit, maka ditambahkan bit ‘1’ pada posisi bit paling akhir, diikuti dengan beberapa bit ‘0’ sedemikian sehingga total panjang bit setelah proses tersebut adalah adalah 448 bit.
      - Jika panjang bit blok pesan terakhir lebih besar atau sama dengan 448 bit, maka ditambahkan bit ‘1’ pada posisi bit paling terakhir, diikuti dengan beberapa bit ‘0’ sedemikian sehingga total panjang bit setelah proses tersebut 512 bit. Kemudian dibuat 448 bit baru yang isi bitnya ‘0’.
      - Jika panjang bit blok pesan terakhir sama dengan 512 bit, maka harus dibuat blok baru untuk menampung proses *message padding*. Bit pertama dari blok baru diisi bit’1’, sedangkan bit-bit berikutnya sampai dengan panjang bit 448 diisi oleh bit ‘0’. Jumlah total bit isian yang ditambahkan adalah 448 bit.

2. Penambahan Panjang Bit

   Setelah proses *message padding*, jumlah bit pada blok terakhir adalah 448 bit. Representasikan M ke dalam bilangan biner untuk memperoleh 64 bit terakhirnya agar total panjang blok terakhir 512 bit.
   - Urutan byte paling kanan dari nilai representasi panjang pesan (M) dijadikan *low order*.
   - Tambahkan representasi M tersebut pada 448 bit terakhir, sehingga jumlah panjang blok terakhir adalah 512 bit.

3. Inisialisasi Nilai *Hash* Awal

   Pada SHA – 256 untuk menyimpan nilai inisialisasi awal dan nilai output sementara digunakan *buffer* H_0, H_1, H_2, H_3, H_4, H_5, H_6, H_7. Di sisi lain, untuk penyimpanan proses sementatat digunakan *buffer* a, b, c, d, e, f, g, h. Nilai H_0, H_1, H_2, H_3, H_4, H_5, H_6, H_7 untuk inisialisasi awal dalam notasi heksadesimal.

   TABEL 2. INISIALISASI AWAL DALAM NOTASI HEKSADESIMAL [9]

H0	6a09e667
H1	bb67ae85
H2	3c6ef372
H3	a54ff53a
H4	510e527f
H5	9b0588c
H6	1f83d9ab
H7	5be0cd19

4. Pemrosesan

   Pemrosesan merupakan bagian inti yang terdiri atas 1 *round* mempunyai 64 operasi. Untuk memproses setiap satu blok pesan 512 bit diperlukan 64 operasi. Setiap blok pesan $M^{(1)}$, $M^{(2)}$, ... , $M^{(n)}$ dengan N adalah jumlah blok pesan. Untuk setiap blok pesan akan dilakukan langkah-langkah:

   a. Persiapkan penjadwalan pesan.

b. Inisialisasi *working* variabel a, b, c, d, e, f, g dan h, untuk $M^{(1)}$ dengan nilai *hash* awal
   a = $H_0^{(I-1)}$
   $b = H_1^{(i-1)}$
   $c = H_2^{(i-1)}$
   $d = H_3^{(i-1)}$
   $e = H_4^{(i-1)}$
   $f = H_5^{(i-1)}$
   $g = H_6^{(i-1)}$
   $h = H_7^{(i-1)}$
c. Hitung masing-masing penjadwalan.
d. Menghitung nilai *hash* perantara untuk masing-masing blok pesan.

5. Output
   Output diperoleh setelah semua blok $M^{(N)}$ 512 bit diproses. Setelah semua langkah pemprosesan dilakukan sejumlah N kali, maka akan didapat 256 bit *message digest*.

V. PENGUJIAN *RANDOMNESS TEST*

Pengujian *Randomness Test* dilakukan setelah ditambahkan fungsi *hash* SHA-256 pada kunci dari AES 128 *bits* maka akan dibandingkan dan diujikan keacakan yang dibentuk dari hasil tersebut. Dari pelaksanaan *Randomness Test* terdapat dua kemungkinan yang dihasilkan yaitu jika data bersifat acak *(random)* terhadap median maka data tersebut homogen, sedangkan jika data memiliki kecenderungan *(trend)* lebih banyak di atas median atau di bawah median maka data tersebut tidak homogen. Pengujian *Randomess Test* dilakukan dengan bantuan dari *software* Cryptool 1.4.4 untuk menguji cipherteks yang dihasilkan oleh Algoritma AES 128 dengan kunci tanpa SHA – 256 dan cipherteks yang dihasilkan oleh Algoritma AES 128 dengan kunci yang telah di*hash*-kan oleh SHA – 256. Dan dengan urutan pengujian dari *randomness test* nya adalah: *Frequency Test, Poker Test, Long Run Test, Run Test, Serial Test.* Berikut adalah hasil uji keacakan dengan *Randomness Test* melalui bantuan aplikasi Cryptool 1.4.4 dengan nilai heksadesimal plainteks yang dienkripsi adalah *frame* pertama dari salah satu video CCTV MPEG-2 yang ada dan diberikan dua kondisi kunci.

1. Cipherteks yang dihasilkan oleh Algoritma AES 128 dengan kunci tanpa SHA – 256.

TABEL 3. HASIL UJI KEACAKAN CIPHERTEKS DENGAN KUNCI TANPA SHA – 256

Randomness Test		**Hasil *Randomness Test***	
Tipe	*Max. Value*	*Value*	*Status*
Frequency Test	3,841	0,062905	*Frequency Test Passed*
Poker Test	14,07000	16,23385	*Poker Test Failed*
Run Test	9,488000	12,902572	*Run Test Failed*
Long Run Test	34	17	*Long Run Test Passed*
Serial Test	5,991000	2,899519	*Serial Test Passed*

2. Cipherteks yang dihasilkan oleh Algoritma AES 128 dengan kunci yang telah di*hash*-kan dengan SHA – 256.

TABEL 4. HASIL UJI KEACAKAN CIPHERTEKS DENGAN KUNCI MENGGUNAKAN SHA – 256

Randomness Test		**Hasil *Randomness Test***	
Tipe	*Max. Value*	*Value*	*Status*
Frequency Test	3,841	0,256576	*Frequency Test Passed*
Poker Test	14,07000	2,283996	*Poker Test Passed*
Run Test	9,488000	5,209291	*Run Test Passed*
Long Run Test	34	18	*Long Run Test Passed*
Serial Test	5,991000	0,681086	*Serial Test Passed*

Dapat dilihat dari Tabel III dan Tabel IV bahwa pada cipherteks yang diuji tanpa SHA – 256 hasil pengujian *randomness test* ada dua pengujian yang tidak berhasil yaitu pada *poker test* dan *run test*. Hasil dari kedua uji tersebut menghasilkan hasil yang lebih dari *max value* yang telah ditentukan. Sedangkan pada cipherteks yang diuji dengan menggunakan SHA – 256 pada semua pengujian yang telah dilakukan berhasil yang berarti tidak melebihi batas dari *max value*. Dapat disimpulkan bahwa kunci AES 128 dengan menggunakan fungsi *hash* SHA – 256 memiliki hasil yang lebih baik karena telah lolos semua proses pengujian yang ada pada *Randomness Test*.

VI. HASIL PEMBAHASAN

Pada penelitian ini akan digunakan pengujian faktor *fidelity* dengan 2 buah cara yaitu *Mean Squared Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR) untuk membuktikan bahwa gambar yang telah dienkripsi dan dekripsi dapat dilihat menggunakan histogram gambar, yang dalam penelitian ini yaitu pada sekumpulan *frame* video CCTV MPEG-2. MSE merupakan parameter yang menunjukkan tingkat kesalahan piksel-piksel citra hasil pemrosesan signal terhadap citra asli [10]. Jika dalam perhitungan steganografi semakin kecil nilai MSE menandakan gambar yang sudah diberi *watermark* memiliki tingkat *fidelity* semakin baik, sedangkan pada perhitungan kriptografi pada hasil enkripsi semakin besar hasil nilai yang telah dienkripsi hasilnya diatas 30 maka tingkat error lebih besar yang artinya juga proses pengacakan berhasil dan hasil lebih baik. Dan jika hasil dekripsi dibawah 30 maka hasil proses dekripsi berhasil karena *frame* yang telah didekripsi hasilnya sama dengan *frame* yang asli.

M = Panjang citra *frame* (dalam piksel)

I(x,y) = nilai piksel dari *frame* cover

N = Lebar citra *frame* (dalam piksel)

I'(x,y) = nilai piksel pada *frame*

$$MSE = \frac{1}{MN}\sum_{X=1}^{M}\sum_{y=1}^{N}\left|\left(f(x,y)-g(x,y)\right)^2\right|$$

PSNR merupakan perhitungan untuk mengetahui perbandingan dari nilai maksimal sinyal gambar atau dari penelitian ini yaitu sekumpulan *frame* yang diambil dari video CCTV MPEG-2 yang telah dipecah sehingga akan diketahui nilai *frame* sebelum dan sesudah disisipi. PSNR merupakan parameter besaran yang menunjukkan rasio tingkat toleransi *noise* tertentu terhadap banyaknya *noise* pada suatu piksel citra [10]. *Noise* yang dimaksudkan di sini adalah kerusakan piksel pada bagian tertentu dalam sebuah citra sehingga mempengaruhi kualitas dari pada piksel tersebut. Dengan kata lain PSNR menunjukkan nilai kualitas suatu piksel citra. Persamaan untuk menghitung nilai komponen PSNR dari sebuah citra dapat dirumuskan sebagai berikut.

$$PSNR = {}^{10}\log 10\frac{255^2}{MSE}$$

PSNR memiliki satuan berupa desibel (dB). PSNR yang lebih tinggi menunjukan kualitas citra hasil keluaran lebih baik atau dapat dikatakan menyerupai citra aslinya. Pengujian pada PSNR didapatkan hasilnya dari perhitungan hasil MSE. Pada steganografi nilai PSNR semakin besar maka akan semakin baik. Namun pada kasus kriptografi, untuk hasil enkripsi jika nilai PSNR lebih rendah dari 30 dB maka hasil akan lebih baik dan *frame* yang telah dienkripsi berarti tidak sama dengan *frame* asli. Dan jika hasil dekripsi memiliki hasil PSNR lebih dari 30 dB maka implementasi algoritma AES 128 dalam proses dekripsi berhasil dengan sangat baik karena *frame* dekripsi menyerupai dengan *frame* asli.

Dataset *frame* didapatkan dari empat buah video CCTV MPEG-2 yang akan diambil dari masing-masing video sebanyak tiga *frame* yaitu, *frame* enkripsi yang pertama, tengah dan terakhir yang telah dipecah. Dapat dilihat dataset *frame* pada Gambar 1, Gambar 2, Gambar 3 dan Gambar 4.

Gambar 1. Dataset *frame* video CCTV (1)

Gambar 2. Dataset *frame* video CCTV (2)

Gambar 3. Dataset *frame* video CCTV (3)

Gambar 4. Dataset *frame* video CCTV (4)

Pengujian faktor *fidelity* yaitu yang pada penelitian ini menggunakan MSE dan PSNR dilakukan dengan menganalisa hasil *frame* yang telah dienkripsi dan hasil *frame* yang telah didekripsi. Berikut pada Tabel 5 adalah hasil pengujian dari *frame* yang telah dienkripsi.

TABEL 5. EVALUASI HASIL MSE DAN PSNR PADA *FRAME* YANG TELAH DIENKRIPSI

Nama Video	Keterangan *Frame*	MSE	PSNR
Video 1	*Frame* Pertama	3,0148	43,3967 dB
	Frame Tengah	3,9614	42,1190 dB
	Frame Terakhir	3,7909	42,3929 dB
Video 2	*Frame* Pertama	4,7798	41,4287 dB
	Frame Tengah	6,2139	40,2680 dB
	Frame Terakhir	6,4009	40,1573 dB
Video 3	*Frame* Pertama	8,9428	38,6535 dB
	Frame Tengah	12,.2556	37,2835 dB
	Frame Terakhir	12,3562	37,2485 dB
Video 4	*Frame* Pertama	3,3717	42,8984 dB
	Frame Tengah	5,1394	41,0723 dB
	Frame Terakhir	5,7185	40,6014 dB

Hasil MSE dari *frame* yang telah dienkripsi menunjukan hasil yang baik dalam kriteria kriptografi yaitu dengan MSE yang paling baik yang berarti *frame* semakin acak dengan nilai 107,1931 dan dengan nilai terendahnya yang masih di atas 30 yaitu 50,8915. Dan hasil PSNR dari frame yang telah dienkripsi pada memperlihatkan bahwa hasil paling baik yaitu dengan nilai 27,8639 dB yang berarti PSNR dibawah 30 dB. Dan untuk rata-rata nilai dari *frame* lainnya ada yang diatas 30 dB namun tidak lebih dari 32 dB yang berarti masih mendekati 30 dB. Setelah diteliti, hasil *frame* yang lebih dari 30 dB memiliki resolusi yang sangat baik pada framenya dibandingkan frame yang dibawah 30 dB. Dari hasil pengujian di atas maka dapat disimpulkan bahwa algoritma AES 128 terbukti sangat baik dilihat dari hasil MSE dan PSNR yang telah diuji pada *frame* yang telah dienkripsi. Berikut pada Gambar 5 adalah contoh perbandingan dari *frame* asli dan *frame* enkripsi yang telah acak.

TABEL 6. EVALUASI HASIL MSE DAN PSNR PADA *FRAME* YANG TELAH DIDEKRIPSI

Nama Video	Keterangan *Frame*	MSE	PSNR
Video 1	*Frame* Pertama	51,8727	31,0411 dB
	Frame Tengah	50,5650	31,1669 dB
	Frame Terakhir	50,8915	31,1352 dB
Video 2	*Frame* Pertama	74,5439	29,4426 dB
	Frame Tengah	68,2075	29,8290 dB
	Frame Terakhir	62,3727	30,2177 dB
Video 3	*Frame* Pertama	104,1035	27,9906 dB
	Frame Tengah	107,1931	27,8639 dB
	Frame Terakhir	106,5236	27,8909 dB
Video 4	*Frame* Pertama	54,7336	30,7830 dB
	Frame Tengah	54,6626	30,7883 dB
	Frame Terakhir	54,2211	30,8237 dB

Setelah melakukan pengujian pada hasil *frame* yang telah dienkripsi maka setelah itu melakukan pengujian terhadap *frame* yang telah didekripsi untuk melihat apakah hasil *frame* yang telah didekripsi sesuai dengan *frame* asli. Berikut pada Tabel VI adalah hasil pengujian dari *frame* yang telah dienkripsi.

Hasil MSE dari *frame* yang telah didekripsi memiliki hasil paling baik dengan nilai 3,0148 yang berarti nilai MSE dibawah 30. Dan rata-rata hasil dari *frame* lainnya juga menunjukkan hasil dibawah 30 yang berarti hasil *frame* dekripsi dengan *frame* asli menggunakan algoritma AES 128 *bits* berjalan dengan baik. Dan pada hasil PSNR

dapat dilihat dari Tabel 6 memperlihatkan dari empat video yang diuji mendapatkan hasil diatas 30 db dengan nilai paling baik yaitu pada video 1 *frame* awal sebesasr 43,4967 dB yang berarti kualitas *frame* yang telah didekripsi menyerupai dengan *frame* asli. Berikut pada

Gambar 5 Perbandingan *frame* asli dan *framem* enkripsi

Gambar 6 adalah contoh perbandingan *frame* asli dan *frame* dekripsi.

Gambar 6. Perbandingan *frame* asli dan *frame* dekripsi

VII. KESIMPULAN

Berdasarkan penelitian ini, pengujian dan analisis terhadap sistem, maka didapatkan kesimpulan sebagai berikut:

1. Metode AES 128 *bits* dapat diimplementasikan untuk proses enkripsi dan proses dekripsi pada video MPEG-2 dengan melakukan proses ekstraksi video menjadi *frame* dahulu dan mengkonversikan *frame* menjadi blok data dari *frame* asli setiap sebesar 128 *bits* atau 16 piksel sampai dengan ukuran maksimum *frame* asli tersebut.
2. Metode SHA-256 terbukti dapat meningkatkan kompleksitas kunci pada AES 128 *bits* terlihat dari hasil uji *Randomness Test* bahwa kunci pada AES 128 *bits* yang menggunakan SHA-256 telah lolos semua tes uji sedangkan yang tidak menggunakan SHA-256 pada tahap Uji Poker gagal dan Uji Run *gagal*.
3. Hasil Uji Faktor *Fidelity*
   a. Berdasarkan evaluasi hasil pengujian *frame* asli dan *frame* enkripsi dari total empat dataset *frame* awal, tengah dan akhir dari masing-masing video, dua dari video yang diuji menghasilkan nilai PSNR pada kriptografi yang ideal yaitu di bawah 30 dB dengan nilai terbaik sebesar 27,8639 dB. Sedangkan untuk nilai MSE semua dataset *frame* yang diuji menghasilkan nilai MSE yang ideal yaitu diatas 30 yang berarti tingkat keacakan semakin error dan dengan nilai MSE terbaik sebesar 107,1931.
   b. Sedangkan berdasarkan evaluasi hasil pengujian *frame* asli dan *frame* dekripsi dari total empat dataset *frame* awal, tengah dan akhir dari masing-masing video, semua dataset *frame* yang diuji memiliki hasil PSNR yang ideal yaitu diatas 30 dB dengan nilai terbaik sebesar 43,3967. Sedangkan untuk nilai MSE juga memiliki hasil yang ideal yaitu di bawah 30 dengan nilai MSE terbaik sebesar 3,0148.

REFERENSI

[1] G. Stoneburner, C. Hayden, and A. Feringa, "Information Technology Security (A Baseline for Achieving Security)."

[2] H. U. Keval, "Effective design, configuration, and use of digital CCTV," *Crit. Rev.*, no. April, pp. 1–289, 2009.

[3] Fitri N. Heriani, "Kejanggalan Bukti CCTV Versi Ahli Digital Forensik dari Kubu Jessica," *Hukum Online*, 2016. [Online]. Available: http://www.hukumonline.com/berita/baca/lt57da722ae99c3/kejanggalan-bukti-cctv-versi-ahli-digital-forensik-dari-kubu-jessica. [Accessed: 01-Jun-2017].

[4] A. Menezes, P. van Oorschot, and S. Vanstone, "Handbook of Applied Cryptography," vol. 19964964, 1996.

[5] R. Munir, *Kriptografi*. Bandung: Informatika, 2006.

[6] I. F. A. SIREGAR, H. R. SYAHFITRI, and TOMMY, "Aplikasi Pengamanan File Dengan Algoritma Aes256 Dan Sha1," no. 70, pp. 1–8, 2010.

[7] P. Deshmukh and V. Kolhe, "Modified AES Based Algorithm for MPEG Video Encryption," no. 978, 2014.

[8] R. Munir, *Advanced Encryption Standard (AES)*. Bandung: Bandung: Departemen Teknik Informatika Institut Teknologi, 2004.

[9] M. Syafriadi, "Analisis Kecepatan dan Keamanan Algoritma Secure Hash Algorithm 256 (SHA-256) Untuk Otentikasi Pesan Teks." 2006.

[10] P. L. T. Irawan, D. J. D. H. Santjojo, and M. Sarosa, "Implementasi Kripto-Steganografi Salsa20 dan BPCS untuk Pengamanan Data Citra Digital," *J. EECCIS*, vol. 8, no. 2, pp. 175–180, 2014.