

Steganografi Video Digital dengan Algoritma LSB (Least Significant Bit) dan Rijndael

By Dwi Aryanto

Steganografi Video Digital dengan Algoritma LSB (Least Significant Bit) dan Rijndael

Digital Video Steganography with LSB Algorithm (Least Significant Bit) and Rijndael

1

Abstrak

Steganografi adalah ilmu dan seni untuk menyembunyikan informasi sehingga informasi yang bersifat rahasia tidak dapat diketahui oleh orang lain, kecuali pengirim dan penerima. Penelitian dirancang untuk membuat sistem steganografi pada video dengan format mp4. Pesan 9 disisipkan pada salah satu frame video dengan terlebih dahulu dienkripsi dengan algoritma Rijndael. Penyisipan pesan pada frame video menggunakan metode *Least significant Bit* (LSB). Ekstraksi frame pada video menggunakan *software* ffmpeg. Pengujian kualitatif dilakukan untuk melihat perubahan frame video dengan indera manusia. Pengujian kuantitatif dilakukan untuk menguji enam video dengan resolusi yang berbeda, sedangkan yang disisipkan ada lima pesan dengan panjang Byte yang bervariasi. Frame yang telah disisipi pesan diukur *noise*-nya dengan *Peak Signal to Noise Ratio* (PSNR). Hasil pengujian menunjukkan bahwa metode LSB tidak bisa digunakan untuk penyisipan pesan yang ukuran Byte-nya lebih besar dari daya tampung video cover. Perubahan kualitas citra akan terjadi apabila ukuran Byte pesan yang disisipkan semakin besar.

Kata Kunci: LSB, PSNR, Steganografi.

Abstract

18

Steganography is the science and art of hiding information so that confidential information cannot be known by others, except the sender and receiver. This research is designed to create a steganography system on video with mp4 format. The message is inserted in one of the video frames, first encrypted with the Rijndael algorithm. The method of inserting messages in the video frame is the Least Significant Bit (LSB) method. Extraction of frames on video using ffmpeg software. Qualitative testing used to identify changes in video frames with human senses. Quantitative testing was carried out by testing six videos with different resolutions, while there were five inserted messages with varying Byte lengths. The frame that has been inserted with a message is measured for noise with the Peak Signal to Noise Ratio (PSNR). The test results show that the LSB method cannot be used for message insertion whose Byte size is greater than the capacity of the cover video. There is a change in image quality if the size of the inserted message bytes is getting bigger.

Keywords: LSB, PSNR, Steganography.

1

Pendahuluan

Informasi merupakan sesuatu yang sangat berharga. Pentingnya kerahasiaan informasi telah menjadi perhatian tersendiri. Berbagai cara digunakan untuk merahasiakan informasi karena jika jatuh ke orang yang tidak berhak akan menimbulkan kerugian. Zaman sekarang informasi tidak hanya dapat disandikan tetapi dapat juga disisipkan kedalam media digital. Teknik menyisipkan pesan dikenal dengan nama steganografi. Steganografi merupakan salah satu cara untuk menyembunyikan pesan kedalam media digital yang secara indra manusia tampak tidak mengandung apa-apa, kecuali bagi orang yang mengerti caranya. Steganografi membutuhkan dua properti, yaitu media penampung dan pesan rahasia. Steganografi dapat digunakan pada berbagai macam media digital, yaitu citra, suara, maupun video [1].

Alum dan Ibrahim [2] memaparkan steganografi sebagai ilmu dan seni untuk menyembunyikan informasi bersifat rahasia sehingga tidak dapat diketahui oleh orang lain, kecuali pengirim dan penerima. Proses steganografi biasanya melibatkan penyandian atau kriptografi. Proses yang dilakukan yaitu dengan enkripsi *plaintext* terlebih dahulu menjadi *Byte cipher* atau pesan rahasia. Kemudian *Byte cipher* disisipkan pada media digital berupa teks, audio, citra. Steganografi merupakan mekanisme untuk menyembunyikan data seperti gambar dan file yang lain dengan file yang lain. Gabungan steganografi dan kriptografi merupakan mekanisme pengamanan data dengan keamanan penuh [3]. Steganografi video dilakukan

dengan mengekstraksi semua frame video kemudian dipilih salah satu frame atau beberapa frame secara acak untuk menyembunyikan pesan yang akan disisipkan [4][5].

Algoritma Rijndael merupakan algoritma yang dibuat oleh Dr. Vincent Rijmen dan Dr. Joan Daemen yang pada tahun 2000 secara resmi dipilih oleh NIST (*National of Standard and Technology*) sebagai *Advanced Encryption Standard* (AES) mengalahkan beberapa algoritma lainnya karena *Rijndael* merupakan algoritma yang memiliki keseimbangan antara keamanan dan fleksibilitas dalam berbagai platform software serta hardware. Beberapa evaluasi yang dilakukan oleh NIST terhadap algoritma Rijndael yaitu: 1. Tidak ada serangan yang diketahui bisa memecahkan *Rijndael*, 2. *Rijndael* menggunakan komponen kotak-S non linier, 3. Enkripsi dan dekripsi *Rijndael* berbeda satu dengan yang lain, 4. *Rijndael* sepenuhnya mendukung ukuran blok dan ukuran kunci 128 bit, 192 bit, serta 256 bit. Pada algoritma *Rijndael* enkripsi terdiri dari empat jenis transformasi Byte, yaitu *SubBytes*, *ShiftRows*, *Mixcolumns*, dan *AddRoundKey*. Tahap awal enkripsi, dalam state mengalami transformasi Byte *addRoundKey*, yaitu melakukan X-OR antara state awal (*plaintext*) dengan *cipher key*. Tahap ini disebut juga *initial round*. Setelah itu state mengalami putaran transformasi sebanyak $n - 1$ kali [6].

Least Significant Bit (LSB) adalah teknik penyembunyian pesan dengan cara menyisipkan pesan pada bit rendah atau bit paling kanan pada file media penampung sebagai media untuk menyembunyikan pesan. LSB adalah barisan data biner yang ada pada media digital dan paling tidak terlalu berpengaruh terhadap perubahan jika nilai datanya dimodifikasi. LSB merupakan salah satu teknik dalam steganografi dengan menambahkan bit data yang akan disembunyikan (pesan) di bit terakhir yang paling cocok atau kurang berarti [7][8].

Misalkan bit pada image dengan ukuran 3 pixel sebagai berikut:

```
(0011111 11101001 11001000)
(0011111 11001000 11101001)
(1100000      00100111      11101001)
```

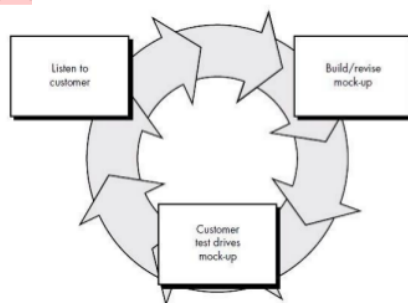
Pesan yang akan disisipkan adalah karakter 'A' yang memiliki biner 10000001, stego image yang dihasilkan adalah:

```
(0011111 11101000 11001000)
(0011110 11001000 11101000)
(3)100000      00100111      11101001)
```

MPEG-4 sub-bagian 14 atau lebih dikenal sebagai MP4 adalah salah satu format berkas pengkodean suara dan gambar/video digital yang dikeluarkan oleh organisasi MPEG. Ekstensi nama berkas jenis MPEG-4 ini banyak menggunakan .mp4 yang merupakan pengembangan dari format QuickTime dari komputer Apple [9].

1 Metode Penelitian

Metode yang dipakai dalam penelitian ini adalah *prototype* model. Bagan mengenai *prototype* model dapat dilihat pada Gambar 1.



Gambar 1 Prototype model

Tahap-tahap dalam *prototype* model menurut Lukman (2016) adalah sebagai berikut:

1. *Listen to Customer*: analisis terhadap permasalahan yang ada, yaitu mendapatkan data dan literatur yang terkait dengan proses *embedding*, ekstraksi, enkripsi dan dekripsi data teks pada video.
2. *Buid/revise mock-up*: perancangan mengenai sistem yang akan dibangun. Selain itu dilakukan pula perancangan *user interface* dan algoritma.
3. *Customer test drives mock-up*: pengujian sistem yaitu menjalankan proses implementasi sistem dan melakukan analisis kualitatif dan kuantitatif terhadap video stego.

Hasil dan Pembahasan

Hasil rancangan sistem diimplementasikan menggunakan bahasa pemrograman *Visual Basic 2010*. Video cover dalam penelitian ini adalah video dengan format mp4 pesan yang disisipkan berupa teks. Proses ekstraksi frame dan audio menggunakan *ffmpeg*. Proses ekstraksi frame dari video dilakukan dengan mengekstrak semua frame yang ada di video dalam format BMP. Informasi jumlah frame dalam video akan menghasilkan jumlah yang sama ketika video diekstrak menjadi frame. Frame dan audio hasil ekstraksi disimpan dalam folder. Perintahnya adalah sebagai berikut:

```
ffmpeg -i " + OFD.FileName + " -vsync cfr -r 25 -f image2 img%4d.bmp -acodec aac audio_out.aac
```

Setelah audio dan semua frame dalam video terekstrak, diambil salah satu frame yang akan disisipi pesan, frame yang disisipi adalah frame ke-10. Pesan yang disisipkan adalah pesan teks dengan format txt atau pesan yang langsung ditulis sekaligus memasukan kunci enkripsi. Proses selanjutnya adalah menyatukan kembali audio dan frame menjadi video kembali. Perintahnya adalah sebagai berikut:

```
Ffmpeg -r 25 -f image2 -i img%04d.bmp -i audio_out.aac -vcodec libx264 -crf 25 -pix_fmt yuv420p -acodec copy Video_out.mp4 -y.
```

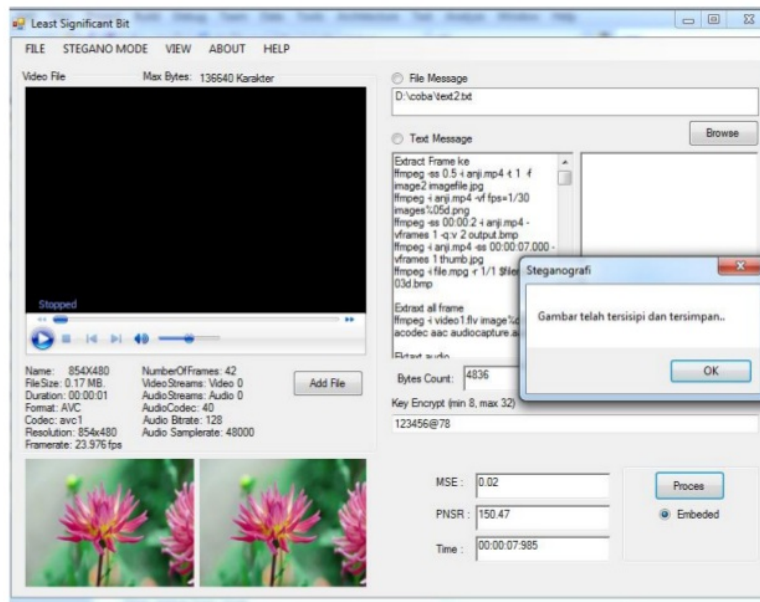
Pesan yang akan disisipkan pada salah satu frame video terlebih dahulu dienkripsi dengan algoritma *Rijndael* menggunakan kunci 128 bit. Proses untuk enkripsi menggunakan *function encrypt*. Kode Program 1 menunjukkan proses enkripsi *plaintext* menggunakan algoritma *Rijndael*.

Kode Program 1. Algoritma enkripsi Rijndael

```
1 Public Shared Function Encrypt _  
2 ( _  
3 ByVal plainText As String, _  
4 ByVal passPhrase As String, _  
5 ByVal saltValue As String, _  
6 ByVal hashAlgorithm As String, _  
7 ByVal passwordIterations As Integer, _  
8 ByVal initVector As String, _  
9 ByVal keySize As Integer _  
10 )  
11 As String  
12 Dim initVectorBytes As Byte()  
13 initVectorBytes = Encoding.ASCII.GetBytes(initVector)  
14 Dim saltValueBytes As Byte()  
15 saltValueBytes = Encoding.ASCII.GetBytes(saltValue)  
16 Dim plainTextBytes As Byte()  
17 plainTextBytes = Encoding.UTF8.GetBytes(plainText)  
18 Dim password As PasswordDeriveBytes  
19 password = New PasswordDeriveBytes _  
20 ( _  
21 passPhrase, _  
22 saltValueBytes, _  
23 HashAlgorithm, _  
24 passwordIterations _  
25 )  
26 Dim keyBytes As Byte()  
27 keyBytes = password.GetBytes(keySize / 8)  
28 Dim symmetricKey As RijndaelManaged  
29 symmetricKey = New RijndaelManaged()  
30 symmetricKey.Mode = CipherMode.CBC  
31 Dim encryptor As ICryptoTransform  
32 encryptor = symmetricKey.CreateEncryptor(keyBytes, initVectorBytes)  
33 Dim memoryStream As MemoryStream  
34 memoryStream = New MemoryStream()  
35 Dim cryptoStream As CryptoStream  
36 cryptoStream = New CryptoStream _  
37 ( _  
38 memoryStream, _  
39 encryptor, _  
40 CryptoStreamMode.Write _  
41 )  
42 cryptoStream.Write(plainTextBytes, 0, plainTextBytes.Length)  
43 cryptoStream.FlushFinalBlock()  
44 Dim cipherTextBytes As Byte()  
45 cipherTextBytes = memoryStream.ToArray()  
46 memoryStream.Close()  
47 cryptoStream.Close()  
48 Dim cipherText As String  
49 cipherText = Convert.ToBase64String(cipherTextBytes)  
50 Encrypt = cipherText  
51 End Function
```

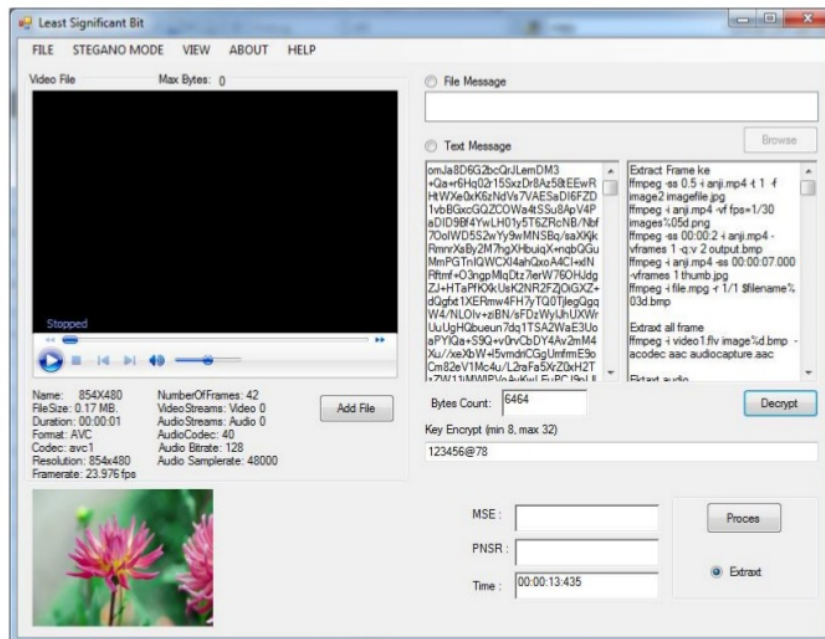
Steganografi menggunakan *LSB* diartikan sebagai penyisipan bit-bit pesan ke citra bitmap (frame video stego) sebagai media penampung pesan. Untuk menyisipkan pesan dibutuhkan data masukan antara lain citra sebagai media untuk menampung pesan, pesan berupa *plaintext* berkarakter *ASCII*, penanda pesan untuk mengenali pesan pada citra *stego*, dan kunci simetris 128 bit sebagai kunci untuk enkripsi *plaintext*

untuk menghasilkan *Byte cipher*. *LSB* menyisipkan bit stegano tidak pada setiap *Byte* citra melainkan bit-bit stegano disisipkan pada *Byte* citra yang memiliki nilai *Byte* 255. Tahapan untuk penyisipan bit-bit pesan dimulai dengan mengecek jumlah *Byte* citra yang memiliki nilai *Byte* citra bernilai 255 pada setiap indeks *width* dan *height* untuk citra yang digunakan sebagai media penampung bit stegano. Kemudian dilakukan pengukuran jumlah pesan yang disisipkan pada *Byte* citra. Selanjutnya hasil pengukuran jumlah dari masing-masing masukan dibandingkan. Jika hasil perbandingan jumlah *Byte* stegano lebih besar maka proses penyisipan tidak dapat dikerjakan, sebaliknya ¹⁷ ukuran pesan lebih kecil dari jumlah *Byte* citra yang bernilai 255 maka proses penyisipan dikerjakan. Proses Penyisipan pesan dapat dilihat pada Gambar 2.



Gambar 2. Proses penyisipan pesan dengan Metode LSB


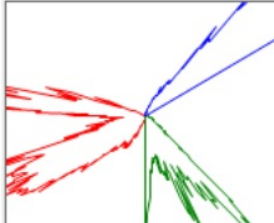

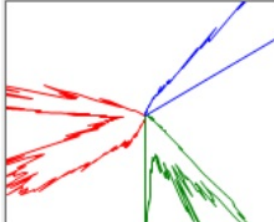
Ekstraksi dengan algoritma *LSB* adalah proses mengekstraksi bit-bit pesan pada citra *stego*. Ekstraksi dilakukan dari sisi penerima pesan, tahapan ekstraksi terdiri dari ekstraksi bit-bit penanda pesan berikutnya hasil *string* biner penanda pesan dikonversi ke *Byte* kemudian *Byte* dikonversi ke *String*. Proses selanjutnya melakukan pengecekan penanda pesan. Jika penanda pesan bernilai sama dengan hasil ekstraksi penanda pesan pada citra *stego* maka ekstraksi bit-bit ukuran *Byte* pesan dengan cara yang sama dengan sebelumnya. Hasil *string* biner panjang pesan selanjutnya dikonversi ke *Byte*, selanjutnya *Byte* panjang pesan dikonversi ke *integer*. Panjang pesan berfungsi sebagai batasan dalam ekstraksi bit-bit pesan. Setelah kedua proses ekstraksi dikerjakan selanjutnya ekstraksi bit-bit pesan dengan cara konversi setiap nilai *Byte* 254 atau 255 citra *stego* sebanyak ukuran *Byte* pesan. Hasil ekstraksi bit-bit pesan dikonversi ke *Byte*. *Byte* pesan yang dihasilkan adalah *Byte cipher* untuk kemudian didekripsi menggunakan algoritma *Rijndael* dengan kunci simetris 128 bit untuk menghasilkan *plaintext*. Gambar 3 menunjukkan proses pengambilan pesan dengan metode *LSB*.


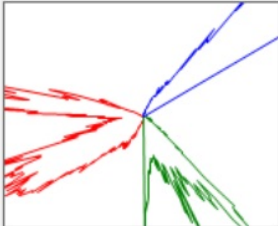

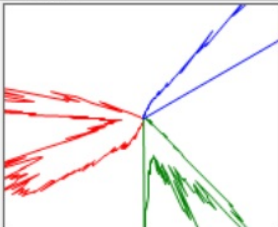

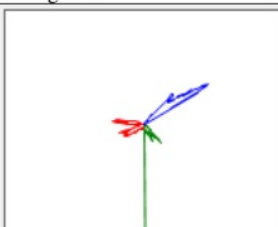

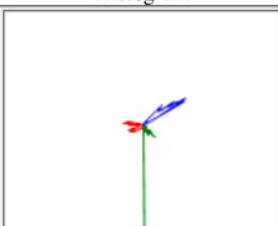


Gambar 3. Proses pengambilan pesan dengan metode LSB

Pengujian dilakukan untuk mengukur kinerja algoritma yang telah diimplementasikan. Pengujian dilakukan dengan dua cara, yaitu kualitatif dan kuantitatif. Pengujian kualitatif bertujuan untuk mengamati perubahan kualitas frame video *stego* dari bentuk citra asli berdasarkan pengamatan menggunakan visual manusia. Pengamatan bersifat subyektif untuk mendeteksi perubahan kualitas citra dan histogram citra yang terjadi pada frame video *stego*. Hasil dari pengujian kualitatif dapat dilihat pada Tabel 1. Semakin besar ukuran Byte pesan yang disisipkan maka histogram frame video semakin berubah.

Tabel 1. Hasil pengujian kualitatif dengan berbagai macam panjang pesan pada video resolusi 640x360

No	Citra Asli	Histogram
1		
	Byte Pesan 1085	Histogram
		
	Byte Pesan 4836	Histogram

2		
	Byte Pesan 8718	Histogram
3		
	Byte Pesan 34195	Histogram
4		
	Byte Pesan 49879	Histogram
5		

Pengujian kuantitatif dilakukan dengan melakukan pengujian pada enam video dengan resolusi yang berbeda, sedangkan yang disisipkan ada lima pesan dengan panjang Byte yang bervariasi. Hasil pengujian kuantitatif dengan ditunjukkan pada Tabel 2.

Tabel 2. Hasil pengujian kuantitatif pada enam video dan lima pesan dengan ukuran Byte yang berbeda

No	Resolusi Video	Panjang Pesan (Byte)	MSE	PNSR	Waktu Penyisipan	Waktu Extraksi	Proses
1	360x240	1085	0,02	143,53	00:00:01:500	00:00:01:460	Berhasil
		4836	0,10	129,62	00:00:03:300	00:00:10:100	Berhasil
		8718	0,18	123,43	00:00:04:900	00:00:23:580	Berhasil
		34195					Gagal
		49879					Gagal
2	420x240	1085	0,01	147,18	00:00:01:590	00:00:01:460	Berhasil
		4836	0,06	132,10	00:00:03:800	00:00:09:870	Berhasil
		8718	0,11	126,51	00:00:05:610	00:00:23:720	Berhasil

		34195	0,41	113,42	00:00:17:210	00:03:55:360	Berhasil
		49879	0,50	111,24	00:00:24:330	Gagal	Gagal
3	640x360	1085	0,01	150,84	00:00:02:330	00:00:01:460	Berhasil
		4836	0,04	137,32	00:00:04:600	00:00:09:810	Berhasil
		8718	0,06	131,44	00:00:06:550	00:00:23:590	Berhasil
		34195	0,25	117,66	00:00:18:500	00:03:57:100	Berhasil
		49879	0,37	113,73	00:00:25:300	00:08:21:280	Berhasil
4	854x480	1085	0	165,86	00:00:03:900	00:00:01:460	Berhasil
		4836	0,02	150,43	00:00:06:530	00:00:10:700	Berhasil
		8718	0,03	144,45	00:00:08:190	00:00:23:520	Berhasil
		34195	0,13	130,01	00:00:20:190	00:03:56:190	Berhasil
		49879	0,20	125,78	00:00:27:670	00:08:20:270	Berhasil
5	1280x720	1085	0	167,36	00:00:10:180	00:00:01:470	Berhasil
		4836	0,01	153,01	00:00:12:330	00:00:09:840	Berhasil
		8718	0,02	147,14	00:00:14:120	00:00:23:730	Berhasil
		34195	0,06	134,86	00:00:25:850	00:03:57:260	Berhasil
		49879	0,09	131,07	00:00:33:290	00:08:15:580	Berhasil
6	1920x1080	1085	0	176,62	00:00:23:170	00:00:01:490	Berhasil
		4836	0	162,75	00:00:24:600	00:00:09:820	Berhasil
		8718	0,01	156,51	00:00:26:810	00:00:23:210	Berhasil
		34195	0,03	142,30	00:00:38:600	00:03:57:630	Berhasil
		49879	0,04	139,00	00:00:46:000	00:08:14:312	Berhasil

Penyisipan pesan pada video dengan resolusi 360x240 terjadi kegagalan proses penyisipan pesan untuk pesan yang berukuran 34195 Byte dan pesan yang berukuran 49879 Byte. Penyisipan pesan untuk video beresolusi 420x240 proses penyisipan berhasil tetapi pada saat proses ekstraksi pesan gagal.

Kesimpulan

Hasil pengujian sistem menunjukkan bahwa metode LSB dan Rijndael tidak bisa digunakan untuk penyisipan pesan yang ukuran Bytenya lebih besar dari daya tampung video cover. Perubahan kualitas citra terjadi apabila ukuran Byte pesan yang disisipkan semakin besar.

Daftar Pustaka

- [1] Khushbu Sahul dan Utkarsh Sharm, Video Steganography: An Approach to Hide Text and Image Data Using Sequential Coding. *International Journal of Advance Engineering and Research Development* Volume 2, Issue 5, May -2015
- [2] Atoum, M. S., Ibrahim, S., Sulong, G. dan M-Ahmad, A., (2012). MP3 Steganography: Review, *International Journal of Computer Science Issues*, Vol. 9, Issue 6, No 3.
- [3] K. Steffy Jenifer et al, International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 5 (1) , 2018, 319-322
- [4] M. Pachhaammal at al, Data Hiding in Multiple Frames Using Base64 Encoding, *Australian Journal Basic and Applied Sciences*, 9(10) Special 2015, Pages: 10-14
- [5] Cachin C., 2005, A Survey Prepared for the Encyclopedia of Cryptography and Security, *Digital Steganography*, IBM Research Zurich Research Laboratory, Switzerland.
- [6] Hmood, N.D., 2012, A Random Key Generation Approach for Rijndael algorithm, *Journal of Al-Nahrain University*, Vol 15 (3), pp. 190-195
- [7] Maninder Pal Singh dan Harmandeep Singh, An Efficient Modified LSB Technique for Video Steganography, *International Journal of Advanced Research in Electrical, Electronics, and Instrumental Engineering*, Vol. 4, Issue 6, June 2015
- [8] Nur Rohman dan Juwita, Deteksi Steganografi Berbasis Least Significant Bit (LSB) dengan Menggunakan Analisis Statistik, *IJC* Vol. 5 No. 1, Jan, 2011.
- [9] Dedy Abdullah dan Doni Nugroho Saputro, Implementasi Algoritma Blowfish dan Metode Least Significant Bit Insertion pada Video Mp4, *Jurnal Pseudocode*, Volume III Nomor 2, 2016.
- [10] Wahyu Lukman (2016). *Prototyping Model*, diambil dari <https://www.scribd.com/doc/58298607/Pengertian-Prototype>.

Steganografi Video Digital dengan Algoritma LSB (Least Significant Bit) dan Rijndael

ORIGINALITY REPORT

29%

SIMILARITY INDEX

PRIMARY SOURCES

1	www.scribd.com Internet	378 words — 14%
2	repository.usu.ac.id Internet	67 words — 2%
3	ejournal.unib.ac.id Internet	51 words — 2%
4	adonaradavid37.blogspot.com Internet	34 words — 1%
5	Al-Ayed, Fadi. "Real-Time Adaptive Intrusion Detection to Secure File Transfer Sessions.", The Catholic University of America, 2017 ProQuest	33 words — 1%
6	ijarcs.info Internet	29 words — 1%
7	Cherry, Denny. "Database Encryption", Securing SQL Server, 2013. Crossref	25 words — 1%
8	ajbasweb.com Internet	22 words — 1%
9	ejurnal.its.ac.id Internet	17 words — 1%
10	www.neliti.com Internet	

15 words — 1%

11 ijcsits.org
Internet

15 words — 1%

12 www.ijert.org
Internet

14 words — 1%

13 Rahman, Mohammad Sad. "An authentication
middleware for prevention of information theft (AMPIT)",
Proquest, 2013.
ProQuest

14 words — 1%

14 repositori.uin-alauddin.ac.id
Internet

13 words — < 1%

15 id.123dok.com
Internet

12 words — < 1%

16 docplayer.info
Internet

11 words — < 1%

17 repository.uinjkt.ac.id
Internet

10 words — < 1%

18 eprints.utm.my
Internet

9 words — < 1%

19 journal.ugm.ac.id
Internet

7 words — < 1%

EXCLUDE QUOTES ON
EXCLUDE BIBLIOGRAPHY ON

EXCLUDE MATCHES < 1%