PENYISIPAN *CIPHERTEXT* ALGORITMA *GOVERNMENT STANDARD* PADA CITRA KE DALAM AUDIO DENGAN ALGORITMA *LEAST SIGNIFICANT BIT*

SKRIPSI

Oleh

Heri Purwantoro
NIM 131810101035

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2018

PENYISIPAN *CIPHERTEXT* ALGORITMA *GOVERNMENT STANDARD* PADA CITRA KE DALAM AUDIO DENGAN ALGORITMA *LEAST SIGNIFICANT BIT*

SKRIPSI

diajukan guna memenuhi tugas akhir dan memenuhi salah satu syarat untuk
menyelesaikan Program Studi Matematika (S1)
dan mencapai gelar Sarjana Sains

Oleh

Heri Purwantoro
NIM 131810101035

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2018

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. kedua orang tua yang selalu mendampingi dan mendukung serta memberi doa.
2. keluarga besar yang selalu memberi semangat.
3. teman-teman ATLAS dan SMOKERS yang terus memberi bantuan.
4. almamater tercinta Jurusan Matematika FMIPA Universitas Jember.

MOTTO

“Nikmatilah setiap detik hidupmu dan jangan pernah menoleh ke belakang **”**

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

nama : Heri Purwantoro

nim : 131810101035

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "Penyisipan *Ciphertext* Algoritma *Government Standard* pada Citra ke dalam *Audio* dengan Algoritma *Least Significant Bit*" adalah benar-benar hasil karya sendiri, kecuali kutipan yang telah saya sebutkan sumbernya, belum pernah diajukan di institusi manapun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, Januari 2018
Yang Menyatakan,

Heri Purwantoro
NIM. 131810101035

SKRIPSI

PENYISIPAN *CIPHERTEXT* ALGORITMA *GOVERNMENT STANDARD* PADA CITRA KE DALAM AUDIO DENGAN ALGORITMA *LEAST SIGNIFICANT BIT*

Oleh
Heri Purwantoro
NIM 131810101035

Pembimbing:

Dosen Pembimbing 1 : Abduh Riski, S.Si., M.Si.
Dosen Pembimbing 2 : Ahmad Kamsyakawuni, S.Si., M.Kom.

PENGESAHAN

Skripsi berjudul "Penyisipan *Ciphertext* Algoritma *Government Standard* pada Citra ke dalam *Audio* dengan Algoritma *Least Significant Bit*" telah diuji dan disahkan pada:

hari, tanggal :
tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Tim Penguji:

Ketua,	Sekretaris,
Abduh Riski, S.Si., M.Si. NIP. 199004062015041001	Ahmad Kamsyakawuni, S.Si., M.Kom. NIP. 197211291998021001
Penguji I,	Penguji II,
Dr. Firdaus Ubaidillah, S.Si., M.Si. NIP. 197006061998031003	Kosala Dwidja Purnomo, S.Si., M.Si. NIP. 196908281998021001

Mengesahkan
Dekan,

Drs. Sujito, Ph.D.
NIP. 196102041987111001

RINGKASAN

Penyisipan *Ciphertext* Algoritma *Government Standard* pada Citra ke dalam *Audio* dengan Algoritma *Least Significant Bit*; Heri Purwantoro, 131810101035; 2018; 79 halaman; Jurusan Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Perkembangan ilmu pengetahuan dan teknologi mempunyai dampak yang besar dalam persebaran pesan dan informasi baik bersifat umum maupun rahasia. Banyak orang yang menyalahgunakan perkembangan tersebut demi keuntungan diri sendiri dengan cara merugikan orang lain. Pengamanan pesan maupun informasi dilakukan agar pesan atau informasi tersebut sampai kepada penerima dengan aman.

Kriptografi adalah teknik pengamanan pesan dengan cara merubah pesan menjadi sandi yang sulit dimengerti orang lain. Streganografi meruakan ilmu yang bertujuan mengamankan pesan dengan cara menyembunyikan pesan ke dalam bentuk lain. Pengamanan pesan menggunakan kriptografi dilanjutkan dengan steganografi bertujuan agar pesan yang dikirim lebih terjaga kerahasiaannya. Algortima yang digunakan untuk kriptografi adalah algoritma *Government Standard* (GOST). Algoritma yang dikembangkan oleh pemerintahan *Uni Soviet* ini, mempunyai proses enkripsi dan dekripsi yang panjang serta rumit, sehingga pesan yang dienkripsi menggunakan algoritma ini sulit untuk dipecahkan. Algoritma *Least Significant Bit* (LSB) digunakan untuk menyembunyikan sandi hasil enkripsi algoritma GOST pada citra ke dalam *audio* dengan tidak merubah bentuk awal citra dan *audio*. Hal ini dikarenakan algoritma LSB menyembunyikan bit pesan ke dalam bit terakhir pada media *cover*, sehingga perubahan hanya terjadi sebesar ukuran pesan yang disembunyikan.

Data yang digunakan berupa pesan 8 dan 40 karakter, citra *Grayscale* dan RGB mempunyai ukuran 256×256 dan 640×452 *pixel* dengan format PNG dan BMP serta audio BTH dan FDD berformat WAV. Pesan (*plaintext*) dienkripsi menggunakan algoritma GOST menghasilkan *ciphertext* kemudian disisipkan pada

citra menggunakan algoritma LSB. Citra hasil penyisipan *ciphertext* disisipkan ke dalam *audio* menggunakan algoritma LSB sehingga menghasilkan *audio* baru.

Analisis keamanan digunakan untuk mengetahui tingkat keamanan dari algoritma yang diajukan. Analisis keamanan yang digunakan yaitu analisis diferensial NPCR dan UACI. Analisis NPCR digunakan untuk menghitung persamaan dari *cover* sebelum dan sesuah penyisipan pesan, sementara UACI digunakan untuk menghitung rata-rata perubahan intensitas yang terjadi pada *cover* setelah penyisipan. Nilai terkecil NPCR pada citra 99,9125% sementara nilai terbesar UACI pada citra 22,3083% hal ini masih berada pada batas maksimal NPCR dan UACI. Nilai NPCR dan UACI pada *audio* bervariasi, bergantung pada ukuran dan jumlah kanal citra yang disisipkan ke dalam audio. Nilai terkecil NPCR pada *audio* 68,6462% dan nilai terbesar UACI pada *audio* 24,5638%. Hasil analisis keamanan dapat disimpulkan bahwa, persentase keamanan ditentukan dari ukuran pesan yang disisipkan dan ukuran media *cover* penyisipan. Semakin besar pesan yang disisipkan, maka media *cover* yang digunakan harus jauh lebih besar, agar presentase keamanan menjadi lebih tinggi. Apabila pesan yang disisipkan lebih besar dari media *cover*, maka persentase keamaan menjadi lebih rendah.

PRAKATA

Puji syukur kehadirat Allah SWT atas segala limpahan rahmat serta hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir dengan judul "Penyisipan *Ciphertext* Algoritma *Government Standard* pada Citra ke dalam *Audio* dengan Algoritma *Least Significant Bit*". Tugas akhir ini disusun untuk memenuhi salah satu syarat guna menyelesaikan pendidikan strata satu (S1) Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Pada kesempatan ini penulis mengucapkan terima kasih atas bantuan dan bimbingan dalam penyusunan tugas akhir ini, terutama kepada yang terhormat:

1. Drs. Sujito, Ph.D., selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
2. Kusbudiono, S.Si., M.Si., selaku Ketua Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
3. Abduh Riski, S.Si., M.Si. selaku Dosen Pembimbing Utama dan Ahmad Kamsyakawuni, S.Si., M.Kom. selaku Dosen Pembimbing Anggota;
4. Dr. Firdaus Ubaidillah, S.Si., M.Si. selaku Dosen Penguji I dan Kosala Dwidja Purnomo, S.Si., M.Si. selaku Dosen Penguji II;
5. Kedua orang tua serta keluarga besar yang terus memberi dukungan serta doa;
6. Dosen dan karyawan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
7. Teman-teman MOMOKA 2G serta Harly Setyawan yang selalu memberi semangat.

Penulis juga menerima kritik dan saran dari berbagai pihak untuk kesempurnaan penyusunan tugas akhir ini. Penulis berharap agar tugas akhir ini bisa bermanfaat.

Jember, Januari 2018 Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Semakin pesatnya perkembangan teknologi, memicu persebaran informasi sangat besar dan cepat sehingga manusia dituntut untuk menemukan cara yang efektif dan efisien dalam penyampaian pesan. Pesan, data maupun informasi rahasia tidak akan berguna apabila sudah diketahui oleh pihak yang tidak berkepentingan. Cara penyampaian pesan yang aman dan rahasia diperlukan oleh pengirim agar pesan tersebut sampai kepada penerima dengan aman.

Kriptografi merupakan suatu cara yang digunakan dalam pengamanan data, serta menjaga kerahasiaan dan keaslian data. Kriptografi merubah pesan awal (*plaintext*) dirubah (enkripsi) menggunakan suatu metode dengan kunci (*key*) menjadi sebuah sandi (*ciphertext*), kemudian sandi tersebut dikirim ke penerima dan diterjemahkan (dekripsi) menjadi pesan awal (*plaintext*). Penyandian pesan bertujuan agar kerahasiaan pesan lebih terjaga dan pesan yang dikirim tidak mudah diketahui oleh orang lain. Salah satu algoritma kriptografi yang digunakan untuk mengamankan pesan adalah algoritma GOST. Algoritma GOST merupakan sebuah algoritma kriptografi modern berbasis pada biner dari *plaintext*.

Government Standard (GOST) adalah sebuah algoritma *block cipher* 64 bit dengan panjang kunci 256 bit untuk jumlah putaran 32 kali *round*. Sah (2012) membahas mengenai aplikasi pengamanan data menggunakan metode *Government Standard* (GOST) pada program *microsoft visual basic*. Hasil yang diperoleh, bahwa kunci algoritma GOST relatif sederhana sehingga rentan dalam pemecahan, akan tetapi proses enkripsi serta dekripsi yang panjang dan rumit membantu mengurangi tingkat kehilangan data pada saat pemecahan. Iqbal (2016) membahas pemahaman teknik kriptografi GOST dengan hasil, penjadwalan kunci yang sederhana membuat GOST mudah dipecahkan. Hal ini bisa dihindari dengan menggunakan kombinasi dari percampuran kunci sehingga lebih sulit untuk dipecahkan.

Steganografi adalah suatu ilmu yang meneliti dan mengembangkan metode penyembunyian informasi ke dalam bentuk lain. Pada era *digital*, steganografi

merupakan suatu metode yang digunakan untuk menyembunyikan pesan ke dalam format lain seperti citra, *audio*, *video* dan *file* lain. Penyembunyian pesan dilakukan dengan cara menyisipkan pesan ke dalam bagian-bagian tertentu dari media penyisipan menggunakan metode yang dikehendaki. Salah satu metode yang digunakan dalam penyisipan adalah metode *Least Significant Bit* (LSB). LSB adalah metode penyisipan pesan dengan cara menyisipkan bit pesan ke dalam bit terakhir dari media penyisipan.

Wirawan (2011) telah membahas mengenai implementasi steganografi pada *audio* dengan format wav. Pesan yang disisipkan ke dalam *audio* berupa citra dengan format JPEG/JPG. Penyisipan citra pada *audio* tidak berpengaruh signifikan pada ukuran *audio.* *Audio* yang telah disisipi citra berubah signifikan apabila *audio* tersebut terkena kompresi, manipulasi amplitudo, pemotongan serta penggabungan dengan *audio* lain.

Wahyuningsih (2016) melakukan penelitian dengan hasil, penyisipan pesan berupa *text* yang telah dienkripsi dengan algoritma *Hill Cipher* kemudian di sisipkan ke dalam *audio*, dimana jumlah bit pesan harus sama dengan atau kurang dari jumlah bit *file audio*. Apabila panjang bit *file audio* lebih pendek dari panjang pesan, maka pesan tidak dapat disisipkan ke dalam *audio*. Waktu penyisipan dan ekstraksi membutuhkan waktu yang relatif cepat dengan ukuran *file* kecil dan waktu lebih lama bila *file* yang digunakan berukuran besar. Besar atau kecilnya *file* yang disisipkan pada *audio* berpengaruh terhadap kualitas suara dari *audio* tersebut.

Pada penelitian ini akan dilakukan pengamanan pesan menggunakan algoritma GOST kemudian disisipkan pada citra ke dalam *audio* menggunakan algoritma LSB. Melakukan proses ekstraksi *audio* dan citra serta melakukan proses dekripsi pada *ciphertext* yang didapat dari proses ekstraksi citra guna mendapatkan pesan awal. Aanalisis keamanan menggunakan metode NPCR dan UACI pada citra serta *audio* hasil penyisipan, untuk mengetahui tingkat keamanan algoritma yang digunakan.

1.2 Rumusan Masalah

Adapun rumusan masalah yang dibahas dalam penelitian ini meliputi:

a. Bagaimana proses enkripsi dan dekripsi menggunakan algoritma *Government Standard* (GOST) serta proses penyisipan dan ekstraksi menggunakan algoritma *Least Significant Bit* (LSB)?
b. Bagaimana hasil analisis keamanan dari metode yang digunakan?

1.3 Batasan Masalah

Adapun batasan masalah dari penelitian ini adalah:

a. Media penyisipan berupa *file* citra dengan format PNG dan BMP serta *file audio* dengan format WAV.
b. Karakter ASCII *Printable Characters*.

1.4 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah:

a. Mengetahui proses enkripsi dan dekripsi menggunakan algoritma *Government Standard* (GOST) serta proses penyisipan dan ekstraksi menggunakan algoritma *Least Significant Bit* (LSB).
b. Mengetahui hasil analisis dari algoritma yang digunakan.

1.5 Manfaat Penelitian

Adapun manfaat yang diperoleh dari penelitian ini antara lain:

a. Mendapat pengetahuan baru mengenai metode yang digunakan untuk mengamankan informasi pada penelitian ini.
b. Mengetahui hasil analisis keamanan dari metode yang digunakan.

BAB 2. TINJAUAN PUSTAKA

2.1 Kriptografi

Kriptografi (*cryptography*) berasal dari bahasa Yunani, yang terdiri dari kata *cryptos* dengan arti rahasia, dan *graphein* yang berarti tulisan. Sehingga kriptografi mempunyai arti sebagai tulisan rahasia. Definisi yang digunakan untuk menyatakan kriptografi, sebagai sebuah ilmu yang bertujuan menjaga kerahasiaan pesan dengan cara merubah pesan awal ke dalam bentuk sandi sehingga sulit dimengerti oleh orang lain.

Kriptografi tidak hanya memberikan keamanan dan kerahasiaan dari informasi yang disampaikan, kriptografi lebih pada teknik dalam penyembunyian informasi itu sendiri. Adapun empat tujuan dari kriptografi yaitu (Menez *et al*., 1996):

a. Kerahasiaan Data

   Pengiriman pesan memerlukan sebuah proses yang digunakan untuk menjaga keamanan isi pesan dari orang yang tidak berkepentingan.

b. Integritas Data

   Integritas data berhubungan dengan penjagaan data dari perubahan secara tidak sah. Untuk menjaga integritas data, sistem yang digunakan harus mampu tahan dari manipulasi pihak lain. Manipulasi data terjadi dalam bentuk penyisipan, penghapusan, pengurangan serta penambahan data lain ke dalam data awal.

c. Autentifikasi Data

   Autentifikasi data berhubungan dengan identifikasi baik secara kesatuan sistem maupun pesan itu sendiri. Informasi yang dikirimkan harus diautentifikasi keaslian, isi data, waktu pengiriman pesan dan lain-lain.

d. Non-repudiasi Data

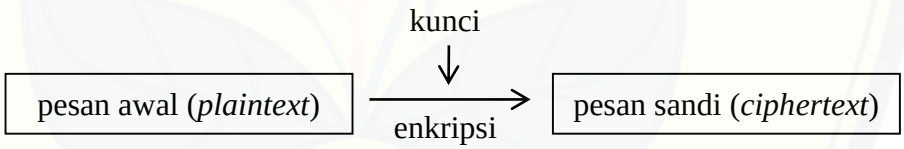
   Non-repudiasi data adalah usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman pesan oleh pihak pengirim.

Berdasarkan perkembangan zaman, kriptografi dibagi menjadi dua jenis, yaitu kriptografi klasik dan kriptografi modern. Kriptografi klasik menggunakan kunci simetris serta menggunakan metode subtitusi dan tranposisi. Metode subtitusi dan tranposisi bekerja pada *plaintext* dengan merubah karakter awal menjadi karakter

baru berbentuk *ciphertext.* Kriptografi modern bekerja pada biner dari *plaintext* dengan menggunakan algoritma yang membentuk *ciphertext.* Operasi yang digunakan mengikuti operasi pada komputer.

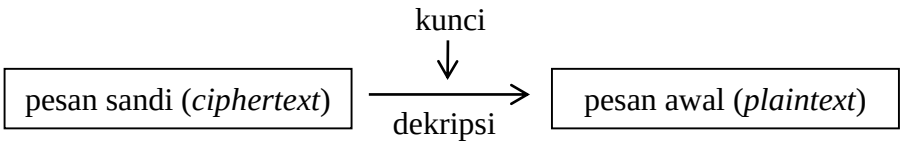
Pada dasarnya kriptografi mempunyai dua langkah utama, yaitu proses enkripsi dan proses dekripsi. Enkripsi adalah proses perubahan pesan awal (*plaintext*) menjadi sandi (*ciphertext*), sedangkan dekripsi adalah proses mengembalikan sandi (*ciphertext*) menjadi bentuk pesan awal (*plaintext*). Proses enkripsi maupun dekripsi memerlukan sebuah kunci untuk merubah pesan awal dan sebaliknya. Kunci yang digunakan haruslah rahasia, hanya pengirim dan penerima pesan yang mengetahui kunci tersebut supaya isi pesan tetap rahasia.

Enkripsi adalah proses perubahan pesan awal (*plaintext*) menjadi runtutan karakter yang tidak mempunyai arti dan mempunyai urutan bit yang tidak beraturan atau sandi (*ciphertext*). Perubahan *plaintext* menjadi *ciphertext* dilakukan dengan menggunakan metode khusus yang diinginkan, serta penggunaan kunci rahasia dengan tujuan agar pesan yang dikirim lebih terjaga kerahasiaannya. Proses enkripsi ditunjukkan pada Gambar 2.1.



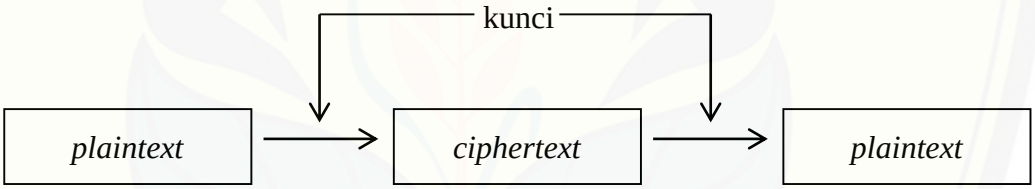
Gambar 2.1 Model sederhana proses enkripsi

Dekripsi adalah suatu proses kebalikan dari enkripsi, merubah *ciphertext* menjadi *plaintext.* Proses dekripsi menggunakan metode dan kunci yang sama pada saat proses enkripsi. Penggunaan metode dan kunci yang sama dimaksudkan agar *plaintext* yang didapatkan dari proses dekripsi *ciphertext* sama dengan *plaintext* awal sebelum enkripsi. Proses dekripsi ditunjukkan pada Gambar 2.2.



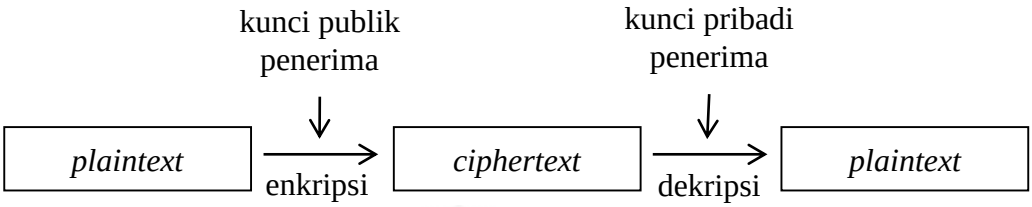
Gambar 2.2 Model sederhana proses dekripsi

Kriptografi menggunakan kunci dalam perubahan *plaintext* menjadi *ciphertext* ataupun sebaliknya. Berdasarkan jumlah kunci, kriptografi dibagi menjadi dua, yaitu sistem algoritma simetris dan sistem algoritma asimetris. Sistem algoritma simetris adalah algoritma dengan kunci tunggal, dimana pengirim dan penerima pesan harus mempunyai kunci yang sama untuk proses enkripsi dan proses dekripsi. Sistem algoritma simetri mempunyai waktu relatif lebih cepat untuk proses enkripsi dan dekripsi, karena kunci yang digunakan sama (Munir, 2006). Algoritma simetris ditunjukkan pada Gambar 2.3.



Gambar 2.3 Model sederhana sistem algoritma simetris

Sistem algoritma asimetris merupakan algoritma dengan kunci yang berbeda untuk proses enkripsi dan dekripsi. Algoritma ini disebut juga sebagai algoritma kunci umum (*public key algorithm*) karena proses enkripsi menggunakan kunci umum (*public key*), kunci umum dapat diketahui oleh orang lain. Berbeda dengan proses enkripsi, kunci untuk proses dekripsinya hanya diketahui oleh penerima pesan atau disebut kunci pribadi (*private key*). Sistem algoritma asimetris ditunjukkan pada Gambar 2.4.



Gambar 2.4 Model sederhana sistem algoritma asimetris

2.2 Steganografi

Steganografi berasal dari bahasa Yunani yang terdiri dari kata *steganos* yang berarti tertutup dan *graphein* yang berarti tulisan. Steganografi adalah sebuah seni penyembunyian pesan ke dalam media lain dengan tujuan agar orang lain tidak menyadari adanya pesan di dalam media tersebut. Lebih mudahnya *steganos* dan *graphein* mempunyai arti tulisan yang tersembunyi (Santoso, 2014).

Berbeda dengan kriptografi, steganografi merupakan proses pengamanan pesan dengan melakukan penyisipan pesan ke dalam media penampung. Kriptografi merubah pesan awal menjadi sandi dengan bentuk yang tidak beraturan, sedangkan steganografi menyisipkan pesan ke dalam media lain tanpa merubah bentuk awal media penampung. Penyisipan dilakukan dengan mengganti beberapa bagian dari media penampung, sehingga bentuk awal tidak banyak berubah. Pada steganografi terdapat kriteria yang digunakan sebagai acuan untuk mengetahui keamanan penyisipan data. Berikut adalah kriteria-kriteria dari steganografi (Munir, 2004).

a. *Fidelity*

   Kualitas media penampung yang telah disisipi pesan tidak boleh berbeda jauh dari kualitas aslinya. Perubahan yang terjadi tidak boleh diketahui oleh indra manusia.

b. *Robustness*

   Data yang disembunyikan harus tahan terhadap perubahan yang dilakukan teradap media penampung.

c. *Recovery*

Data yang telah disisipkan pada media penampung harus dapat diambil kembali tanpa merusak kualitas data awal.

Steganografi membutuhkan dua properti, yaitu media penampung (*cover*) dan pesan rahasia. Media penampung yang umum digunakan dalam bentuk gambar, *audio* ataupun video. Sedangkan pesan yang disembunyikan dapat berupa artikel, gambar, video, *audio*, kode program atau pesan yang lain. Dalam steganografi terdapat dua proses yaitu, proses penyisipan (*insertion*) dan ekstraksi (*extraction*).

Insertion merupakan proses penyisipan pesan ke dalam media penampung dalam bentuk lain atau disebut media *cover*. *Extraction* adalah proses penguraian pesan yang telah disembunyikan pada media *cover* kembali ke bentuk awal pesan. Skema proses *insertion* dan *extraction* pada steganografi ditunjukkan pada Gambar 2.5.

Gambar 2.5 Proses steganografi

2.3 *Government Standard* (GOST)

GOST merupakan singkatan dari *Gosudarstvennyi Standard* atau *Government Standard*, merupakan suatu algoritma *block cipher* yang dikembangkan oleh warga kebangsaan Uni Soviet. Algoritma GOST dikembangkan oleh pemerintah Uni Soviet pada masa perang dingin tahun 1970, digunakan untuk menyembunyikan data maupun informasi yang bersifat rahasia. Algoritma GOST merupakan sebuah algoritma enkripsi dengan proses 32 *round* (putaran) yang menggunakan 64 bit *block cipher* dan 256 bit *key* (Munir, 2006).

GOST menggunakan 8 buah *S-Box* yang berbeda dan menggunakan operasi *XOR*. GOST menggunakan kunci 256 bit dengan rincian $k_1, k_2, k_3, \dots, k_{256}$. Kunci tersebut dimasukkan ke dalam 8 kelompok yaitu, $K_1, K_2, K_3, K_4, K_5, K_6, K_7,$ dan K_8.

$K_1 = (k_{32}, ..., k_1)$

$K_2 = (k_{64}, ..., k_{33})$

$K_3 = (k_{96}, ..., k_{65})$

$K_4 = (k_{128}, ..., k_{97})$

$K_5 = (k_{160}, ..., k_{129})$

$K_6 = (k_{192}, ..., k_{161})$

$K_7 = (k_{224}, ..., k_{193})$

$K_8 = (k_{256}, ..., k_{225})$

Terdapat 8 buah blok kunci, K_1 sampai dengan K_8. Karena terdapat 32 putaran, maka penggunaan kunci dilakukan penjadwalan (Herryawan, 2010). Penjadwalan kunci enkripsi dan dekripsi ditunjukkan pada Tabel 2.1 dan Tabel 2.2.

Tabel 2.1 Penjadwalan kunci enkripsi

Putaran	1	2	3	4	5	6	7	8
Kunci	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8
Putaran	9	10	11	12	13	14	15	16
Kunci	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8
Putaran	17	18	19	20	21	22	23	24
Kunci	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8
Putaran	25	26	27	28	29	30	31	32
Kunci	K_8	K_7	K_6	K_5	K_4	K_3	K_2	K_1

Tabel 2.2 Penjadwalan kunci dekripsi

Putaran	1	2	3	4	5	6	7	8
Kunci	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8
Putaran	9	10	11	12	13	14	15	16
Kunci	K_8	K_7	K_6	K_5	K_4	K_3	K_2	K_1
Putaran	17	18	19	20	21	22	23	24
Kunci	K_8	K_7	K_6	K_5	K_4	K_3	K_2	K_1
Putaran	25	26	27	28	29	30	31	32
Kunci	K_8	K_7	K_6	K_5	K_4	K_3	K_2	K_1

Tabel *S-Box* dibuat pemerintah Rusia dan digunakan oleh Federasi Bank Sentral Rusia (*Central Bank of Rusian Federation*). Tabel *S-Box* dapat dilihat pada Tabel 2.3.

Tabel 2.3 *S-Box*

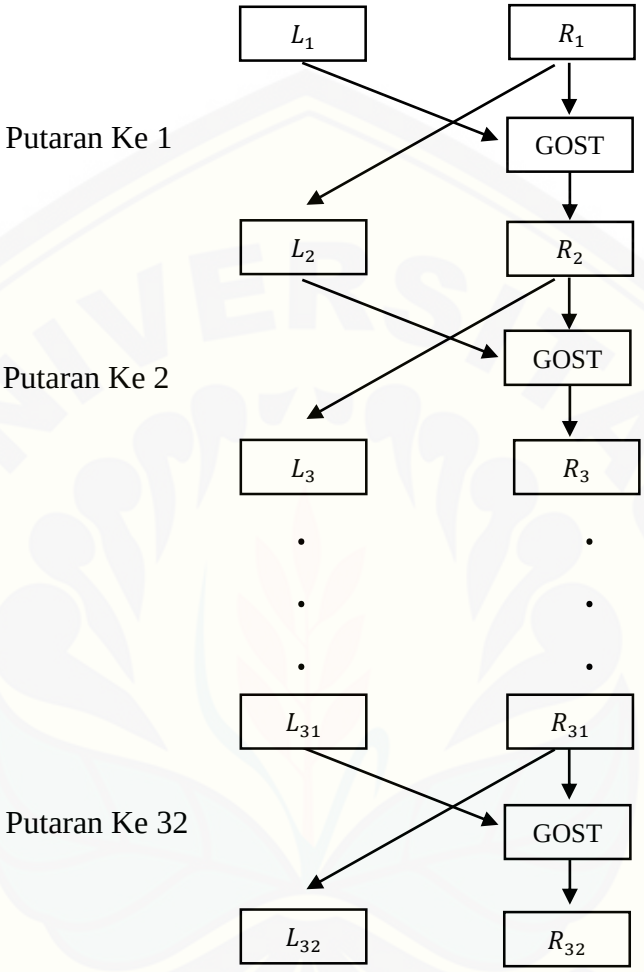
S-Box	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3
2	14	11	4	12	6	13	15	10	2	3	8	1	0	7	5	9
3	5	8	1	13	10	3	4	2	14	15	12	7	6	0	9	11
4	7	13	10	1	0	8	9	15	14	4	6	12	11	2	5	3
5	6	12	7	1	5	15	13	8	4	10	9	14	0	3	11	2
6	4	11	10	0	7	2	1	1	3	6	8	5	9	12	14	14
7	13	11	4	1	3	15	5	9	0	10	14	14	6	8	2	12
8	1	15	13	0	5	7	10	4	9	2	3	7	6	11	8	12

2.3.1 Proses Enkripsi

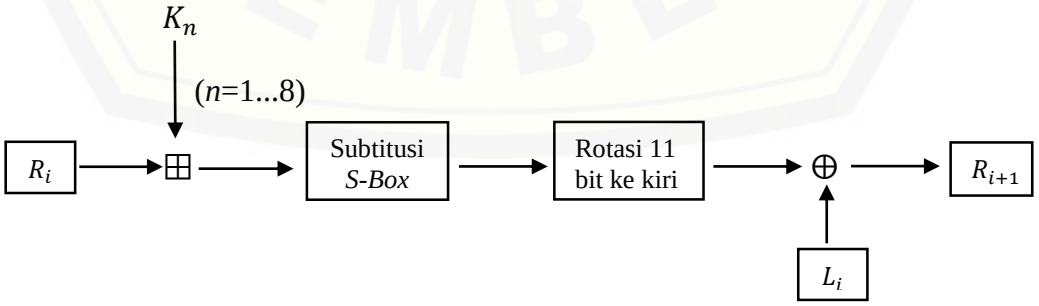
Proses enkripsi dari algoritma GOST untuk satu *round* (putaran) dijabarkan sebagai berikut.

a. 64 bit *plaintext* dibagi menjadi dua buah bagian, L_1 dan R_1 masing-masing bagian terdapat 32 bit.
b. $(R_1 + K_n)$ mod 2^{32} dimana K_n adalah kunci ke-*n*. Hasil penjumlahan dan modulo 2^{32} menghasilkan 32 bit.
c. Hasil penjumlahan modulo 2^{32} dibagi menjadi 8 bagian, masing-masing bagian terdiri dari 4 bit. Setiap bagian dimasukkan ke dalam tabel *S-Box* yang berbeda, 4 bit pertama menjadi *input* dari *S-Box* pertama, 4 bit kedua menjadi *input S-Box* kedua dan seterusnya.
d. Setelah mendapatkan hasil subtitusi dari *S-Box,* digabungkan kembali menjadi 32 bit, kemudian dilakukan operasi rotasi *left shift* sebanyak 11 bit.
e. Hasil dari rotasi *left shift* dilakukan operasi $\oplus$ dengan L_1, menghasilkan R_2.
f. Sementara R_1 yang tidak dilakukan operasi apapun adalah hasil dari L_2 .
g. Untuk putaran selanjutnya dilakukan dengan cara yang sama dengan menggunakan R_2 dan L_2 sebagai biner enkripsi.

Enkripsi dari algoritma GOST masing-masing ditunjukkan pada Gambar 2.6 dan Gambar 2.7.



Gambar 2.6 Skema enkripsi algoritma GOST



Gambar 2.7 Proses algoritma GOST

2.3.2 Proses Dekripsi

Proses dekripsi merupakan kebalikan dari proses enkripsi, urutan proses dekripsi tidak berubah dari proses enkripsi. Perbedaan yang terjadi terdapat pada penjadwalan kunci setiap putaran. Penjadwalan kunci setiap putaran untuk proses dekripsi sebagai berikut (Munir, 2006).

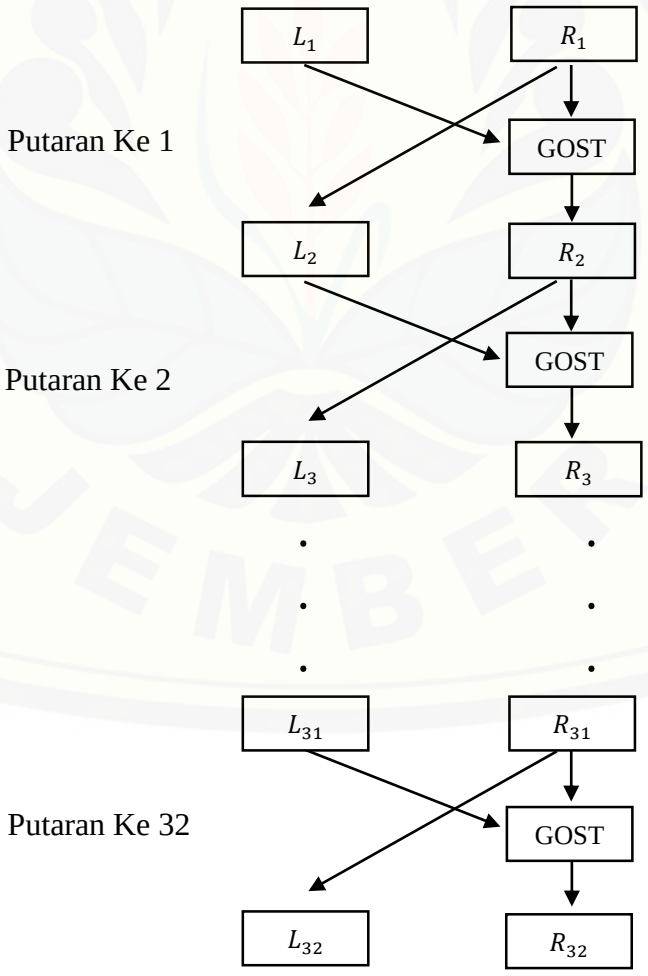
Putaran 1-8 : $K_1, K_2, K_3, \ldots, K_8$

Putaran 9-16 : $K_8, K_7, K_6, \ldots, K_1$

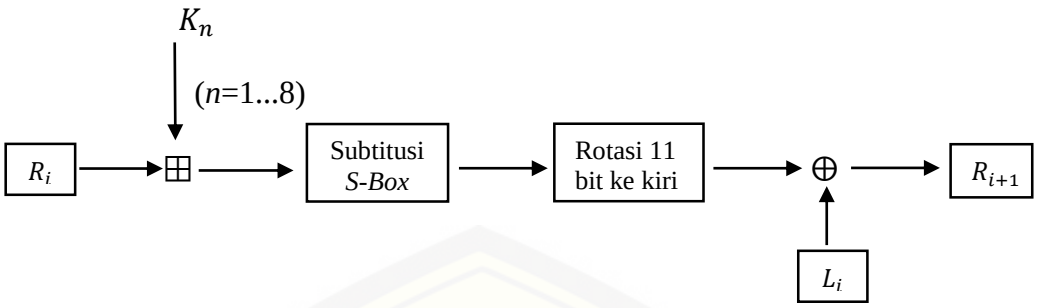
Putaran 17-24 : $K_8, K_7, K_6, \ldots, K_1$

Putaran 25-32 : $K_8, K_7, K_6, \ldots, K_1$

Dekripsi dari algoritma GOST masing-masing ditunjukkan pada Gambar 2.8 dan 2.9.



Gambar 2.8 Skema dekripsi algoritma GOST



Gambar 2.9 Proses algoritma GOST

2.4 *Least Significant Bit* (LSB)

Metode *Least Significant Bit* (LSB) merupakan salah satu metode dalam steganografi yang digunakan untuk menyisipkan pesan atau data ke dalam format yang berbeda. Metode *Least Significant Bit* (LSB) berfungsi mengganti bit terakhir dari media *cover* dengan bit pesan atau data yang akan disisipkan. Pesan atau data yang disisipkan ke dalam media lain harus disesuaikan dengan ukuran *cover* tersebut. Apabila pesan atau data yang disisipkan melebihi ukuran *cover*, maka penyisipan hanya dilakuan sepanjang ukuran *cover*.

Least Significant Bit (LSB) adalah bit yang mempunyai nilai terendah dalam barisan biner. Sedangkan bit yang memiliki nilai tertinggi disebut *Most Significant Bit* (MSB). Pada data atau *file* biasanya terdapat bit yang mempunyai peranannya kurang penting dan dapat diganti dengan informasi lain tanpa merusak *file* tersebut. Karena memanfaatkan bit yang kurang berarti, metode ini tidak digunakan pada media yang mengalami kompresi karena akan menghilangkan bit-bit LSB tersebut. Penggunaan metode LSB secara umum tidak merubah ukuran *file* gambar atau *audio* yang memiliki resolusi tinggi/ bit *rate* tinggi (Wahyudi, 2014).

Proses penyisipan pesan pada *file*, dilakukan terhadap bit LSB *cover* dan bit LSB pesan. Pesan yang disisipkan cenderung mempunyai panjang yang dinamis, panjang pesan disesuaikan dengan *cover* penyisipan. Pada citra *digital* penggantian bit-bit LSB berpengaruh terhadap warna. Penggantian bit LSB akan menyebabkan perubahan sebesar satu angka lebih tinggi ataupun satu angka lebih rendah dari nilai awal. Perubahan ini tidak menimbulkan perbedaan warna yang signifikan, karena perubahan hanya terjadi satu bit di atas atau di bawah bit awal.

Sama seperti penyisipan pesan ke dalam citra, penyisipan pesan ke dalam *audio* menggunakan metode LSB dilakukan dengan mengganti bit LSB pada *audio* dengan bit pesan. Bit pesan yang disisipkan pada bit *audio* dengan tingkat yang lebih rendah atau lebih tinggi dari bit *audio*. Penggantian bit yang kurang signifikan dilakukan setelah menentukan bilangan biner dari pesan dan bilangan biner dari *audio*.

Sebagai contoh metode LSB, penyisipan pesan ke dalam media citra sebagai berikut. Bila terdapat karakter 'A' dengan kode ASCII 65 yang akan disisipkan ke dalam citra dengan nilai setiap *pixel* 30, 18, 210, 90, 89, 172, 50, 190, 100. Karakter 'A' dirubah menjadi bentuk biner dengan nilai 01000001. Sedangkan setiap *pixel* dari citra dirubah dalam bentuk biner.

00011110	00010010	11010010
01011010	01011001	10101100
00110010	10111110	01100100

Setelah pesan dan media *cover* dirubah menjadi biner, nilai biner yang kurang signifikan dari media *cover* yang ditunjukkan dengan bit bergaris bawah diganti dengan biner dari pesan.

Biner pesan:

01000001

Nilai baru berubah menjadi:

00011110	00010011	11010010
01011010	01011000	10101100
00110010	10111111	01100100

Bit yang bergaris bawah adalah bit *pixel* citra yang telah diganti dengan bit pesan. *Pixel* dalam citra berubah menjadi 30, 19, 210, 90, 88, 172, 50, 191, 100. Perubahan yang terjadi tidak terlalu signifikan.

2.5 Kode ASCII

American Standard Code for Information Interchange (ASCII) adalah sebuah kode dalam bentuk karakter yang bersifat umun serta banyak digunakan dalam teknologi, khususnya pada komputer. Kode ASCII terdapat 256 karakter yang

terdiri dari angka, huruf, tanda baca serta simbol. Kode ASCII yang sering digunakan dan terdapat pada *keyboard* komputer adalah kode ASCII *pritntable*. Kode ASCII *printable* dimulai dengan karakter spasi pada kode 32 sampai karakter *Delete* pada kode 127. Kode ASCII terdiri dari angka, huruf tanda baca dan beberapa simbol. Kode ASCII *printable character* terdapat pada Lampiran A.

2.6 Citra

Citra adalah suatu gambaran (representasi), kemiripan atau imitasi dari suatu objek. Secara harfiah citra (*image*) adalah gambar pada bidang dua dimensi. Citra terdiri dari dua sifat yaitu citra analog dan citra *digital*. Citra analog adalah citra yang bersifat *continue* seperti gambar pada monitor televisi, foto *sinar-X* dan lain sebagainya. Citra analog tidak dapat direpresentasikan oleh komputer sehingga tidak dapat langsung diolah oleh komputer, akan tetapi harus melalui proses konversi analog ke *digital* terlebih dahulu. Citra *digital* adalah citra yang dapat diolah langsung oleh komputer secara numerik dan disimpan pada komputer sebagai angka untuk menunjukkan besar intensitas pada setiap *pixel*.

Citra *digital* berukuran N x M dinyatakan dengan matriks yang berukuran N baris dan M kolom sebagai berikut :

$$f(x,y) = \begin{bmatrix} f(1,1) & \cdots & f(1,M) \\ \vdots & \ddots & \vdots \\ f(N,1) & \cdots & f(N,M) \end{bmatrix}$$

Setiap elemen pada citra *digital* disebut dengan *pixel* (*picture element*). Jadi citra yang berukuran N x M mempunyai NM buah *pixel* (Dulimarta, 1997).

Munir (2004) mengelompokkan citra menjadi dua kelompok berdasarkan pergerakannya, yaitu citra diam dan citra bergerak. Citra diam adalah citra tunggal yang tidak dapat bergerak. Citra bergerak adalah rangkaian citra diam yang ditampilkan secara beruntun sehingga memberikan kesan bergerak. Menurut nilai *pixel*nya citra terbagi menjadi tiga jenis (Putra, 2010).

a. Citra Biner

   Citra biner juga disebut dengan citra *B and W* (*Black* dan *White*) atau citra monokrom. Kemungkinan warna yang dimiliki adalah hitam atau putih. Pada waktu penampilan gambar, 0 adalah warna putih dan 1 adalah warna hitam, jadi

citra biner mempunyai latar belakang putih dan objek berwarna hitam. Contoh citra biner dapat dilihat pada Gambar 2.10.

Gambar 2.10 Gambar citra biner

(Sumber: Putra, 2010)

b. Citra *Grayscale*

Citra *grayscale* adalah suatu citra *digital* yang hanya mempunyai satu nilai kanal pada setiap *pixel*nya, karena ditentukan oleh nilai satu intensitas warna. Warna yang terdapat pada citra *grayscale* dapat berupa warna hitam, warna keabuan hingga warna putih. Citra *grayscale* mempunyai *range* 256 artinya skala keabuan dari 0 sampai 255. Intensitas 0 menyatakan hitam, intensitas 255 menyatakan putih dan nilai antara 0 sampai 255 menyatakan warna keabuan yang teletak antara hitam dan putih. Contoh citra *grayscale* dapat dilihat pada Gambar 2.11.

Gambar 2.11 Gambar citra *grayscale*

(Sumber: Putra, 2010)

c. Citra Warna (RGB)

Citra warna dikenal dengan nama citra *spectral* karena tediri dari tiga warna utama yaitu merah (*red*), hijau (*green*) dan biru (*blue*), warna lain dapat diperoleh dengan mencampurkan ketiga warna pokok tersebut. Seperti pada citra *grayscale* citra RGB memiliki *range* 0-255. Setiap *pixel* pada citra warna mempunyai tiga kanal, yaitu kanal merah, hijau dan biru. Contoh citra RGB dapat dilihat pada Gambar 2.12.

Gambar 2.12 Gambar citra RGB

(Sumber: Putra, 2010)

2.7 Audio

Audio adalah sebuah fenomena yang dihasilkan oleh getaran suatu benda, berupa sinyal analog dengan amplitudo yang berubah secara terus menerus terhadap waktu dan frekuensi. Perbedaan tekanan terjadi selama benda bergetar dengan pola osilasi yang dinamakan gelombang. Perulangan gelombang yang sama pada interval tertentu disebut sebagai periode. Format *file audio* adalah sebuah format yang diguakan untuk menyimpan data berupa suara *digital* pada komputer. Data yang tersimpan bisa tanpa kompresi ataupun terkompresi dengan tujuan untuk mengurangi ukuran *file* (Binanto, 2010).

Codec adalah suatu prosess *encoding* dan *decoding* data *audio* mentah, dimana data tersebut disimpan pada format *file* tertentu. Satu jenis format *file audio* hanya mendukung jenis data *audio* tersebut, multimedia *container* format seperti AVI

dapat mendukung beberapa jenis format *audio*. Ada tiga jenis format *file audio* (Wahyuningsih, 2016).

a. Format *audio* yang tidak terkompresi seperti, WAV, AIFF, AU atau *raw header–less* PCM.
b. Format *audio* dengan kompresi *lossless*, seperti FLAC, *Monkey's Audio* (yang berformat APE), WavPck (yang berformat WV), TTA, ATRAC *Advance Lossless, Aple Lossless* (dengan format m4a), MPEG-4 SLS, MPEG-4 ALS, MPEG-4 DTS, *Windows Media Audio Lossless* (WMA *Lossess*), dan *Shorten* (SHN).
c. Format *audio* dengan kompresi *lossy*, seperti MP3, Vobris, Mousepack, AAC, ATRAC dan *Windows Media Lossy* (WMA *Lossy*).

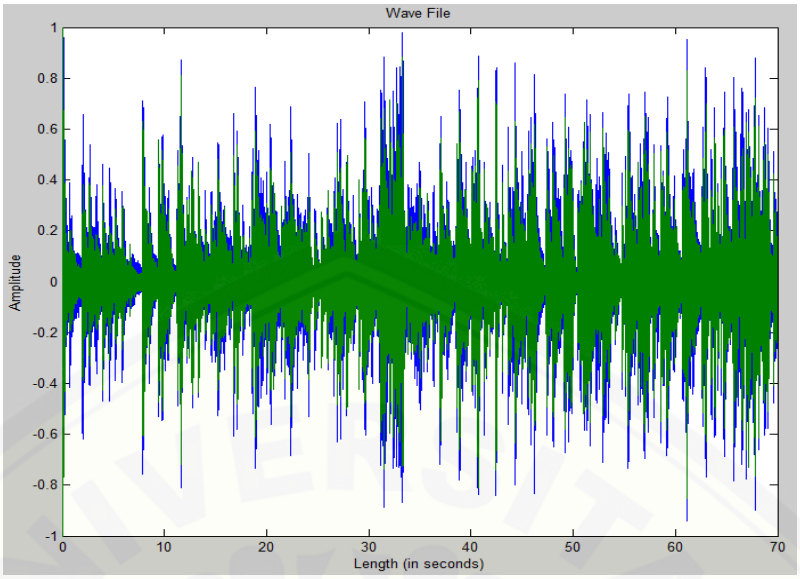
Format *file audio* tanpa kompresi yang paling sering ditemui adalah PCM (*Pulse Code Modulation*), biasanya tersimpan dalam format wav pada Windows dan sebagai aiff dalam Mac.Os. WAV adalah format *file* yang fleksibel untuk menyimpan kombinasi *audio* dengan bit *rates* maupun tanpa bit *rates*. WAV akan mengkoding semua suara, baik suara yang kompleks maupun tanpa suara, dengan jumlah bit yang sama setiap satuan waktunya.

Struktur RIFF (*Resource Interchange File Format*) merupakan struktur yang digunakan dalam Windows. RIFF mengatur data *file* ke dalam bagian-bagian tertentu yang memiliki *header* dan mempunyai ukuran sendiri atau disebut sebagai *chunk*. Data dari suatu bagian bisa memiliki sub-bagian seluruh data dalam file dengan struktur RIFF.

Wahyuningsih (2016) menjelaskan bahwa format *file audio* pada umumnya adalah .wav. *File audio* dapat dikenali menggunakan *software* MATLAB dengan perintah wavread. *File audio* yang akan disisipi pesan dirubah terlebih dahulu menjadi biner dengan langkah-langkah sebagai berikut.

a. Membaca Sinyal *Audio*

MATLAB membaca sinyal *audio* dengan cara menuliskan *syntax* `[y fs]=wavread('nama file'.vaw)` dengan `y` sebagai kolom yang berisi sampel matriks sinyal *audio* dan `fs` adalah *sample rate.* Contoh *sample rate* ditunjukkan pada Gambar 2.13.



Gambar 2.13 Contoh *sample rate*

b. Nilai Integer

Mengambil nilai integer dari sinyal *audio* disesuaikan dengan panjang pesan yang digunakan. Setelah mendapatkan nilai integer dari sinyal *audio*, (nilai `y` yang diperoleh dari proses `wavread`) nilai negatif dirubah menjadi positif dikalikan dengan 128 dilakukan pembulatan ke bawah. Hasil yang diperoleh adalah nilai integer *audio* dalam bentuk desimal.

c. Mengubah Desimal Menjadi Biner

Setelah mendapat nilai desimal dari *audio*, nilai desimal tersebut dirubah menjadi nilai biner sesuai kode ASCII. Proses perubahan nilai desimal menjadi nilai biner pada MATLAB menggunakan perintah `dec2bin`. Setelah mendapatkan nilai biner dari *audio* maka *audio* siap untuk disisipi pesan.

2.8 Analisis Keamanan

Analisis keamanan dilakukan dengan tujuan mengetahui tingkat keamanan dari algoritma yang digunakan pada penelitian ini. Analisis diferensial digunakan untuk mengetahui berapa banyak perubahan yang terjadi pada media penyisipan setelah dilakukan penyisipan pesan. Analisis diferensial digunakan dengan menentukan perbedaan dari citra maupun *audio* sebelum dan sesudah dilakukan penyisipan

pesan. Langkah yang dilakukan dengan menghitung nilai dari *Number of Pixels Change Rate* (*NPCR*) dan *Unified Average Changing Intensity* (*UACI*).

$$NPCR = \frac{\sum_{i,j,k} D(i,j,k)}{W \times H \times C} \times 100\% \quad (2.1)$$

$$UACI = \frac{1}{W \times H \times C}\left[\sum_{i,j,k} \frac{|C(i,j,k) - C'(i,j,k)|}{255}\right] \times 100\% \quad (2.2)$$

W , *H* dan *C* masing-masing adalah leba, tinggi dan kanal citra sementara $D(i,j)$ ditentukan sebagai berikut:

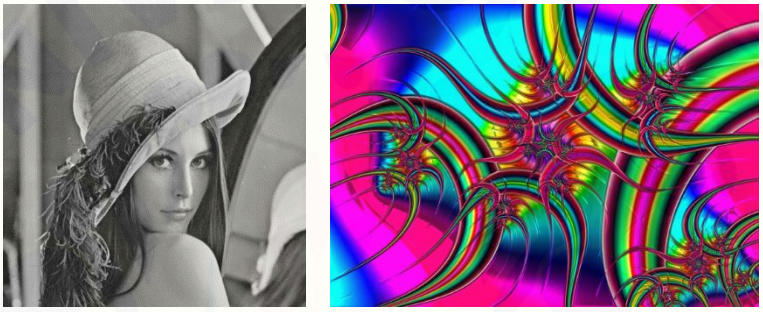
$$D(i,j) = \begin{cases} 1, & C(i,j) = C'(i,j) \\ 0, & C(i,j) \neq C'(i,j) \end{cases}$$

dimana $C(i,j)$ dan $C'(i,j)$ masing-masing adalah nilai dari derajat keabuan baris ke-*i* dan kolom ke-*j* dari citra C dan C′. Kriteria agar algoritma steganografi dikatakan baik, apabila media *cover* setelah penyisipan terjadi perubahan maksimal 10% untuk NPCR, sementara untuk UACI rata-rata perubahan intensitas yang terjadi kurang dari 30% (Yaimin *et al*., 2016). NPCR digunakan untuk menghitung persentase perubahan *pixel* dari seluruh total *pixel* citra, sementara UACI digunakan untuk menghitung persentase rata-rata perubahan intensitas dari *pixel* seluruh yang berubah.

BAB 3. METODE PENELITIAN

3.1 Data Penelitian

Data yang digunakan dalam penelitian ini adalah teks pesan rahasia berupa karakter sesuai pada kode ASCII *printable character* yang telah dienkripsi dengan algoritma GOST. Data yang digunakan sebagai *cover* penyisipan berupa citra *grayscale* (GRY) dan citra warna (RGB) dengan format png serta bmp, masing-masing ditunjukkan pada Gambar 3.1 (a) dan (b). *Cover audio* berupa musik *Beast and The Harlot* (BTH) dan musik *Faded* (FDD) dengan format wav.



(a) (b)

Gambar 3.1 (a) Citra GRY dan (b) Citra RGB

(Sumber: researchgate.net (a) dan cdn.pixabay.com (b))

3.2 Langkah-Langkah Penelitian

Langkah-langkah pada penelitian ini diuraikan secara sistematis sebagai berikut.

a. Studi Literatur

   Tahap ini dilakukan dengan mempelajari teori-teori yang berkaitan dengan penelitian. Teori yang dipelajari adalah teknik kriptografi menggunakan algoritma *Government Standard* (GOST) dan teori steganografi menggunakan algoritma *Least Significant Bit* (LSB) pada citra dan *audio*.

b. Analisis Data

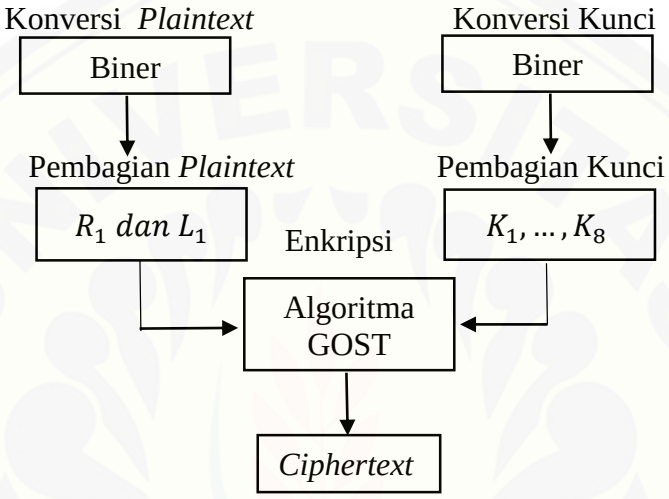
   1. Proses enkripsi dan penyisipan pesan

      Proses enkripsi pesan menggunakan algoritma GOST dan proses penyisipan *ciphertext* pada citra ke dalam *audio* menggunakan algoritma LSB. Proses

enkripsi dan penyisipan bertujuan untuk mengamankan pesan agar tidak diketahui oleh orang lain.

a) Enkripsi pesan dengan algoritma GOST

Proses enkripsi pada dasarnya terdapat dua bagian yaitu proses pembangkitan kunci dan proses enkripsi. Proses enkripsi diuraikan pada Gambar 3.2.



Gambar 3.2 Proses enkripsi

1) Pembangkitan kunci

Pembangkitan kunci dilakukan dengan mengkonversi karakter kunci menjadi bentuk biner berdasarkan kode ASCII dan membagi kunci sesuai dengan pengelompokan kunci untuk proses enkripsi.

2) Konversi *plaintext*

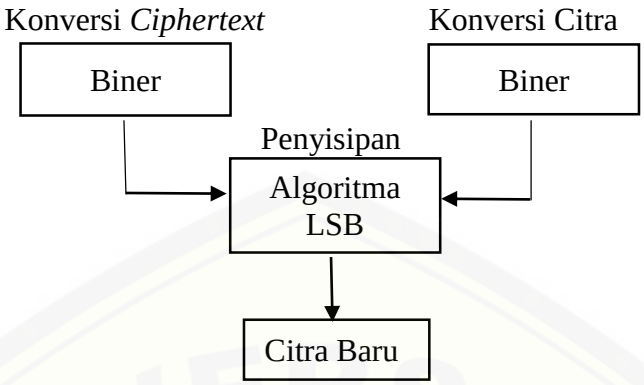
Plaintext awal dalam bentuk karakter, dikonversi menjadi biner dengan ketentuan menggunakan kode ASCII.

3) Enkripsi

Plaintext hasil konversi menjadi biner dilakukan proses enkripsi dengan kunci yang didapatkan dari proses pembangkitan kunci menggunakan algoritma GOST dan menghasilkan *ciphertext*.

b) Penyisipan pada citra

Proses kedua dilakukan dengan menyisipkan *ciphertext* pada citra, proses penyisipan dapat dilihat pada Gambar 3.3.



Gambar 3.3 Proses penyisipan *ciphertext* pada citra

1) Konversi *ciphertext*

   Ciphertext yang didapat dari proses enkripsi menggunakan algoritma GOST dirubah dalam bentuk biner berdasarkan kode ASCII.

2) Konversi citra

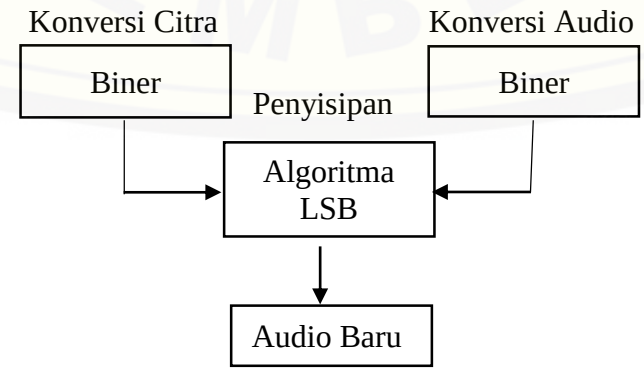
   Citra dirubah menjadi biner dengan menentukan derajat keabuan dari setiap *pixel*, kemudian nilai derajat keabuan disesuaikan dengan nilai pada kode ASCII sehingga didapatkan biner citra.

3) Proses penyisipan

   Biner *ciphertext* dan biner citra dilakukan proses penyisipan menggunakan algoritma LSB menghasilkan citra baru.

c) Penyisipan pada *audio*

Proses penyisipan citra pada *audio* sama dengan proses penyisipan *ciphertext* ke dalam citra. Proses penyembunyian citra ke dalam *audio* ditunjukkan pada Gambar 3.4.



Gambar 3.4 Proses penyisipan citra pada *audio*

1) Konversi citra

Citra baru hasil penyisipan dirubah dalam bentuk biner dengan menghitung derajat keabuan. Nilai yang didapatkan dari derajat keabuan setiap *pixel* disesuaikan dengan nilai kode ASCII, kemudian dirubah dalam bentuk biner.

2) Konversi *audio*

Proses perubahan *audio* didasarkan pada interval frekuensi *audio* tersebut. Proses mendapatkan nilai interval dilakukan dengan membaca *audio* pada MATLAB menggunakan perintah `[y f]=audioread‘namaaudio.wav’;`. Nilai yang didapatkan berasal dari variabel `y`, nilai negatif dirubah menjadi postif, kemudian dikali 255 dan dibulatkan ke bawah, disesuaikan dengan kode ASCII sehingga didapatkan biner *audio*.

3) Proses penyisipan

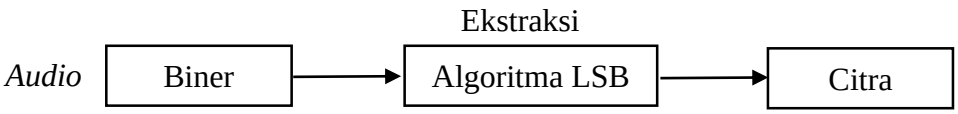
Proses penyisipan pada *audio* sama seperti proses penyisipan pada citra, yaitu mengganti bit kurang berarti pada biner *audio* dengan bit dari biner citra setelah dikonversi menghasilkan *audio* baru.

2. Proses ekstraksi dan dekripsi *ciphertext*

Untuk mendapatkan pesan yang telah disembunyikan ke dalam *audio* dan citra dilakukan proses ekstraksi menggunakan algoritma LSB. Untuk mengembalikan pesan yang berupa sandi menjadi pesan awal dilakukan proses dekripsi menggunakan algoritma GOST.

a) Ekstraksi *audio*

Setelah mendapatkan *audio* baru yang telah disisipi pesan, kemudian dilakukan proses ekstraksi. Proses ekstraksi *audio* ditunjukkan pada Gambar 3.5.



Gambar 3.5 Proses ekstraksi *audio*

1) Konversi *audio*

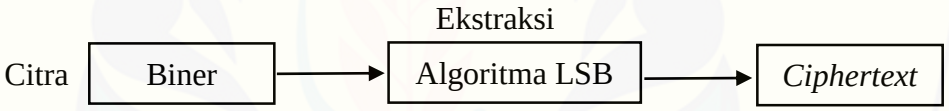
Audio yang telah disisipi pesan dilakukan proses perubahan menjadi biner dengan perintah `[y f]=audioread‘namaaudio.wav’;` pada MATLAB. Nilai negatif dirubah menjadi positif, dikali 255 dan dibulatkan ke bawah, disesuaikan dengan kode ASCII sehingga didapatkan biner *audio*.

2) Proses ekstraksi

Proses ekstraksi dilakukan dengan mengambil bit terakhir pada bit *audio*. Bit terakhir ini mengandung bit untuk biner citra, pengambilan bit dilakukan dengan algoritma LSB.

b) Ekstraksi citra

Citra yang didapatkan dari proses ekstraksi *audio*, kemudian diekstraksi kembali untuk mendapatkan *ciphertext*. Proses ekstraksi citra ditunjukkan pada Gambar 3.6.



Gambar 3.6 Proses ekstraksi citra

1) Konversi citra

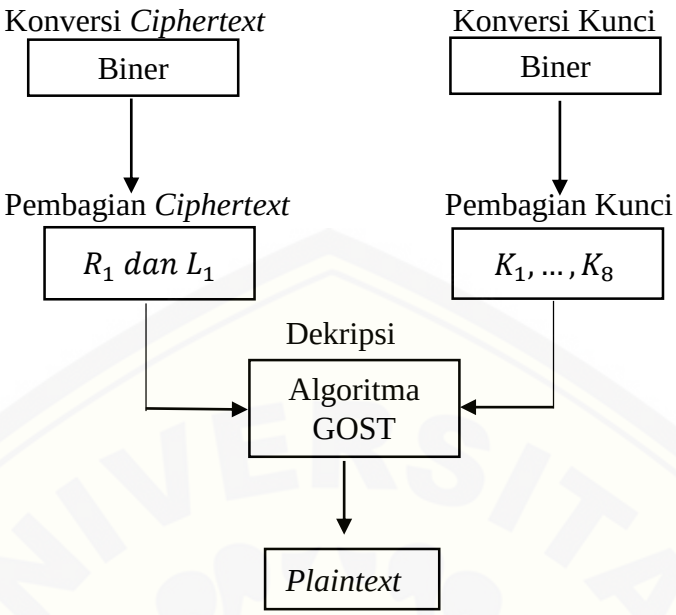
Citra hasil ekstraksi dirubah dalam bentuk biner dengan menghitung derajat keabuan. Nilai derajat keabuan setiap *pixel* disesuaikan dengan nilai kode ASCII, dan dirubah dalam bentuk biner.

2) Proses ekstraksi

Sama seperti proses ekstraksi *audio*, proses ekstraksi citra dilakukan dengan mengambil bit terakhir citra. Bit terakhir yang diambil adalah bit dari biner *ciphertext*, pengambilan bit dilakukan dengan algoritma LSB.

c) Dekripsi dengan algoritma GOST

Proses dekripsi bertujuan untuk mendapatkan *plaintext* asli dari pesan yang disembunyikan. Proses dekripsi ditunjukkan pada Gambar 3.7.



Gambar 3.7 Proses dekripsi

1) Penyusunan *ciphertext*

Biner yang didapatkan dari proses ekstraksi citra diurutkan masing-masing menjadi 8 bit, biner tersebut adalah biner dari *ciphertext*.

2) Pembangkitan kunci

Pembangkitan kunci dilakukan dengan cara merubah karakter kunci ke dalam bentuk biner dan membagi kunci yang telah dirubah dengan aturan pengelompokan kunci proses dekripsi.

3) Proses dekripsi

Proses dekripsi hampir sama dengan proses enkripsi, perbedaan yang terjadi pada penjadwalan kunci. Penjadwalan kunci yang digunakan untuk proses dekripsi..

c. Perancangan Program

Perancangan program yang dilakukan menggunakan *software* MATLAB R2015a, program yang dirancang menggunakan GUI (*Graphic User Interface*) dengan mengatur posisi tombol dan mengatur warna serta membuat program pada MATLAB.

d. Pembuatan Program

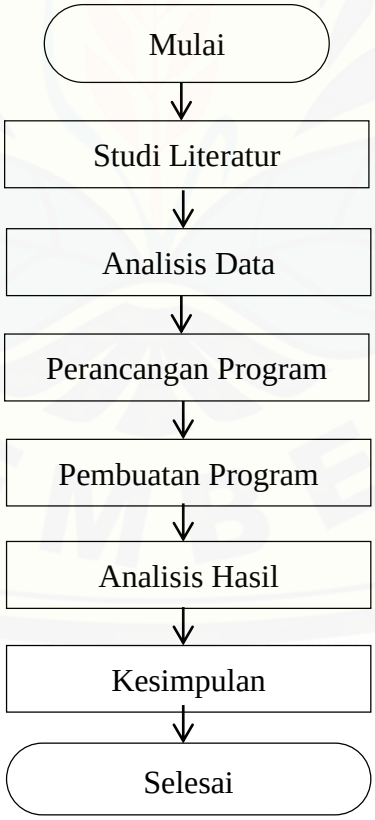
Program enkripsi dan dekripsi *plaintext* dibuat berdasarkan algoritma *Government Standard* (GOST), sedangkan penyisipan dan ekstraksi dibuat berdasarkan algoritma *Least Signicant Bit (*LSB).

e. Analisis Hasil

Menguji jalannya program yang telah dibuat untuk menentukan apakah setiap proses telah berjalan dengan baik sesuai dengan hasil yang diinginkan atau tidak, serta menguji tingkat keamanan dengan metode yang diajukan.

f. Kesimpulan

Mengambil kesimpulan dari penelitian yang dilakukan dengan menganalisis proses penyisipan *ciphertext* dan menganalisis persentase perubahan *cover* setelah penyisipan. Langkah-langkah penelitian menggunakan *flowchart* ditampilkan pada Gambar 3.8.

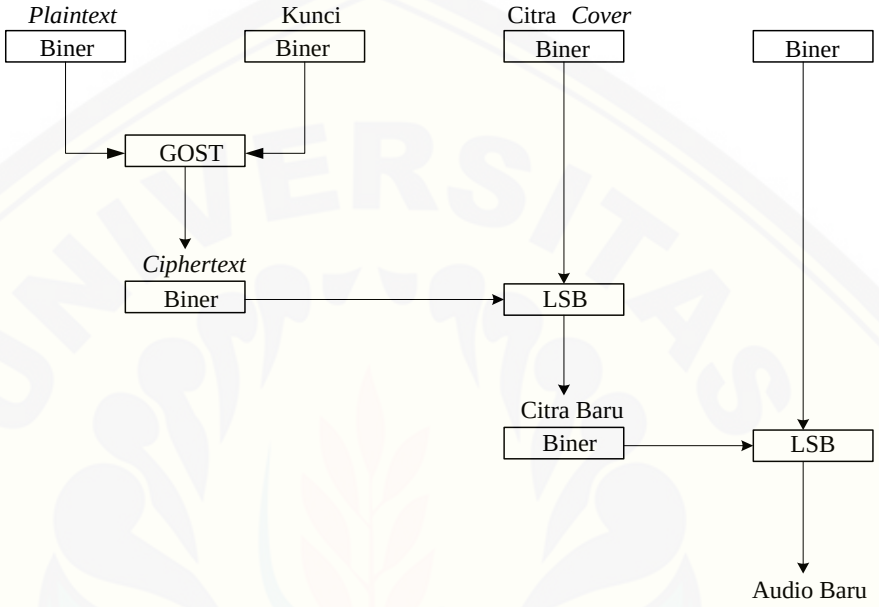


Gambar 3.8 *Flowchart* penelitian

g. Skema penelitian

1. Skema enkripsi dan penyisipan pesan

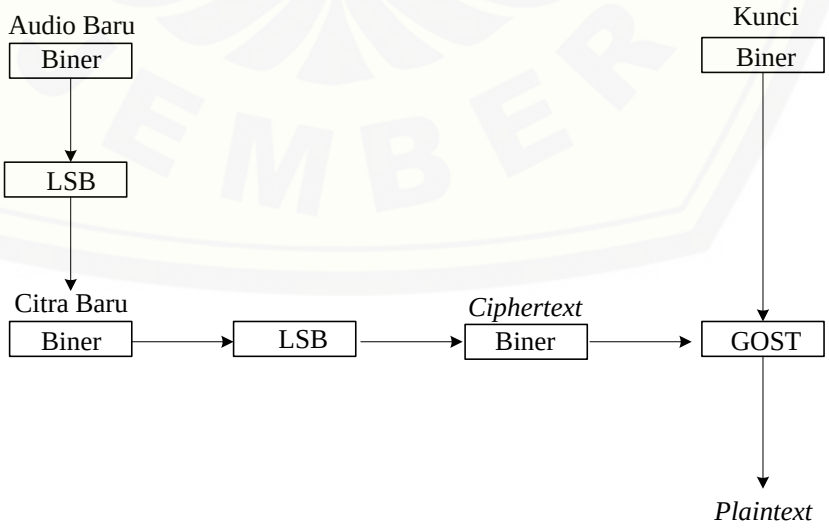
Untuk memudahkan pemahaman dalam penelitian ini, diberikan skema proses enkripsi dan penyisipan pesan digambarkan secara lengkap pada Gambar 3.9.



Gambar 3.9 Skema proses enkripsi dan penyisipan pesan

2. Skema ekstraksi dan dekripsi pesan

Untuk skema proses ekstraksi dan dekripsi pesan digambarkan secara lengkap pada Gambar 3.10.



Gambar 3.10 Skema proses ekstraksi dan dekripsi pesan

BAB 5. PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, terdapat beberapa kesimpulan sebagai berikut.

a. Besarnya ukuran pesan berpengaruh terhadap perubahan *cover* citra setelah penyisipan. Semakin besar ukuran pesan yang disisipkan pada citra, menyebabkan adanya perubahan yang besar pada citra hasil penyisipan.
b. Besar ukuran *pixel* citra yang disisipkan, berpengaruh pada *audio* dengan adanya *noise* yang semakin lama, hal itu dapat diminimalisir dengan menyisipkan citra ke dalam *audio* dengan ukuran yang jauh lebih besar dari ukuran citra.
c. Dekripsi *ciphertext* hasil ekstraksi citra, menggunakan kunci yang sama pada saat proses enkripsi, sehingga menghasilkan *plaintext* yang sama sebelum proses enkripsi.
d. Hasil analisis NPCR dan UACI menunjukkan bahwa persentase tingkat keamanan ditentukan dari ukuran pesan yang disisipkan dan ukuran media *cover* penyisipan. Semakin besar ukuran media *cover*, maka semakin besar pula persentase keamanan. Apabila ukuran pesan yang disisipkan lebih besar dari ukuran media *cover*, maka semakin kecil persentase keamanan.

5.2 Saran

Saran yang diberikan kepada peneliti selanjutnya meliputi:

a. Algoritma kriptografi yang digunakan sebaiknya menggunakan lebih dari dua algoritma, supaya keamanan pesan lebih sulit untuk dipecahkan.
b. Penyisipan dilakukan ke dalam bentuk lain seperti *video*, rar atau *document*.

DAFTAR PUSTAKA

Binanto, I. 2010. *Multimedia Digital Dasar Teori dan Pengembangannya*. Yogyakarta: ANDI.

Dulimarta, H.S. 1997. *Diktat Kuliah Pengolahan Citra*. Bandung: Jurusan Teknik Informatika Institut Teknologi Bandung.

Herryawan, I, P. 2010. Aplikasi Keamanan Data Menggunakan Metoda Kriptografi GOST. *Jurnal TSI*. 1(2): 138-149

Iqbal, M. 2016. The Understanding of GOST Cryptography Technique. *International Journal of Engineering Trends and Technology*. 39 (3): 168-172.

Munir, R. 2006. *Diktat Kuliah IF5054 Kriptografi*. Bandung: Departemen Teknik Informatika, Institut Teknologi Bandung.

Munir, R. 2004. *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Bandung: Departemen Teknik Informatika, Institut Teknologi Bandung.

Menez, J, A. P, C, van Oorschot dan S, A, Vanstone. 1996. *Handbook of Applied Cryptography*. CRC Press.Inc.

Putra, D. 2010. *Pengolahan Citra Digital*. Edisi I. Yogyakarta: ANDI.

Sah, A. 2012. *Aplikasi Pengamanan Data Menggunakan Metode Government Standard* (*GOST*). *Skripsi*. Yogyakarta: Jurusan Teknik Informatika Sekolah Tinggi Manajemen Informatika dan Komputer AMIKOM.

Santoso, S. 2014. Steganografi Audio (WAV) Menggunakan Metode LSB (Least Significant Bit). *Jurnal Informatika* ISSN: 1987-8989. 9(2): 214-224.

Wahyudi, A, A. 2014. Implementasi Steganografi Berkas File MP3 menggunakan Metode LSB (*Least Significant Bit*) Pada Perangkat Mobile Android. *Skripsi*. Yogyakarta: Fakultas Sains dan Teknologi Universitas Islam Negeri Sunan Kalijaga.

Wahyuningsih, Y. 2016. Penyembunyian Pesan Terenkripsi Hill Chiper Pada Audio File dengan Metode Least Significant Bit. *Skripsi.* Jember: Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Wirawan, S. 2011. Implementasi Steganografi pada Berkas Audio WAV untuk Penyisipan Pesan Gambar Menggunakan Metode Low Bit Coding. *Skripsi.* Depok. Fakultas Komputer Sains dan Teknologi Informasi Universitas Guna Dharma.

Yaimini, J. R, Bansal. G, Sharma. B, Kumar. 2016. Image Encryption Schemes: A Complete Survey. *International Journal of Signal Processing, Image Processing and Pattern Recognition.* 9 (7): 157-192.

LAMPIRAN

Lampiran A. Kode ASCII

Tabel 1. Kode ASCII *Control Character* (Character code 0-31)

Dec	***Hex***	***Bin***	***Symbol***	***Dec***	***Hex***	***Bin***	***Symbol***
0	0	0000 0000	NUL	16	10	0001 0000	DLE
1	1	0000 0001	SOH	17	11	0001 0001	DC1
2	2	0000 0010	STX	18	12	0001 0010	DC2
3	3	0000 0011	ETX	19	13	0001 0011	DC3
4	4	0000 0100	EOT	20	14	0001 0100	DC4
5	5	0000 0101	ENQ	21	15	0001 0101	NAK
6	6	0000 0110	ACK	22	16	0001 0110	SYN
7	7	0000 0111	BEL	23	17	0001 0111	ETB
8	8	0000 1000	BS	24	18	0001 1000	CAN
9	9	0000 1001	HT	25	19	0001 1001	EM
10	A	0000 1010	LF	26	1A	0001 1010	SUB
11	B	0000 1011	VT	27	1B	0001 1011	ESC
12	C	0000 1100	FF	28	1C	0001 1100	FS
13	D	0000 1101	CR	29	1D	0001 1101	GS
14	E	0000 1110	SO	30	1E	0001 1110	RS
15	F	0000 1111	SI	31	1F	0001 1111	US

Tabel 2. Kode ASCII *Printable Character* (Character code 32-127)

Dec	***Hex***	***Bin***	***Symbol***	***Dec***	***Hex***	***Bin***	***Symbol***
32	20	0010 0000	Spasi	80	50	0101 0000	P
33	21	0010 0001	!	81	51	0101 0001	Q
34	22	0010 0010	”	82	52	0101 0010	R
35	23	0010 0011	#	83	53	0101 0011	S
36	24	0010 0100	$	84	54	0101 0100	T
37	25	0010 0101	%	85	55	0101 0101	U
38	26	0010 0110	&	86	56	0101 0110	V
39	27	0010 0111	’	87	57	0101 0111	W
40	28	0010 1000	(	88	58	0101 1000	X
41	29	0010 1001	)	89	59	0101 1001	Y
42	2A	0010 1010	*	90	5A	0101 1010	Z
43	2B	0010 1011	+	91	5B	0101 1011	[

44	2C	0010 1100	,	92	5C	0101 1100	\
45	2D	0010 1101	-	93	5D	0101 1101	]
46	2E	0010 1110	.	94	5E	0101 1110	^
47	2F	0010 1111	/	95	5F	0101 1111	_
48	30	0011 0000	0	96	60	0110 0000	~
49	31	0011 0001	1	97	61	0110 0001	a
50	32	0011 0010	2	98	62	0110 0010	b
51	33	0011 0011	3	99	63	0110 0011	c
52	34	0011 0100	4	100	64	0110 0100	d
53	35	0011 0101	5	101	65	0110 0101	e
54	36	0011 0110	6	102	66	0110 0110	f
55	37	0011 0111	7	103	67	0110 0111	g
56	38	0011 1000	8	104	68	0110 1000	h
57	39	0011 1001	9	105	69	0110 1001	i
58	3A	0011 1010	:	106	6A	0110 1010	j
59	3B	0011 1011	;	107	6B	0110 1011	k
60	3C	0011 1100	<	108	6C	0110 1100	l
61	3D	0011 1101	=	109	6D	0110 1101	m
62	3E	0011 1110	>	110	6E	0110 1110	n
63	3F	0011 1111	?	111	6F	0110 1111	o
64	40	0100 0000	@	112	70	0111 0000	p
65	41	0100 0001	A	113	71	0111 0001	q
66	42	0100 0010	B	114	72	0111 0010	r
67	43	0100 0011	C	115	73	0111 0011	s
68	44	0100 0100	D	116	74	0111 0100	t
69	45	0100 0101	E	117	75	0111 0101	u
70	46	0100 0110	F	118	76	0111 0110	v
71	47	0100 0111	G	119	77	0111 0111	w
72	48	0100 1000	H	120	78	0111 1000	x
73	49	0100 1001	I	121	79	0111 1001	y
74	4A	0100 1010	J	122	7A	0111 1010	z
75	4B	0100 1011	K	123	7B	0111 1011	{
76	4C	0100 1100	L	124	7C	0111 1100	|
77	4D	0100 1101	M	125	7D	0111 1101	}
78	4E	0100 1110	N	126	7E	0111 1110	~
79	4F	0100 1111	O	127	7F	0111 1111	Delete

Tabel 3. *The Extended* ASCII *Codes* (Character code 128-255)

Dec	***Hex***	***Bin***	***Symbol***	***Dec***	***Hex***	***Bin***	***Symbol***
128	80	1000 0000	€	192	C0	1100 0000	À

129	81	1000 0001		193	C1	1100 0001	Á
130	82	1000 0010	‚	194	C2	1100 0010	Â
131	83	1000 0011	*ƒ*	195	C3	1100 0011	Ã
132	84	1000 0100	„	196	C4	1100 0100	Ä
133	85	1000 0101	…	197	C5	1100 0101	Å
134	86	1000 0110	†	198	C6	1100 0110	Æ
135	87	1000 0111	‡	199	C7	1100 0111	Ç
136	88	1000 1000	ˆ	200	C8	1100 1000	È
137	89	1000 1001	‰	201	C9	1100 1001	É
138	8A	1000 1010	Š	202	CA	1100 1010	Ê
139	8B	1000 1011	‹	203	CB	1100 1011	Ë
140	8C	1000 1100	Œ	204	CC	1100 1100	Ì
141	8D	1000 1101		205	CD	1100 1101	Í
142	8E	1000 1110	Ž	206	CE	1100 1110	Î
143	8F	1000 1111		207	CF	1100 1111	Ï
144	90	1001 0000		208	D0	1101 0000	Đ
145	91	1001 0001	‘	209	D1	1101 0001	Ñ
146	92	1001 0010	’	210	D2	1101 0010	Ò
147	93	1001 0011	“	211	D3	1101 0011	Ó
148	94	1001 0100	”	212	D4	1101 0100	Ô
149	95	1001 0101	•	213	D5	1101 0101	Õ
150	96	1001 0110	–	214	D6	1101 0110	Ö
151	97	1001 0111	—	215	D7	1101 0111	×
152	98	1001 1000	˜	216	D8	1101 1000	Ø
153	99	1001 1001	™	217	D9	1101 1001	Ù
154	9A	1001 1010	š	218	DA	1101 1010	Ú
155	9B	1001 1011	›	219	DB	1101 1011	Û
156	9C	1001 1100	œ	220	DC	1101 1100	Ü
157	9D	1001 1101		221	DD	1101 1101	Ý
158	9E	1001 1110	ž	222	DE	1101 1110	Þ
159	9F	1001 1111	Ÿ	223	DF	1101 1111	ß
160	A0	1010 0000		224	E0	1110 0000	à
161	A1	1010 0001	¡	225	E1	1110 0001	á
162	A2	1010 0010	¢	226	E2	1110 0010	â
163	A3	1010 0011	£	227	E3	1110 0011	ã
164	A4	1010 0100	¤	228	E4	1110 0100	ä
165	A5	1010 0101	¥	229	E5	1110 0101	å
166	A6	1010 0110	¦	230	E6	1110 0110	æ
167	A7	1010 0111	§	231	E7	1110 0111	ç
168	A8	1010 1000	¨	232	E8	1110 1000	è
169	A9	1010 1001	©	233	E9	1110 1001	é

170	AA	1010 1010	ª	234	EA	1110 1010	ê
171	AB	1010 1011	«	235	EB	1110 1011	ë
172	AC	1010 1100	¬	236	EC	1110 1100	ì
173	AD	1010 1101		237	ED	1110 1101	í
174	AE	1010 1110	®	238	EE	1110 1110	î
175	AF	1010 1111	¯	239	EF	1110 1111	ï
176	B0	1011 0000	°	240	F0	1111 0000	ð
177	B1	1011 0001	±	241	F1	1111 0001	ñ
178	B2	1011 0010	²	242	F2	1111 0010	ò
179	B3	1011 0011	³	243	F3	1111 0011	ó
180	B4	1011 0100	´	244	F4	1111 0100	ô
181	B5	1011 0101	µ	245	F5	1111 0101	õ
182	B6	1011 0110	¶	246	F6	1111 0110	ö
183	B7	1011 0111	·	247	F7	1111 0111	÷
184	B8	1011 1000	¸	248	F8	1111 1000	ø
185	B9	1011 1001	¹	249	F9	1111 1001	ù
186	BA	1011 1010	º	250	FA	1111 1010	ú
187	BB	1011 1011	»	251	FB	1111 1011	û
188	BC	1011 1100	¼	252	FC	1111 1100	ü
189	BD	1011 1101	½	253	FD	1111 1101	ý
190	BE	1011 1110	¾	254	FE	1111 1110	þ
191	BF	1011 1111	¿	255	FF	1111 1111	ÿ

Lampiran B. Potongan Citra RGB Kanal *Red*

222	222	225	226	223	221	219	220	220	219	. . .	230
223	224	226	226	222	220	220	221	221	221	. . .	218
223	225	227	226	223	221	220	222	221	222	. . .	178
223	223	226	226	225	223	221	222	223	224	. . .	122
226	225	223	224	227	226	224	225	226	225	. . .	84
227	226	223	221	225	226	226	226	227	223	. . .	78
227	226	222	221	225	225	226	226	225	222	. . .	86
227	225	223	225	224	225	227	226	224	221	. . .	92
224	224	226	227	227	227	227	227	223	221	. . .	95
222	223	224	227	228	227	226	225	223	222	. . .	95
223	223	226	227	228	228	225	224	222	224	. . .	97
226	226	228	228	227	228	225	223	224	226	. . .	99
226	226	226	226	229	229	228	226	227	230	. . .	102
225	225	224	225	228	229	227	228	229	231	. . .	102
227	226	223	226	227	228	228	230	232	232	. . .	100
229	229	226	227	226	227	229	231	233	232	. . .	99
.											
.											
224	222	221	221	223	225	225	223	221	220	. . .	180

Lampiran C. Potongan Citra RGB Kanal *Green*

138	137	136	137	137	136	134	134	135	135	. . .	145
137	137	136	137	137	135	133	132	134	135	. . .	132
136	136	135	136	136	134	131	130	132	133	. . .	93
134	134	133	134	135	133	131	129	130	130	. . .	43
132	131	132	132	134	133	131	129	129	129	. . .	14
130	130	131	131	132	133	132	130	130	130	. . .	14
131	131	131	131	131	132	132	130	130	132	. . .	22
132	132	132	131	130	131	131	130	131	132	. . .	23
133	132	131	131	130	131	131	132	132	132	. . .	21
133	132	131	131	131	131	132	133	133	132	. . .	21
131	130	131	132	133	133	133	134	134	133	. . .	24
128	129	130	131	133	134	133	134	134	133	. . .	27
127	128	129	131	133	134	134	134	134	132	. . .	29
130	131	132	133	134	134	134	133	133	132	. . .	30
132	133	134	134	134	134	134	133	133	133	. . .	31
133	133	134	133	134	135	136	134	134	135	. . .	33
.											
.											
24	25	27	26	20	19	42	95	158	194	. . .	71

Lampiran D. Potongan Citra RGB Kanal *Blue*

127	126	125	125	123	119	117	116	113	110	. . .	119
125	125	124	124	121	118	116	114	112	110	. . .	111
121	121	120	120	118	115	113	111	110	109	. . .	92
116	116	116	116	114	112	110	107	107	107	. . .	71
112	112	112	112	111	110	108	105	105	106	. . .	61
109	109	109	108	108	108	107	105	105	107	. . .	62
108	107	107	105	106	107	107	105	105	107	. . .	63
108	107	106	104	105	106	107	106	106	106	. . .	59
108	107	105	104	105	105	106	107	106	106	. . .	58
108	107	105	106	106	105	106	107	108	108	. . .	59
107	107	106	107	108	106	107	109	109	110	. . .	60
106	107	107	109	109	108	109	109	110	111	. . .	61
106	107	108	109	109	109	109	109	110	111	. . .	63
108	109	109	109	109	109	108	108	109	110	. . .	64
109	110	110	109	109	109	108	107	109	110	. . .	65
109	109	109	108	107	108	109	108	110	111	. . .	67
.											
.											
58	60	63	66	61	57	72	112	157	182	. . .	80

Lampiran E. Potongan Citra GRY Kanal *Graysacle*

162	162	162	163	164	161	157	155	158	157	. . .	141
162	161	161	162	163	161	157	155	157	157	. . .	150
160	159	159	160	161	160	156	154	156	156	. . .	113
159	158	157	158	159	158	155	154	156	155	. . .	57
158	157	156	156	158	158	155	154	156	155	. . .	41
158	156	155	156	157	158	156	155	156	156	. . .	51
158	156	155	155	157	158	157	156	157	157	. . .	55
158	157	155	155	157	158	158	157	158	157	. . .	50
155	155	155	156	156	157	157	157	159	159	. . .	43
155	155	156	156	156	157	157	157	158	158	. . .	51
155	156	156	156	157	157	157	157	158	158	. . .	56
156	156	156	157	157	157	158	158	157	157	. . .	54
156	156	157	157	157	158	158	158	157	158	. . .	51
157	157	157	157	158	158	158	159	158	159	. . .	51
157	157	157	158	158	158	159	159	159	160	. . .	56
158	157	158	158	158	159	159	160	162	163	. . .	50
160	158	158	158	158	159	160	161	162	165	. . .	48
.											
.											
49	46	50	50	56	47	55	109	172	202	. . .	103

Lampiran F. Potongan Sampel Matriks *Audio* Musik BTH

0.6862	0.1679	0.6921	0.1437	0.8064	0.1531	0.7184	0.0413
0.3101	-0.1741	0.1767	-0.2349	0.0459	-0.2573	0.0817	-0.1277
-0.0289	-0.0054	-0.0139	0.0548	0.0848	0.2068	0.0502	0.294
0.1283	0.4948	0.1541	0.5665	0.0713	0.4803	0.1126	0.4681
0.6130	0.8110	0.7656	0.8627	0.8200	0.8311	0.7631	0.7323
0.2198	0.2904	0.0760	0.1392	-0.0633	0.0360	-0.0278	0.0994
0.4688	0.4795	0.5918	0.5894	0.6005	0.6207	0.5187	0.4800
0.7855	0.4009	0.8765	0.3699	0.7607	0.1303	0.7568	-0.0548
0.9135	-0.0329	0.9651	0.0501	0.8565	0.1010	0.8001	0.2502
0.5854	0.5956	0.2466	0.6131	0.0839	0.7343	0.0843	0.8470
0.3485	0.9087	0.5088	0.8763	0.5763	0.7921	0.6768	0.7741
0.6116	0.5820	0.5018	0.5104	0.5132	0.5497	0.5682	0.5765
0.6087	0.5076	0.4851	0.5114	0.3010	0.5078	0.1945	0.5212
0.1992	0.7097	0.2072	0.7189	0.2774	0.6807	0.4410	0.6470
0.6334	0.6248	0.7045	0.7045	0.8114	0.7440	0.7914	0.6346
0.6196	0.3740	0.5269	0.2737	0.4553	0.2543	0.4385	0.3876
0.0177	0.6479	-0.2131	0.7018	-0.2888	0.7544	-0.2565	0.7963
-0.1772	0.6754	-0.0285	0.6517	0.1957	0.6940	0.3889	0.7767
0.2701	0.5690	0.1414	0.3438	0.2394	0.3166	0.2745	0.2382
0.2267	0.0170	0.4593	0.1379	0.5274	0.1291	0.4324	0.0171

0.2350	-0.0661	0.1335	-0.0717	0.1187	0.0306	0.1884	0.2318
0.2840	0.4099	0.4008	0.4640	0.5182	0.5000	0.5229	0.4674
0.4822	0.4232	0.4339	0.3527	0.3007	0.2223	0.2573	0.1831
0.4552	0.2990	0.5132	0.3499	0.5417	0.4505	0.4872	0.4637
0.4884	0.3949	0.4812	0.3322	0.4406	0.2310	0.4951	0.2260

Lampiran G. Program Enkripsi GOST

```
function s=Enkripsi_GOST(plaintext,key);
format long
for i=1:8
      plain2bin(8*i-7:8*i)=dec2bin(double(plaintext(i)),8);
end
R(1,:)=plain2bin(32:-1:1);
L(1,:)=plain2bin(64:-1:33);
for i=1:32
      key2bin(8*i-7:8*i)=dec2bin(double(key(i)),8);
end
for i=1:8
      K(i,:)=key2bin(32*i:-1:32*i-31);
End
kl=[K ; K ; K ; K(8:-1:1,:)];
S_Box=[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 ;
4 10 9 2 13 8 0 14 6 11 1 12 7 15 5 3 ;
14 11 4 12 6 13 15 10 2 3 8 1 0 7 5 9 ;
5 8 1 13 10 3 4 2 14 15 12 7 6 0 9 11 ;
7 13 10 1 0 8 9 15 14 4 6 12 11 2 5 3 ;
6 12 7 1 5 15 13 8 4 10 9 14 0 3 11 2 ;
4 11 10 0 7 2 1 13 3 6 8 5 9 12 15 14 ;
13 11 4 1 3 15 5 9 0 10 14 7 6 8 2 12 ;
1 15 13 0 5 7 10 4 9 2 3 14 6 11 8 12];
kk=0;
for i=2:33
      kk=kk+1;
      hasil(i-1,:)=dec2bin(str2num(sprintf('%f',mod((bin2dec(R(i-
      1,:))+bin2dec(kl(i-1,:))),2^32))),32);
for j=1:8
      sbox(i-1,4*j-3:4*j)=dec2bin(S_Box(j+1,bin2dec(hasil(i-1,4*j-
      3:4*j))+1),4);
end
RLS(i-1,:)=[sbox(i-1,12:32) sbox(i-1,1:11)];
for k=1:32
      R(i,k)=num2str(xor(str2num(RLS(i-1,k)),str2num(L(i-1,k))));
      L(i,:)=R(i-1,:);
end
```

```
end
R32=R(32,:);
L32=R(33,:);
blkR32=R32(32:-1:1);
blkL32=L32(32:-1:1);
RL=[blkR32 blkL32];
for i=1:8
      text(2*i-1:2*i)=dec2hex((bin2dec(RL(8*i-7:8*i))),2);
end
s=text;
```

Lampiran H. Program Dekripsi GOST

```
function s=Dekripsi_GOST(plaintext, key);
format long
for i=1:8
       plain2bin(8*i-7:8*i)=dec2bin(hex2dec(plaintext(2*i-1:2*i)),8);
end
R(1,:)=plain2bin(32:-1:1);
L(1,:)=plain2bin(64:-1:33);
for i=1:32
       key2bin(8*i-7:8*i)=dec2bin(double(key(i)),8);
end
for i=1:8
       K(i,:)=key2bin(32*i:-1:32*i-31);
end
km=[K ; K ; K ; K(8:-1:1,:)];
kl=km(32:-1:1,:);
S_Box=[0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 ;
4 10 9 2 13 8 0 14 6 11 1 12 7 15 5 3 ;
14 11 4 12 6 13 15 10 2 3 8 1 0 7 5 9 ;
5 8 1 13 10 3 4 2 14 15 12 7 6 0 9 11 ;
7 13 10 1 0 8 9 15 14 4 6 12 11 2 5 3 ;
6 12 7 1 5 15 13 8 4 10 9 14 0 3 11 2 ;
4 11 10 0 7 2 1 13 3 6 8 5 9 12 15 14 ;
13 11 4 1 3 15 5 9 0 10 14 7 6 8 2 12 ;
1 15 13 0 5 7 10 4 9 2 3 14 6 11 8 12];
kk=0;
for i=2:33
       kk=kk+1;
       hasil(i-1,:)=dec2bin(str2num(sprintf('%f',mod((bin2dec(R(i-
       1,:))+bin2dec(kl(i-1,:))),2^32))),32);
for j=1:8
       sbox(i-1,4*j-3:4*j)=dec2bin(S_Box(j+1,bin2dec(hasil(i-1,4*j-
       3:4*j))+1),4);
end
RLS(i-1,:)=[sbox(i-1,12:32) sbox(i-1,1:11)];
for k=1:32
       R(i,k)=num2str(xor(str2num(RLS(i-1,k)),str2num(L(i-1,k))));
       L(i,:)=R(i-1,:);
```

```
end
end
R32=R(32,:);
L32=R(33,:);
blkR32=R32(32:-1:1);
blkL32=L32(32:-1:1);
RL=[blkR32 blkL32];
for i=1:8
      text(i)=char(bin2dec(RL(8*i-7:8*i)));
end
s=text;
```