PENYISIPAN PESAN PADA CITRA DENGAN METODE *MODIFIED LEAST SIGNIFICANT BIT* DAN ALGORITMA *DATA ENCRYPTION STANDART*

SKRIPSI

Oleh

Fajar Rizky Hogantara
NIM 121810101072

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2016

PENYISIPAN PESAN PADA CITRA DENGAN METODE *MODIFIED LEAST SIGNIFICANT BIT* DAN ALGORITMA *DATA ENCRYPTION STANDART*

SKRIPSI

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

Oleh

Fajar Rizky Hogantara

NIM 121810101072

JURUSAN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS JEMBER

2016

PERSEMBAHAN

Skripsi ini saya persembahkan untuk :

1. Kakek, Nenek, Bapak Kabul Santoso S.Kep, Ibu Nurhayati yang telah memberikan cinta dan kasih.
2. Adikku tercinta Nadia Yolanda.
3. Teman – teman BATHIC'S12 yang selalu memberikan bantuan.
4. Keluarga besar dari bapak dan ibu yang namanya tidak bisa saya sebut satu – satu .
5. Almamater tercinta Jurusan Matematika FMIPA UNEJ, SMAN 1 GLENMORE, SMPN 3 GLENMORE, SDN 9 TEGALHARJO DAN SDN 6 MUHAMMADIYAH GENTENG

MOTTO

“Manusia dapat dihancurkan, manusia dapat dikalahkan, namun manusia tidak dapat dimusnahkan dan dihancurkan selama manusia percaya pada hatinya sendiri”

PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : Fajar Rizky Hogantara

NIM : 121810101072

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul ”Penyisipan Pesan pada Citra dengan Metode *Modified Least Significant Bit* dan Algoritma *Data Encryption Standart*” adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi manapun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

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

Jember, Desember 2016
Yang menyatakan,

Fajar Rizky Hogantara
NIM 121810101072

SKRIPSI

PENYISIPAN PESAN PADA CITRA DENGAN METODE *MODIFIED LEAST SIGNIFICANT BIT* DAN ALGORITMA *DATA ENCRYPTION STANDART*

Oleh

Fajar Rizky Hogantara

NIM 121810101072

Pembimbing:

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

Dosen Pembimbing 2 : M. Ziaul Arif, S.Si., M.Sc.

PENGESAHAN

Skripsi berjudul ”Penyisipan Pesan pada Citra dengan Metode *Modified Least Significant Bit* dan Algoritma *Data Encryption Standart*”
telah diuji dan disahkan pada:
hari, tanggal :
tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Tim Penguji:

Ketua,	Sekretaris,
Ahmad Kamsyakawuni, S.Si., M.Kom.	M. Ziaul Arif, S.Si., M.Sc.
NIP. 197211291998021001	NIP. 198501112008121002
Penguji I,	Penguji II,
Prof. Drs. Kusno, DEA., PhD	Dr. Alfian Futuhul Hadi, S,Si., M.Si
NIP. 196101081986021001	NIP. 19740719200021001

Mengesahkan
Dekan

Drs. Sujito, Ph.D.
NIP. 196102041987111001

PRAKATA

Puji syukur kehadirat Tuhan Yang Maha Esa atas segala kuasa-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul "Penyisipan Pesan Pada Citra dengan Metode *Modified Least Significant Bit* dan Algoritma *Data Encryption Standart*". Penulisan tugas akhir ini dilakukan guna memenuhi salah satu syarat untuk menyelesaikan Program Studi Matematika (S1) dan mencapai gelar sarjana Sains pada Jurusan Matematika FMIPA Universitas Jember.

Pada kesempatan ini, dengan segala hormat penulis mengucapkan terima kasih kepada:

1. Bapak Ahmad Kamsyakawuni, S.Si., M.Kom selaku dosen pembimbing utama dan Bapak M. Ziaul Arif S.Si., M.Sc selaku dosen pembimbing anggota.
2. Bapak Prof. Drs. Kusno, DEA., PhD dan Bapak Dr. Alfian Futuhul Hadi, S.Si., M.Si selaku dosen penguji.
3. Bapak Drs. Sujito, Ph.D. selaku dekan Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.
4. Keluarga Bathic's 12 yang telah banyak membantu.
5. Dosen FMIPA dan guru guruku mulai dari TK, SD, SMP, dan SMA yang telah banyak memberikan ilmu kepada saya.

Penulis menyadari bahwa penulisan ini masih jauh dari sempurna. Oleh karena itu, penulis mengharap kritik dan saran demi kesempurnaan penelitian selanjutnya. Semoga tugas akhir ini dapat bermanfaat bagi kita semua.

Jember, Desember Penulis

RINGKASAN

Penyisipan Pesan pada Citra dengan Metode *Modified Least Significant Bit* dan Algoritma *Data Encryption Standart*; Fajar Rizky Hogantara, 121810101072; 2016; 58 Halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Perkembangan kemajuan teknologi informasi semakin memudahkan para pelaku kejahatan komputer untuk mendukung kegiatannya, sehingga mengganggu privasi seseorang. Oleh karena itu, diperlukan sebuah sistem atau aplikasi sebagai pengaman data. Teknik pengamanan informasi yang dapat digunakan adalah kriptografi dan steganografi.

Kriptografi adalah ilmu yang mempelajari teknik – teknik matematika yang ditujukan untuk keamanan informasi. Kriptografi memiliki dua konsep yang sangat penting yaitu enkripsi dan dekripsi. Enkripsi adalah proses dimana informasi atau data yang hendak dikirim diubah menjadi bentuk yang hampir tidak dikenali sebagai informasi awalnya dengan menggunakan algoritma tertentu. Dekripsi adalah kebalikan dari enkripsi yaitu mengubah kembali bentuk tersamar tersebut menjadi informasi awal.

Steganografi adalah seni menyisipkan pesan didalam pesan lain sehingga orang lain tidak menyadari ada sesuatu didalam pesan tersebut. Salah satu metode yang akan dibahas dalam penelitian ini adalah algoritma DES (*Data Encryption Standart*) dengan menambahkan steganografi. Salah satu metode steganografi yang digunakan adalah *Modified Least Significant Bit* (MLSB). Metode ini merupakan hasil dari pengembangan metode *Least Significant Bit* (LSB). Pada metode ini bilangan 8 bit akan dikonversi menjadi bilangan 5 bit.

Data yang digunakan adalah data ASCII (*Standard Code for Information Interchange*) dan gambar (*image*) berekstensi bmp, png, gif. Pada penelitian ini proses pengaman pesan berupa text menggabungkan kriptografi dan steganografi. *Plaintext* dienkripsi menggunakan algoritma DES dengan kunci 64 bit atau 8 karakter sehingga menghasilkan *ciphertext*. Masing – masing karakter pada

ciphertext diubah menjadi desimal sesuai kode ASCII, selanjutnya dikonversi menjadi bilangan biner 8 bit. Selanjutnya bilangan biner 8 bit dikonversi menggunakan metode *Modified Least Siginificant Bit* (MLSB) sehingga menjadi 5 bit. Bit yang telah dikonversi disisipkan kedalam media citra menggunakan metode *Least Significant Bit* (LSB) dan menghasilkan *Stego Object*. *Stego Object* inilah yang akan dikirim kepada pikah kedua untuk dikstrak kembali. Proses ekstraksi pesan dengan mengambil bit terakhir pada citra dan mengkonversi menggunakan metode *Modified Least Significant Bit* (MLSB) lalu didekripsi menggunakan algoritma *Data Encryption Standart* (DES).

Waktu komputasi untuk menyisipkan pesan bergantung pada panjang pesan yang disisipkan. Semakin besar ukuran pesan, waktu yang dibutuhkan untuk penyisipan semakin lama. Ukuran citra sebelum dan sudah penyisipan tidak terjadi perubahan sama sekali.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Perkembangan kemajuan teknologi informasi saat ini semakin memudahkan para pelaku kejahatan komputer dengan menyalahgunakan teknologi komputer untuk mendukung kegiatannya, dimana aktivitas para pelaku kejahatan komputer sangat mengganggu privasi seseorang. Oleh karena itu, diperlukan sebuah sistem atau aplikasi sebagai pengaman data sehingga dapat mempersulit para pelaku kejahatan komputer dan membantu para penguna teknologi dalam hal pengaman data yang diakses tersebut.

Salah satu sistem yang diperlukan sebagai pengaman suatu data adalah kriptografi, secara umum kriptografi adalah ilmu yang mempelajari teknik – teknik matematika yang ditujukan untuk keamanan informasi. Kriptografi tidak berarti hanya memberikan keamanan informasi saja, namun lebih kearah metode – metode yang digunakan. Salah satu metode yang akan dibahas dalam penelitian ini adalah algoritma DES dengan menambahkan steganografi. Steganografi merupakan seni untuk menyisipkan pesan didalam pesan lain sehingga orang lain tidak menyadari ada sesuatu didalam pesan tersebut. Penambahan steganografi untuk menyembunyikan *ciphertext* hasil enkripsi algoritma DES ditujukan untuk mempersulit para pelaku kejahatan komputer untuk meretas daya yang telah diamankan tersebut sehinga keamanan lebih terjamin.

Muhendra (2016) telah membahas Implementasi Kriptografi *Affine Cipher* Pada Citra Digital Hasil Steganografi Metode *Parity Coding* Dengan *Pseudo Random Number Generator* (PRNG). Pada proses penelitiannya dibutuhkan input berupa *plaintext* yang akan dienkripsi menjadi *ciphertext* menggunakan dua buah kunci. Kemudian *ciphertext* disisipkan secara acak pada media citra digital. Sehingga cukup untuk meningkatkan keamanan penyembunyian pesan.

Rosdiana (2015) telah membahas Pengkodean Citra Digital Hasil Steganografi dengan *Metode Least Significant Bit* untuk Data Teks Terenkripsi

dengan Algoritma *Hill Cipher*. Pada proses penelitiannya *plaintext* akan dienkripsi menjadi *ciphertext* menggunakan kunci *K* yang merupakan matrik ordo 2×2. Kemudian *ciphertext* disisipkan pada citra digital dengan mengganti bit terakhir. Sehingga citra yang disisipkan terlihat sama dengan citra aslinya.

1.2 Rumusan Masalah

Adapun rumusan masalah yang didapat dari latar belakang sebagai berikut.

a. Bagaimana pengaruh proses enkripsi dan dekripsi serta penyisipan dan ekstraksi pesan pada citra (*image*) terhadap waktu komputasi?
b. Bagaimana metode *Modified Least Significant Bit* (MLSB) dapat meningkatkan keamanan penyembunyian pesan?

1.3 Batasan Masalah

Adapun batasan masalah dari penulisan skripsi ini adalah:

a. Media digital penampung berupa file citra (*image*) dengan ekstensi bmp, gif dan png.
b. Karakter yang akan digunakan adalah ASCII (*American Strandart Code for Information Interchange*)

1.4 Tujuan

Adapun tujuan dari penelitian ini adalah:

a. Mengetahui pengaruh waktu komputasi dari proses enkripsi dan dekripsi, serta penyisipan dan ekstraksi pesan dari sebuah citra (*image*).
b. Mengetahui bahwa metode *Modified Least Significant Bit* (MLSB) dapat meningkatkan keamanan penyembunyian pesan.

1.5 Manfaat Penelitian

Manfaat yang diharapkan dari penulis proposal matematika ini antara lain:

a. Menambah pengetahuan baru tentang kriptografi dan steganografi.
b. Memberikan motivasi pada penulis lain untuk menggunakan metode yang berbeda.

c. Hasil penulisan ini diharapkan dapat digunakan sebagai pengembangan atau perluasan ilmu dan aplikasi dalam kriptografi dan steganografi.

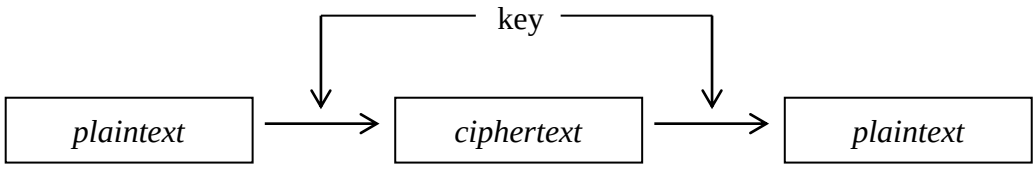
BAB 2. TINJAUAN PUSTAKA

2.1 Kriptografi

Kriptografi (*cryptography*) berasal dari bahasa Yunani: *"cryptos"* artinya *"secret"* (rahasia), sedangkan *"graphein"* artinya *"writing"* (tulisan). Jadi, kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan. Kata "seni" berasal dari fakta sejarah bahwa pada masa masa awal sejarah kriptografi, setiap orang mungkin mempunyai cara yang unik untuk merahasiakan pesan. Cara-cara unik tersebut mungkin berbeda – beda pada setiap pelaku kriptografi sehingga setiap cara menulis pesan rahasia mempunyai nilai estetika tersendiri sehingga kriptografi berkembang menjadi sebuah seni merahasiakan pesan (Munir, 2006).

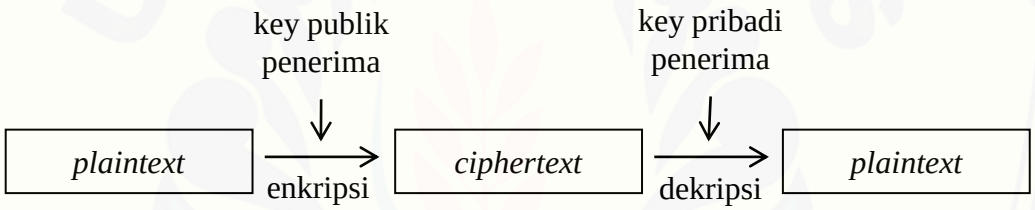
Kriptografi memiliki dua konsep penting yaitu enkripsi dan dekripsi. Enkripsi adalah proses dimana informasi yang akan dikirim diubah menjadi bentuk yang hampir tidak dikenali. Dekripsi adalah adalah proses mengubah kembali bentuk tidak dikenali menjadi informasi awal. Sebuah pesan atau data yang masih asli dan belum mengalami penyandian dikenal dengan istilah *plaintext*. Kemudian setelah disamarkan dengan suatu cara penyandian, maka *plaintext* ini disebut sebagai *ciphertext* (Pabokory dkk, 2015).

Berdasarkan jumlah kunci, ada dua jenis sistem kriptografi yaitu sistem kriptografi simetris dan sistem kriptografi asimetris. Sistem kriptografi simetris menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Algoritma kriprografi simetris sering disebut algoritma kunci rahasia, algoritma kunci tunggal, atau algoritma satu kunci, dan mengharuskan pengirim dan penerima menyetujui suatu kunci tertentu. Kelebihan dari algoritma kriprografi simetris adalah waktu proses untuk enkripsi dan dekripsi relatif cepat. Hal ini disebabkan efesiensi yang terjadi pada pembangkit kunci. Sedangkan kelemahan dari algoritma kriprografi simetris adalah untuk tiap pengiriman pesan dengan *user* yang berbeda dibutuhkan kunci yang berbeda juga, sehingga akan terjadi kesulitan dalam manajemen kunci tersebut (Nathasia dkk, 2011).



Gambar 2.1 Model Sederhana Sistem Kriptografi Simetris

Sedangkan sistem kriptografi asimetris dikenal dengan kriptografi kunci-publik (*public-key cryptography*), menggunakan dua jenis kunci, yaitu kunci public (*public key*) dan kunci rahasia (*secret key*). Kunci publik merupakan kunci yang digunakan untuk mengenkripsi pesan. Sedangkan kunci rahasia digunakan untuk mendekripsi pesan. Kunci publik bersifat umum, sehingga dapat dilihat oleh siapa saja. Sedangkan kunci rahasia adalah kunci yang dirahasiakan dan hanya orang-orang tertentu saja yang boleh mengetahuinya (Nathasia dkk, 2011).



Gambar 2.2 Model Sederhana Sistem Kriptografi Asimetris

Berdasarkan cara memproses teks (*plaintext*), *cipher* dapat dapat dikategorikan menjadi dua jenis: *block cipher* dan *stream cipher*. *Block cipher* bekerja dengan memproses data secara blok, dimana beberapa karakter atau data digabungkan menjadi satu blok. Setiap proses satu blok menghasilkan keluaran satu blok juga. Sementara itu *stream cipher* bekerja memproses masukan (karakter atau data) secara terus – menerus dan menghasilkan data pada saat bersamaan (Primartha, 2011).

2.2 Algoritma DES

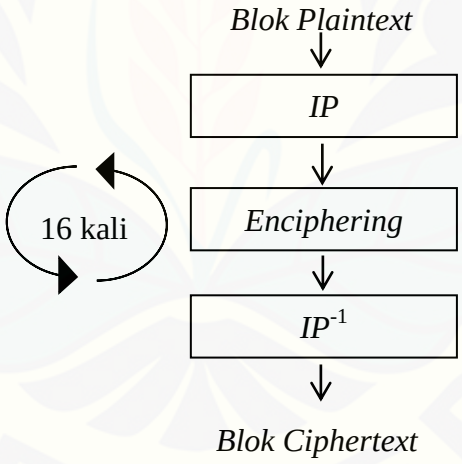
DES (*Data Encryption Standart*) merupakan salah satu metode penyandian dengan sistem *block cipher*. Dalam sistem penyandiannya dilakukan pengacakan blok demi blok dengan blok *plaintext* 64 bit dan menghasikan *ciphertext* yang juga 64 bit, algoritma yang digunakan adalah kunci simetris dengan panjang kunci 64 bit.

Algoritma DES termasuk algoritma kunci simetris, dimana untuk proses enkripsi dan dekripsi menggunakan kunci yang sama. Sehingga meskipun kriptoanalis mengerti dengan baik algoritma yang digunakan untuk menyandikan pesan tersebut, seorang kriptografer tidat dapat mendekripsikan pesan tersebut jika tidak memiliki kunci yang digunakan (Danuri, 2011).

2.2.1 Algoritma Enkripsi DES

Plaintext dienkrip dalam blok-blok 64 bit menjadi 64-bit data *ciphertext* menggunakan kunci 56 bit kunci internal (*internal key*). DES mentransformasikan input 64 bit dalam beberapa tahap enkripsi ke dalam output 64bit. Dengan demikian, DES termasuk *block cipher*. Dengan tahapan dan kunci yang sama, DES digunakan untuk membalik enkripsi. Kunci internal pada algoritma DES dibangkitkan dari kunci eksternal (*external key*) 64 bit (Primartha, 2011).

Skema global dari proses algoritma DES dapat dilihat pada Gambar 2.3.

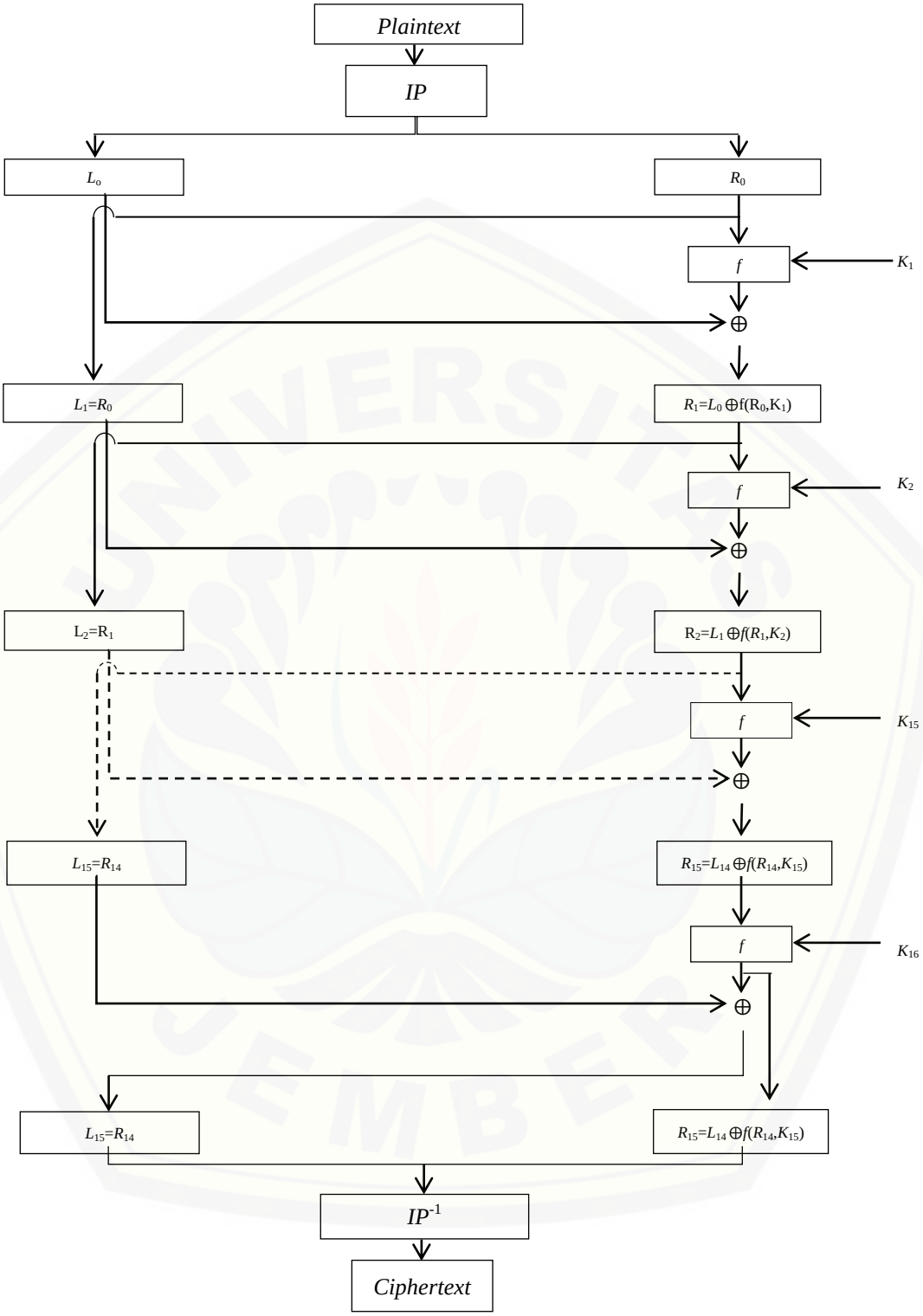


Gambar 2.3 Skema Global Algoritma DES

Skema global algoritma DES adalah sebagai berikut:

a. Blok *plaintext* dipermutasi dengan matriks permutasi awal (*initial permutation* atau *IP*).
b. Hasil permutasi awal kemudian di *enchipering* sebanyak 16 kali putaran. Setiap putaran menggunakan kunci internal yang berbeda.

c. Hasil *enchipering* kemudian dipermutasi dengan matriks permutasi balikan (*invers initial permutation* atau IP^{-1}) menjadi blok *ciphertext*.
d. Skema algoritma DES dapat dilihat pada Gambar 2.4.



Gambar 2.4 Skema Dasar Enkripsi Algoritma DES

Dalam algoritma DES, terdapat kunci eksternal dan kunci internal. Kunci internal dibangkitkan dari kunci eksternal yang diberikan oleh pengguna. Kunci

internal dapat dibangkitkan sebelum proses enkripsi ataupun bersamaan dengan proses enkripsi. Kunci eksternal panjangnya 64 bit atau 8 karakter. Karena ada 16 putaran, maka kunci internal yang dibutuhkan sebanyak 16 buah, yaitu K_1, K_2, ..., K_{16}. Untuk mengaitkan kunci internal diperlukan beberapa langkah.

Kunci eksternal 64 bit, dikompresi terlebih dahulu menjadi 54 bit menggunakan matriks permutasi kompresi PC-1. Dalam permutasi tiap bit ke-8 dari 8 *byte* kunci akan diabaikan. Sehingga akan ada penggunaan 8 bit dari 64 bit awal kunci eksternal.

Setelah didapatkan 56 bit hasil permutasi, selanjutnya 56 bit ini akan dibagi menjadi 2 bagian, kiri dan kanan, yang masing-masing panjangnya 28 bit. Lalu ke-2 bagian tersebut akan disimpan ke dalam C_0 dan D_0.

C_0: berisi bit-bit dari K pada posisi:

57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18
10, 2, 59, 51, 43, 35, 27, 19,11, 3, 60, 52, 44, 36

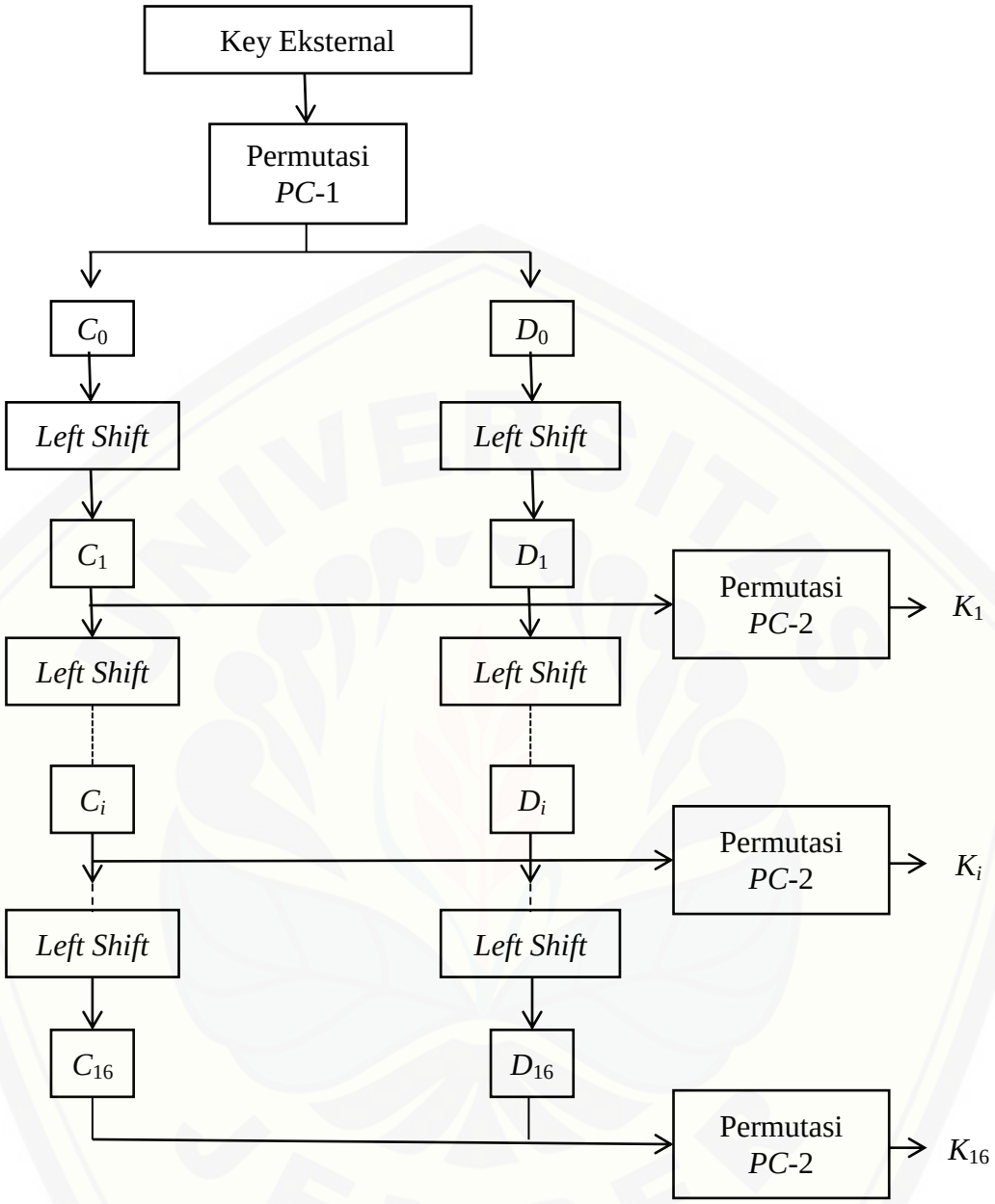
D_0: berisi bit-bit dari K pada posisi:

63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22
14, 6, 61, 53, 45, 37, 29, 21, 21, 13, 5, 28, 20, 12

Proses selanjutnya adalah ke-2 bagian (C_0dan D_0) digeser ke kiri (*Left Shift*) sepanjang 1 atau 2 bit, tergantung pada tiap putaran. Perputaran ini bersifat *wrapping* atau *round-shift*.

Hasil dari pergeseran C_0 dan D_0 akan didapatkan nilai dari C_1 dan C_2. Begitu seterusnya, hingga proses tersebut menghasilkan C_{16} dan D_{16}. Untuk mendapatkan kunci internal pertama (K_1), maka bit dari C_0 dan D_0 tadi dilakukan permutasi kompresi dengan menggunakan matriks PC-2.

Jadi setiap kunci K_i, mempunyai panjang 48 bit. Apabila proses pergeseran bit -bit dijumlahkan semuanya, maka jumlah seluruhnya sama dengan 28 putaran. Jumlah ini sama dengan jumlah bit pada C_i dan D_i. Oleh karena itu, setelah putaran ke-16 akan didapatkan kembali $C_{16} = C_0$ dan $D_{16} = D_0$. Gambar 2.5 akan memperlihatkan bagaimana cara pembangkitan kunci internal pada algoritma DES.



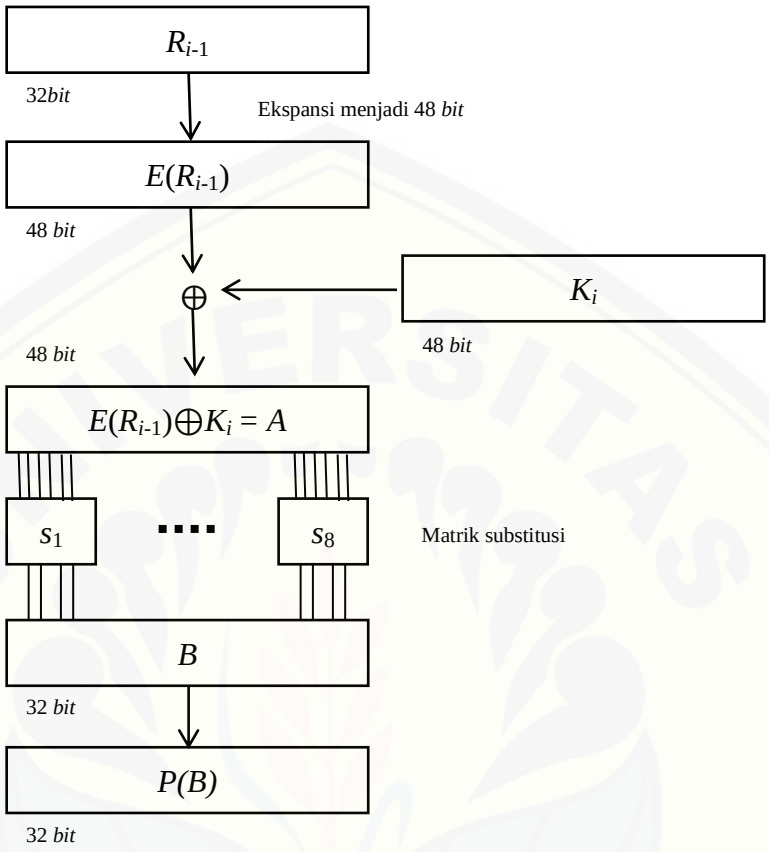
Gambar 2.5 Proses pembangkitan kunci internal DES

Proses *enciphering* terhadap blok *plaintext* dilakukan setelah permutasi awal. Setiap blok *plaintext* mengalami 16 kali putaran *enciphering*. Setiap putaran *enciphering* secara matematis dinyatakan sebagai:

$$L_i = R_{i-1} \quad (2.1)$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i) \quad (2.2)$$

Diagram fungsi *f* diperlihatkan pada Gambar 2.6.



Gambar 2.6 Diagram komputasi fungsi

E merupakan fungsi ekspansi yang memperluas blok R_{i-1} yang mempunyai panjang 32 bit menjadi blok 48 bit. Hasil ekspansi $E(R_{i-1})$, yang panjangnya 48 bit di-XOR-kan dengan K_i yang panjangnya 48 bit menghasilkan vektor A yang panjangnya juga 48 bit. Kemudian vektor A dikelompokkan menjadi 8 bagian, yang masing-masing bagian berisi 6 bit, dan merupakan masukkan dari proses substitusi.

Proses substitusi menggunakan 8 buah kotak-S (*S-box*). Kotak-S adalah matriks yang berisi substitusi sederhana yang memetakan satu atau lebih bit dengan satu atau lebih bit lainnya.

Dalam algoritma DES kotak-S yang digunakan adalah 6×4 *S-box* yang berarti menerima masukkan 6 bit dan menghasilkan keluaran 4 bit. Kelompok 6 bit pertama menggunakan S_1, kelompok 6 bit berikutnya menambahkan S_2 dan

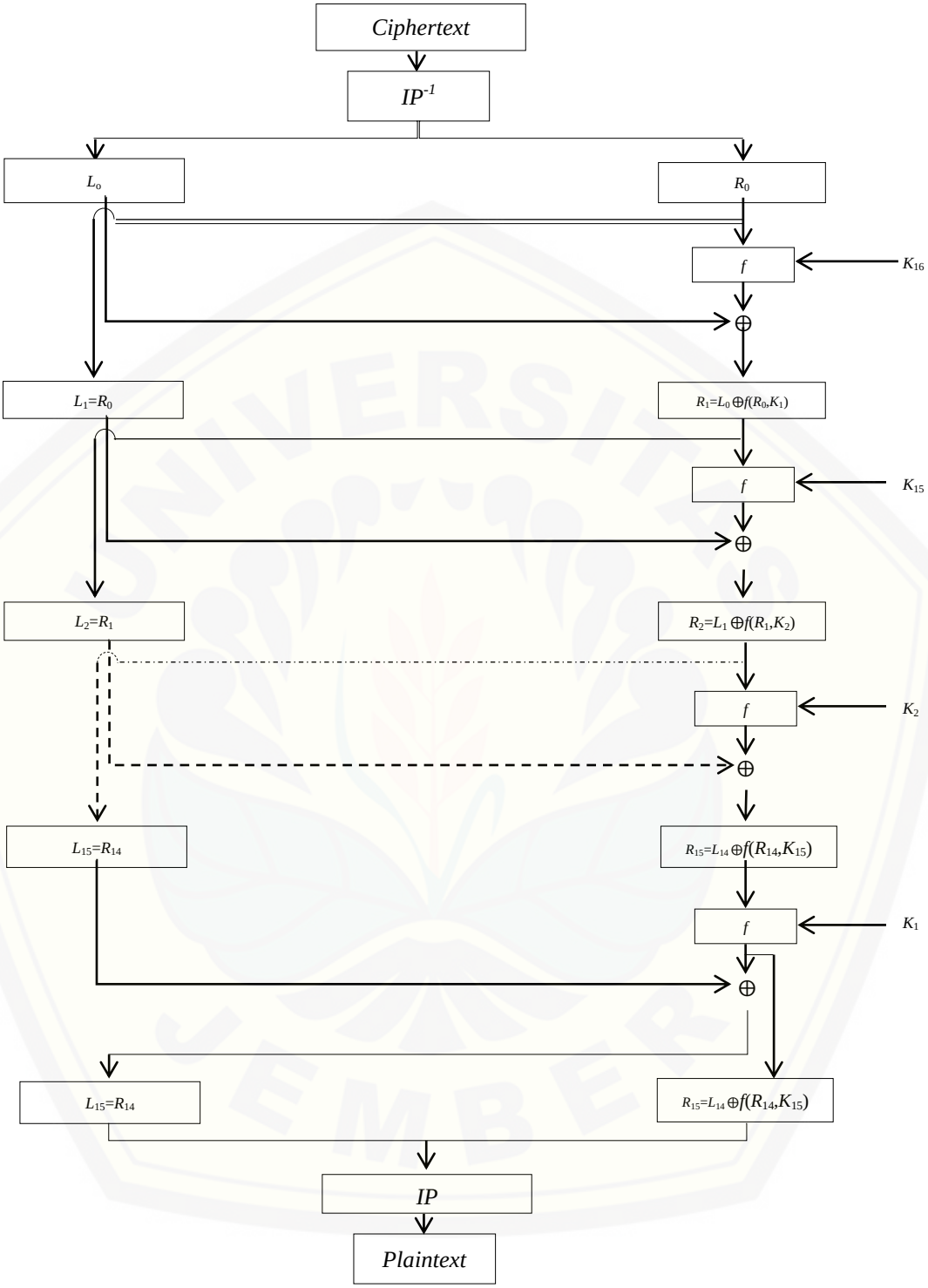
seterusnya sampai menggunakan S_8, sehingga secara keseluruhan akan menghasilkan 32 bit keluaran yang dinamakan dengan vektor B.

Setelah didapat vektor B, maka selanjutnya pada vektor B dilakukan proses permutasi, yang bertujuan untuk mengacak hasil proses substitusi kotak-S. Permutasi dilakukan dengan menggunakan matriks permutasi P (*P-box*). Keluarannya menghasilkan *P*(*B*) yang juga merupakan keluaran dari fungsi *f*. Proses selanjutnya yaitu bit-bit *P*(*B*) di-XOR-kan dengan L_{i-1} untuk mendapatkan R_i.

Proses selanjutnya yaitu permutasi terakhir yang dilakukan setalah 16 kali putaran terhadap gabungan dari blok kiri (*L*) dan blok kanan (*R*). Proses permutasi dilakukan dengan menggunakan matriks permutasi balikan (*invers initial permutation*) atau IP^{-1}.

2.2.2 Algoritma Dekripsi DES

Pada algoritma DES proses dekripsi dan enkripsinya menggunakan kunci yang sama. Proses dekripsi pada *ciphertext* merupakan proses kebalikan dari proses enkripsi. Jika pada proses enkripsi urutan kunci yang digunakan adalah $K_1, K_2, \dots, K_{16}$, maka untuk proses dekripsi urutan kunci yang digunakan adalah $K_{16}, K_{15}, \dots, K_1$. Masukkan awalnya adalah R_{16} dan L_{16} untuk *deciphering*. Blok R_{16} dan L_{16} diperoleh dengan mempermutasikan *ciphertext* dengan matriks permutasi IP^{-1} (Primartha, 2011). Skema proses dekripsi diperlihatkan pada Gambar 2.7.



Gambar 2.7 Skema Dasar Dekripsi Algoritma DES

2.3 Citra Digital

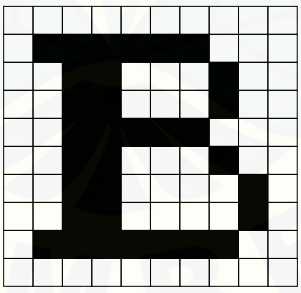
Pengolahan citra digital (*Digital Image Processing*) adalah sebuah disiplin ilmu yang mempelajari tentang teknik-teknik mengolah citra. Secara matematis, citra merupakan fungsi kontinyu (*continue*) dengan intensitas cahaya pada bidang dua dimensi. Agar dapat diolah dengan komputer digital, maka suatu citra harus dipresentasikan secara numerik dengan nilai-nilai diskrit. Repersentasi dari fungsi kontinyu menjadi nilai-nilai diskrit disebut digitalisasicitra (Sutoyo dkk, 2009).

Sebuah citra digital dapat diwakili oleh sebuah matriks dua dimensi $f(x, y)$ yang terdiri dari *M* kolom dan *N* baris, dimana perpotongan antara kolom dan baris disebut piksel (*pixel = picture element*) atau elemen terkecil dari sebuah citra.

Berdasarkan warna-warna penyusunnya, citra digital dapat dibagi menjadi tiga macam (Alasdair, 2004) yaitu:

2.3.1 Citra Biner

Citra yang setiap piksel hanya terdiri dari warna hitam atau putih, karena hanya ada dua warna untuk setiap piksel, maka hanya perlu 1 bit per piksel (0 dan 1) atau apabila dalam 8 bit (0 dan 255), sehingga sangat efisien dalam hal penyimpanan.



Gambar 2.8 Citra Biner

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	1	1	1	0	0	0
0	0	1	1	0	0	0	1	0	0
0	0	1	1	0	0	0	0	1	0
0	0	1	1	0	0	0	0	1	0
0	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0

Gambar 2.9 Representasi Citra Biner

2.3.2 Citra *Grayscale*

Citra yang setiap pikselnya mempunyai warna gradasi mulai dari putih sampai hitam (0 – 255). Rentang tersebut berarti bahwa setiap piksel dapat diwakili oleh 8 bit, atau 1 byte. Rentang warna pada citra *grayscale* sangat cocok digunakan untuk pengolahan file gambar.

Gambar 2.10 Citra *Grayscale* (abu-abu)

(sumber: http://www.situshewan.com/2014/08/jenis-ayam-hutan-yang-paling-langka.html)

2.3.3 Citra Berwarna

Citra yang nilai *piksel*-nya merepresentasikan warna tertentu. warna tersebut adalah merah (*Red*), hijau (*Green*) dan biru (*Blue*). Jika masing-masing warna memiliki range 0 - 255, maka totalnya adalah 255^3 = 16.581.375 (16 K) variasi

warna berbeda pada gambar, *Color image* ini terdiri dari tiga matriks yang mewakili nilai merah, hijau dan biru untuk setiap pikselnya.

Gambar 2.11 Citra Berwarna (RGB)

(sumber: http://www.situshewan.com/2014/08/jenis-ayam-hutan-yang-paling-langka.html)

2.4 Kode ASCII (*Standard Code for Information Interchange*)

Kode ASCII merupakan standar internasional dalam kode huruf dan simbol yang bersifat universal. Kode ASCII digunakan oleh komputer dan alat komunikasi lainnya untuk menunjukkan teks (Rachmawanto, 2010). Karakter kode 32 sampai 125 yang berisi huruf, digit, tanda baca dan beberapa simbol lain yang sebagian besar ditemukan di *keyboard* disebut ASCII *printable character*. ASCII *printable character* dapat dilihat pada tabel 2.1, untuk tabel secara lengkap dapat dilihat pada lampiran A.

Tabel 2.1 Karakter ASCII (*Standard Code for Information Interchange*)

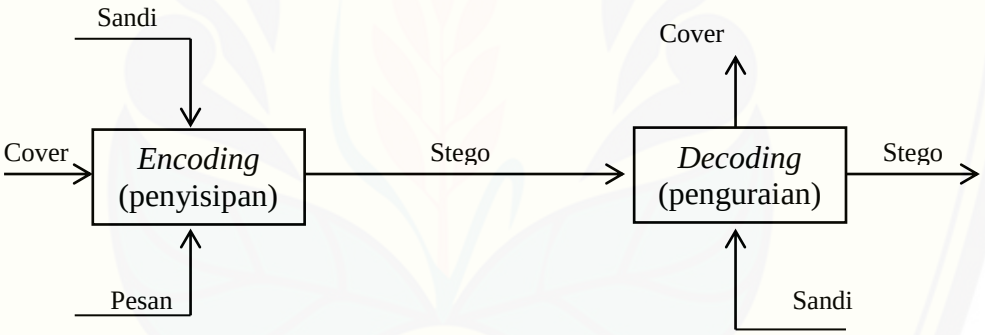
Desimal	Karakter	Desimal	Karakter	Desimal	Karakter
32	Space	64	@	96	`
33	!	65	A	97	a
34	“	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d

Desimal	Karakter	Desimal	Karakter	Desimal	Karakter
37	%	69	E	101	e
38	&	70	F	102	f
39	‘	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	’	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	-	127	Δ

2.5 Steganografi

Steganografi adalah seni komunikasi dengan menyembunyikan atau menyamarkan keberadaan pesan rahasia dalam suatu media penampungnya sehingga orang lain tidak menyadari adanya pesan di dalam media tersebut. Kata steganografi merupakan istilah yang berasal dari bahasa yunani yang berasal dari kata *steganos* (tertutup) dan *graphein* (tulisan). Lebih lengkapnya *steganos* dan *graphein* atau *graptos* memiliki arti menulis tulisan yang tersembunyi (Sutoyo, 2009).

Penyisipan pesan ke dalam media *covertext* dinamakan *encoding*, sedangkan ekstraksi pesan dari *stegotext* dinamakan *decoding*. Kedua proses ini memerlukan kunci rahasia agar hanya pihak yang berhak saja yang dapat melakukan penyisipan dan ekstraksi pesan. Berikut merupakan ilustrasi proses *Encoding* dan *Decoding* pada steganografi:



Gambar 2.12 Proses Steganografi

Di dalam steganografi citra digital, *hiddentext* atau *embedded message* merupakan teks yang akan disisipkan ke dalam *covertext* atau *cover object*, yaitu file yang digunakan sebagai media penampung pesan yang akan disisipkan. Hasil dari e*ncoding* atau *embedding* pesan ke dalam file citra akan dihasilkan *stegotext* atau *stego object* yang merupakan file yang berisikan pesan yang telah disisipkan.

2.5.1 *Least Significant Bit* (LSB)

Metode *Least Significant Bit* (LSB) bekerja dengan cara mengganti bit terakhir dari masing-masing piksel dengan pesan yang akan disisipkan, sehingga ukuran gambar tidak akan berubah. Namun pesan/data yang akan disisipkan

terbatas, sesuai dengan ukuran citra. Selain itu, kualitas citra akan sedikit berubah setelah mengalami proses penyisipan pesan. Namun perubahan yang tidak significant tersebut tidak nampak oleh mata (Arhami, 2005).

Cara kerja metode LSB dapat dijelaskan melalui contoh di bawah ini:

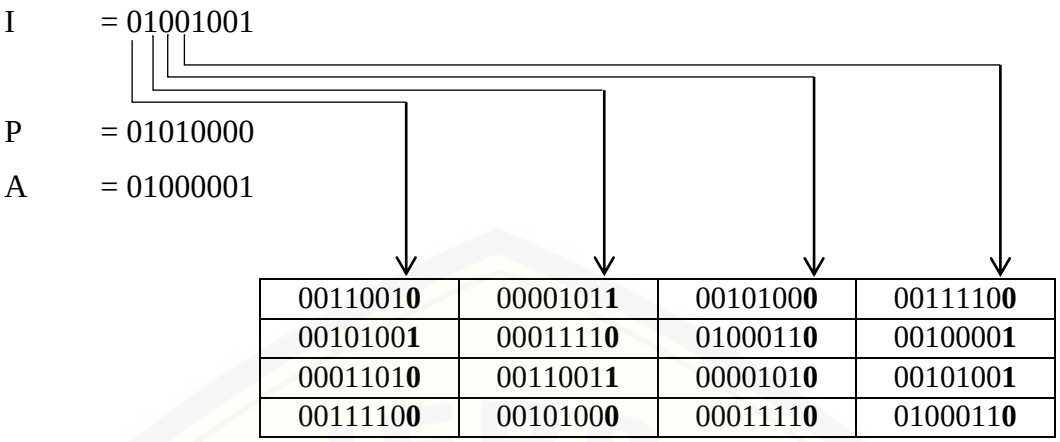
10	50	90	65
20	60	25	35
30	70	25	45
40	80	35	75

Gambar 2. 13 Matriks Citra Penampung

Pada Gambar 2.11 di atas nilai tiap-tiap piksel citra penampung adalah 10, 50, 90, 65, 20, 60, 25, 35, 30, 70, 15, 45, 40, 80, 35, 75. Kemudian nilai piksel di atas dilakukan pengubahan ke nilai *biner* menjadi: 00001010, 00110010, 01011010, 01000001, 00010100, 00111100, 00011001, 100011, 00011110, 01000110, 00001111, 00101101, 00101000, 01010000, 00100011, 01001011.

Diberikan *embed* berupa teks “IPA” lalu diubah ASCII (*Standard Code for Information Interchange*) menjadi 73, 80, 65 jika diubah kedalam biner menjadi 01001001, 01010000, 01000001.

Penyisipan dengan metode LSB dilakukan dengan mengganti 1 bit terakhir dari piksel citra penampung dengan 1 bitdari bit *embed* teks ‘IPA’.



Gambar 2. 14 Penyisipan Metode LSB

2.5.2 *Modified Least Significant Bit* (MLSB)

Metode *Modified Least Significant Bit* (MLSB) adalah suatu metode dari hasil pengembangan metode LSB yang sudah ada. Modifikasi pesan dengan algoritma MLSB dimana bit pesan yang seharusnya 1 karakter memiliki panjang 8 bit akan dimodifikasi menjadi 5 bit (31 desimal). Pada algoritma ini karakter dan angka direpresentasikan dalam 5 bit sebelum disisipkan kedalam citra asli dengan teknik LSB (*Least Significant Bit*).

Adapun cara kerja algoritma *Modified Least Significant Bit* (MLSB) adalah sebagai berikut (Zaher, 2011):

a. *Small Letter* (61h – 7Ah)

   Jika tipe karakter adalah *small letter* (a – z) , maka harus menempatkan 5 bit diawal yaitu 1Bh, kemudian konversikan menjadi 5 bit.

b. *Capital Letter* (41h – 5Ah)

   Jika tipe karakter adalah *capital letter* (A – Z), maka harus menempatkan 5 bit diawal yaitu 1Ch, kemudian konversi menjadi 5 bit.

c. *Space*

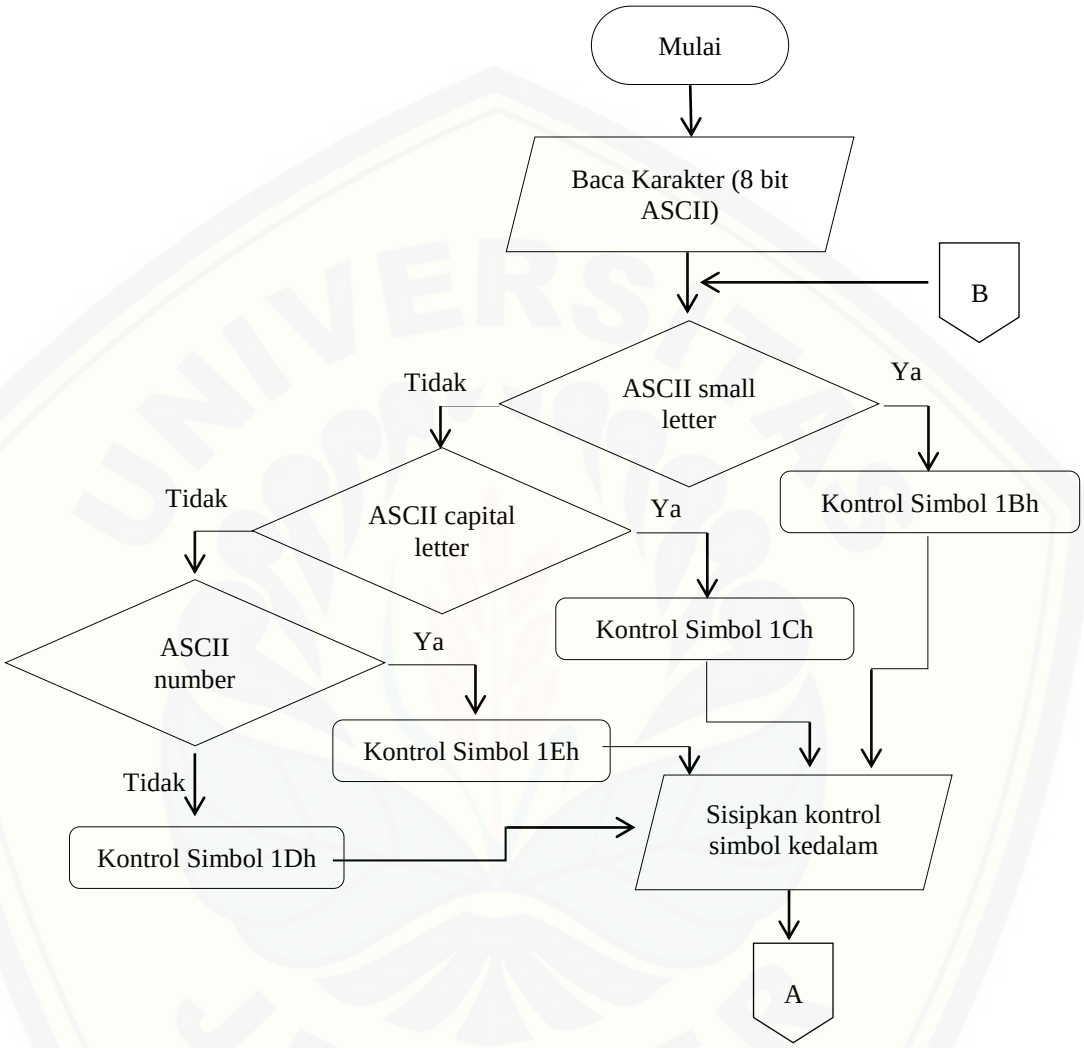
   Pada karakter spasi, maka harus menempatkan 5 bit yaitu 1Dh sebagai ganti dari 20h (Kode ASCII dari spasi) .

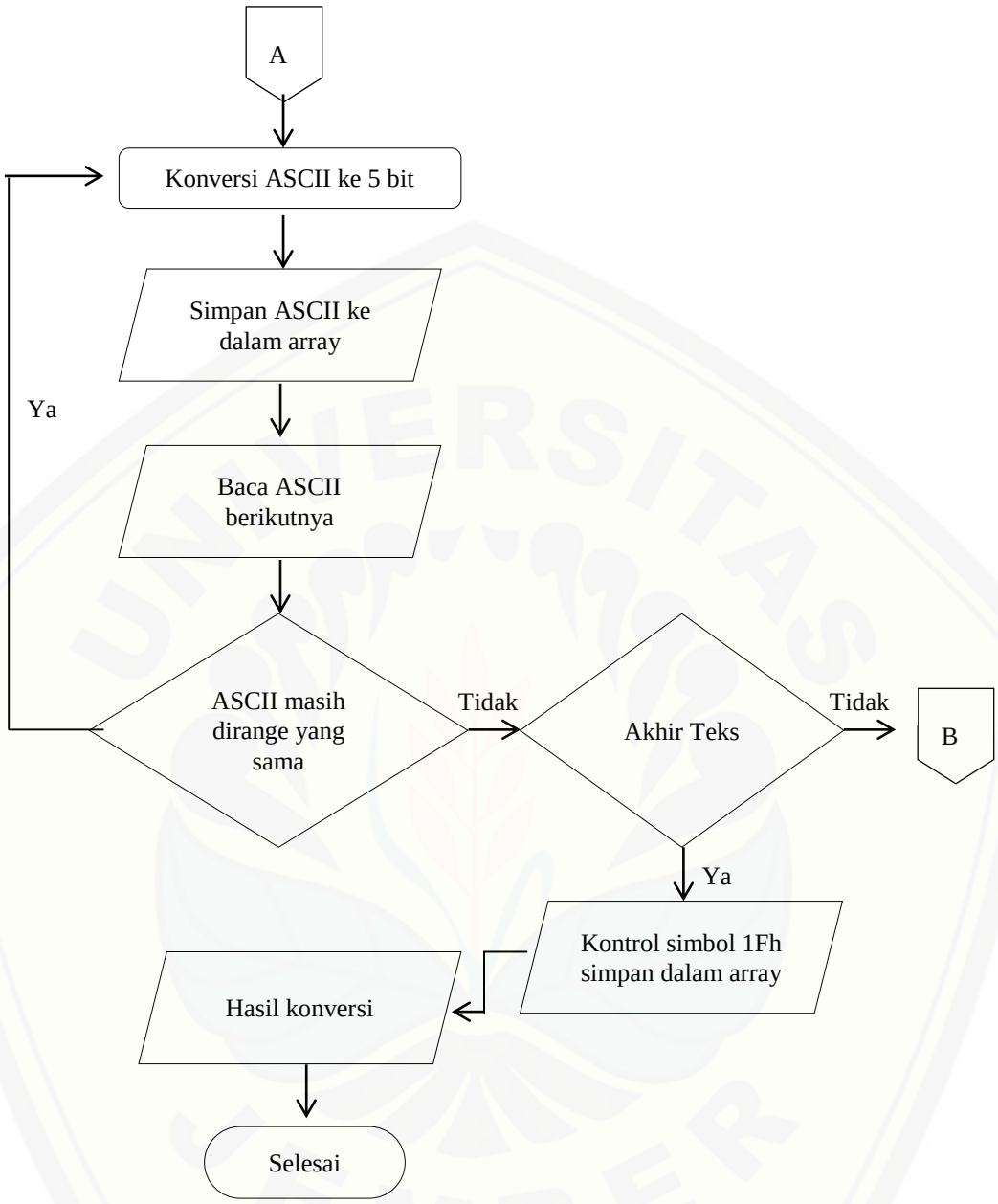
d. *Number* (20h – 39h)

   Pada tipe karakter yang berupa angka, harus menempatkan 1Eh diawal.

e. *End Text*

Untuk menandakan teks telah berakhir, maka harus disisipkan 1Fh. *Flowchart* dari metode MLSB (*Modified Least Significant Bit*) dapat dilihat pada Gambar 2.15:





Gambar 2.15 Flowchart *Modified Least Significant Bit* (MLSB)

BAB 3. METODE PENELITIAN

3.1 Data Penelitian

Data yang digunakan dalam penelitian ini adalah data teks berupa pesan rahasia. Karakter-karakter dari pesan tersebut merupakan karakter ASCII. Selain data teks, data yang digunakan adalah gambar (*image*) berekstensi bmp (*bitmap*), png dan gif.

3.2 Langkah-Langkah Penelitian

Langkah-langkah yang akan dilakukan pada penelitian ini, secara sistematis diuraikan sebagai berikut:

a. Studi Literatur

Tahap ini dilakukan untuk mempelajari beberapa teori terkait dengan penelitian yang dilakukan. Teori yang dipelajari dalam hal ini adalah teknik kriptografi algoritma *DES* (*Data Encryption Standart*) dan steganografi menggunakan metode *MLSB* (*Modified Least Significant Bit*) pada citra digital

b. Analisa Data

1) Enkripsi *plaintext* menggunakan algoritma *DES*

Data *plaintext* digeser sesuai kunci sehingga didapatkan *plaintext* baru. Data *plaintext* yang baru, kemudian di enkripsi dalam blok-blok 64 bit menjadi 64 bit data *ciphertext* menggunakan kunci 56 bit.

2) Konversi MLSB menjadi 5 bit

Pada Gambar 2.13 dijelaskan proses kerja dari metode *Modified Least Significant Bit* (MLSB) dimana data di konversi dari 8 bit menjadi 5 bit. Baca karakter (8 bit) lalu tentukan kode ASCII karakter untuk menentukan kontrol simbol sebagai penanda jenis karakter berikutnya.

3) Penyisipan LSB pada citra (*Embeding*)

Setelah mengkonversi biner dari 8 bit menjadi 5 bit, selanjutnya adalah menyisipkan *ciphertext* ke dalam citra dengan mengganti *bit* terakhir setiap nilai piksel dengan *bit-bit* dari *ciphertext*.

4) Ekstraksi LSB pada citra (*Decoding*)

Pesan yang telah disisipkan diekstrak menggunakan metode *Least Significant Bit*. Hasil dari langkah ini adalah *hiddentext* yang berupa *ciphertext*, dan jumlah karakter.

5) Konversi MLSB menjadi 8 bit

Pesan yang telah diekstrak menggunakan metode LSB, dikonversi kembali menggunakan metode MLSB sesuai dengan control simbolnya pada pada Gambar 2.15.

6) Dekripsi *Ciphertext* menggunakan algoritma *DES*

Ciphertext yang telah konversi menggunakan metode MLSB didekripsi menggunakan algoritma DES seperti pada Gambar 2.7.

c. Perancangan Program

Pada langkah ini menggunakan *software* Matlab 2009a dan melakukan perancangan desain GUI (*Graphic User Interface*) seperti tata letak tombol-tombol untuk tiap proses yang dibutuhkan serta tata letak *properties* pendukung program yang lainnya.

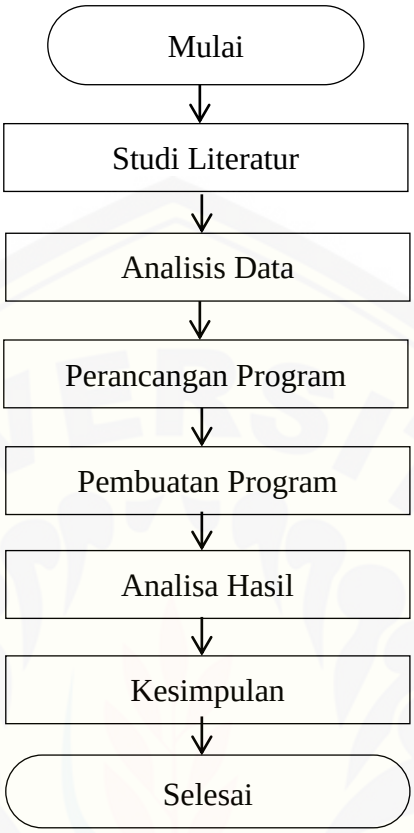
d. Pembuatan Program

Pembuatan program dilakukan berdasarkan konsep algoritma *Data Encryption Standart* untuk proses enkripsi dan dekripsi data teks dan citra digital hasil steganografi dengan metode *Modified Least Significant Bit.*

e. Kesimpulan

Mengambil kesimpulan dari penelitian yang dilakukan, yaitu menganalisis hasil sebelum dan sesudah citra disisipkan pesan dan dilakukan proses enkripsi. Selain itu, dilakukan analisis waktu komputasi untuk proses penyisipan dan ekstraksi pesan.

Berikut merupakan *flowchart* dari langkah-langkah penelitian:



Gambar 3.1 *Flowchart* Penelitian

BAB 5. PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian yang dilakukan, didapatkan beberapa kesimpulan sebagai berikut:

a. Penyisipan pesan dan ekstraksi pesan dipengaruhi oleh panjang *plaintext*. Semakin panjang *plaintext*, waktu komputasi yang dibutuhkan untuk mengisipkan dan mengekstraksi pesan semakin lama.
b. Penggunaan metode *Modified Least Significant Bit* dalam menyisipkan pesan dapat meningkatkan keamanan penyembunyian pesan rahasia. Karena metode *Modified Least Significant Bit* (MLSB) mengkonversi bilangan dengan panjang biner 8 bit menjadi 5 bit.

5.2 Saran

Saran yang diberikan kepada peneliti selanjutnya adalah:

a. Menerapkan metode MLSB kedalam algoritma kriptografi klasik maupun modern
b. Menerapkan steganografi pada media audio dan video.

DAFTAR PUSTAKA

Arhami, M., dan A. Desiani. 2005. *Pemrograman MATLAB*, Yogyakarta: ANDI.

McAndrew Alasdair, 2004, *An Introduction to Digital Image Processing with Matlab. Notes forSCM2511 Image Processing 1*, School of Computer Science and Mathematics Victoria University of Technology.

Muhendra, A.Z. 2016. *Implementasi Kriptografi Affine Cipher Pada Citra Digital Hasil Steganografi Metode Parity Coding Dengan Pseudocode Random Number Generator (PRNG)*. Tidak Diterbitkan. Jember: Jurusan Matematika FMIPA UNEJ.

Munir, Rinaldi. 2006. *Kriptografi*. Bandung: Penerbit Informatika.

Nathasia, N.D., Wicaksono, A.E. 2011.*Penerapan Teknik Kriptografi Stream-Cipher Untuk Pengaman Basis Data*. *Jurnal Basis Data*, **6** (1): 1-22.

Pabokory, F.N., Astuti, I.F. dan Kridalaksana, A.H. 2015. *Implementasi Kriptografi Pengamanan Data Pada Pesan Teks, Isi File Dokumen, Dan File Dokumen Menggunakan Algoritma Advanced Encryption Standard*. Jurnal Informatika Mulawarman, **10** (1): 20-31.

Primartha, R. 2011.Penerapan Enkripsi Dan Dekripsi File Menggunakan Algoritma *Data Encryption Standard* (DES). *Jurnal Sistem Informatika*, **3**(2): 371-387.

Rachmawanto, E. H. 2009. *Teknik Keamanan Data Menggunakan Kriptografi dengan Algoritma Vernam Cipher dan Steganografi Metode End of File (EOF)*. Tidak Diterbitkan. Skripsi. Semarang: Universitas Dian Nuswantoro.

Rahardjo, B. 2002. *Keamanan Sistem Informasi Berbasis Internet*, Bandung: PT Insan Komunikasi Indonesia,.

Rosdiana, G.T. 2015. *Pengkodean Citra Digital Hasil Steganografi dengan Metode Least Significant Bit untuk Data Teks Terenkripsi dengan Algoritma Hill Cipher*. Tidak Diterbitkan. Jember: Jurusan Matematika FMIPA UNEJ.

Sutoyo, T. 2009. *Teori Pengolahan citra digital*. Yogyakarta. Penerbit: ANDI.

Sutoyo. T.M., Edy, S., Vincent, Dwi N.O., Wijanarto. 2009. "Teori *Pengolahan Citra Digital"*, Andi Yogyakarta dan UDINUS Semarang.

Zaher, M.A. 2011. *Modified Least Significant Bit*. *Computer and Information Science*, **4** (1): 60-67.

LAMPIRAN

Lampiran A. Kode ASCII

Dec	Binary	Hex	ASCII	Dec	Binary	Hex	ASCII
1	00000001	01		129	10000001	81	•
2	00000010	02		130	10000010	82	‚
3	00000011	03		131	10000011	83	*ƒ*
4	00000100	04		132	10000100	84	„
5	00000101	05		133	10000101	85	…
6	00000110	06		134	10000110	86	†
7	00000111	07		135	10000111	87	‡
8	00001000	08		136	10001000	88	ˆ
9	00001001	09		137	10001001	89	‰
10	00001010	0A		138	10001010	8A	Š
11	00001011	0B		139	10001011	8B	‹
12	00001100	0C		140	10001100	8C	Œ
13	00001101	0D		141	10001101	8D	•
14	00001110	0E		142	10001110	8E	Ž
15	00001111	0F		143	10001111	8F	•
16	00010000	10		144	10010000	90	•
17	00010001	11		145	10010001	91	‘
18	00010010	12		146	10010010	92	’
19	00010011	13		147	10010011	93	“
20	00010100	14		148	10010100	94	”
21	00010101	15		149	10010101	95	•
22	00010110	16		150	10010110	96	–
23	00010111	17		151	10010111	97	—
24	00011000	18		152	10011000	98	˜
25	00011001	19		153	10011001	99	™
26	00011010	1A		154	10011010	9A	š

Dec	Binary	Hex	ASCII	Dec	Binary	Hex	ASCII
27	00011011	1B		155	10011011	9B	›
28	00011100	1C		156	10011100	9C	œ
29	00011101	1D		157	10011101	9D	•
30	00011110	1E	-	158	10011110	9E	ž
31	00011111	1F		159	10011111	9F	Ÿ
32	00100000	20	spasi	160	10100000	A0	
33	00100001	21	!	161	10100001	A1	¡
34	00100010	22	"	162	10100010	A2	¢
35	00100011	23	#	163	10100011	A3	£
36	00100100	24	$	164	10100100	A4	¤
37	00100101	25	%	165	10100101	A5	¥
38	00100110	26	&	166	10100110	A6	¦
39	00100111	27	'	167	10100111	A7	§
40	00101000	28	(	168	10101000	A8	¨
41	00101001	29	)	169	10101001	A9	©
42	00101010	2A	*	170	10101010	AA	ª
43	00101011	2B	+	171	10101011	AB	«
44	00101100	2C	,	172	10101100	AC	¬
45	00101101	2D	-	173	10101101	AD	
46	00101110	2E	.	174	10101110	AE	®
47	00101111	2F	/	175	10101111	AF	¯
48	00110000	30	0	176	10110000	B0	°
49	00110001	31	1	177	10110001	B1	±
50	00110010	32	2	178	10110010	B2	²
51	00110011	33	3	179	10110011	B3	³
52	00110100	34	4	180	10110100	B4	´
53	00110101	35	5	181	10110101	B5	µ
54	00110110	36	6	182	10110110	B6	¶
55	00110111	37	7	183	10110111	B7	·

Dec	Binary	Hex	ASCII	Dec	Binary	Hex	ASCII
56	00111000	38	8	184	10111000	B8	¸
57	00111001	39	9	185	10111001	B9	¹
58	00111010	3A	:	186	10111010	BA	º
59	00111011	3B	;	187	10111011	BB	»
60	00111100	3C	<	188	10111100	BC	¼
61	00111101	3D	=	189	10111101	BD	½
62	00111110	3E	>	190	10111110	BE	¾
63	00111111	3F	?	191	10111111	BF	¿
64	01000000	40	@	192	11000000	C0	À
65	01000001	41	A	193	11000001	C1	Á
66	01000010	42	B	194	11000010	C2	Â
67	01000011	43	C	195	11000011	C3	Ã
68	01000100	44	D	196	11000100	C4	Ä
69	01000101	45	E	197	11000101	C5	Å
70	01000110	46	F	198	11000110	C6	Æ
71	01000111	47	G	199	11000111	C7	Ç
72	01001000	48	H	200	11001000	C8	È
73	01001001	49	I	201	11001001	C9	É
74	01001010	4A	J	202	11001010	CA	Ê
75	01001011	4B	K	203	11001011	CB	Ë
76	01001100	4C	L	204	11001100	CC	Ì
77	01001101	4D	M	205	11001101	CD	Í
78	01001110	4E	N	206	11001110	CE	Î
79	01001111	4F	O	207	11001111	CF	Ï
80	01010000	50	P	208	11010000	D0	Đ
81	01010001	51	Q	209	11010001	D1	Ñ
82	01010010	52	R	210	11010010	D2	Ò
83	01010011	53	S	211	11010011	D3	Ó
84	01010100	54	T	212	11010100	D4	Ô

Dec	Binary	Hex	ASCII	Dec	Binary	Hex	ASCII
85	01010101	55	U	213	11010101	D5	Õ
86	01010110	56	V	214	11010110	D6	Ö
87	01010111	57	W	215	11010111	D7	×
88	01011000	58	X	216	11011000	D8	Ø
89	01011001	59	Y	217	11011001	D9	Ù
90	01011010	5A	Z	218	11011010	DA	Ú
91	01011011	5B	[	219	11011011	DB	Û
92	01011100	5C	\	220	11011100	DC	Ü
93	01011101	5D	]	221	11011101	DD	Ý
94	01011110	5E	^	222	11011110	DE	Þ
95	01011111	5F	_	223	11011111	DF	ß
96	01100000	60	`	224	11100000	E0	à
97	01100001	61	a	225	11100001	E1	á
98	01100010	62	b	226	11100010	E2	â
99	01100011	63	c	227	11100011	E3	ã
100	01100100	64	d	228	11100100	E4	ä
101	01100101	65	e	229	11100101	E5	å
102	01100110	66	f	230	11100110	E6	æ
103	01100111	67	g	231	11100111	E7	ç
104	01101000	68	h	232	11101000	E8	è
105	01101001	69	i	233	11101001	E9	é
106	01101010	6A	j	234	11101010	EA	ê
107	01101011	6B	k	235	11101011	EB	ë
108	01101100	6C	l	236	11101100	EC	ì
109	01101101	6D	m	237	11101101	ED	í
110	01101110	6E	n	238	11101110	EE	î
111	01101111	6F	o	239	11101111	EF	ï
112	01110000	70	p	240	11110000	F0	ð
113	01110001	71	q	241	11110001	F1	ñ

Dec	Binary	Hex	ASCII	Dec	Binary	Hex	ASCII
114	01110010	72	r	242	11110010	F2	ò
115	01110011	73	s	243	11110011	F3	ó
116	01110100	74	t	244	11110100	F4	ô
117	01110101	75	u	245	11110101	F5	õ
118	01110110	76	v	246	11110110	F6	ö
119	01110111	77	w	247	11110111	F7	÷
120	01111000	78	x	248	11111000	F8	ø
121	01111001	79	y	249	11111001	F9	ù
122	01111010	7A	z	250	11111010	FA	ú
123	01111011	7B	{	251	11111011	FB	û
124	01111100	7C	|	252	11111100	FC	ü
125	01111101	7D	}	253	11111101	FD	ý
126	01111110	7E	~	254	11111110	FE	þ
127	01111111	7F	□	255	11111111	FF	ÿ
128	10000000	80	€				

Lampiran B. *Script* AlgoritmaEnkripsi DES

```
function s=plain2cip(pesanawal, key);

% Proses pengacakan plaintext dengan cara menggeser biner --------
---------
for i=1:8
    psngeser(8*i-7:8*i)=[dec2bin(double(pesanawal(i)),8)];
end
for i=1:8
    keygeser(i)=mod(char(key(i)),64);
end
pesanawal_geser=keygeser;

for i=1:length(pesanawal_geser)
    psngeser(i+1,1:64)=[psngeser(i,pesanawal_geser(i)+1:64)
psngeser(i,1:pesanawal_geser(i))];
end
hsl_gsr=psngeser(length(pesanawal_geser)+1,:);
for i=1:8
    pesan(i)=[char(bin2dec(hsl_gsr(8*i-7:8*i)))];
end
% ----------------------------------------------------------------
---------

plain2bin=dec2bin(double(pesan),8); % Ubahlah plaintext kedalam
bentuk biner
key2bin=dec2bin(double(key),8); %Ubahlah key kedalam bentuk biner
% Tabel S-Box
S1=[14 4 13 1 2 15 11 8 3 10 6 12 5 9 0 7  ; 0 15 7 4 14 2 13 1 10
6 12 11 9 5 3 8 ; 4 1 14 8 13 6 2 11 15 12 9 7 3 10 5 0 ; 15 12 8
2 4 9 1 7 5 11 3 14 10 0 6 13];
S2=[15 1 8 14 6 11 3 4 9 7 2 13 12 0 5 10 ; 3 13 4 7 15 2 8 14 12
0 1 10 6 9 11 5 ; 0 14 7 11 10 4 13 1 5 8 12 6 9 3 2 15 ; 13 8 10
1 3 15 4 2 11 6 7 12 0 5 14 9];
S3=[10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8 ; 13 7 0 9 3 4 6 10 2 8
5 14 12 11 15 1 ; 13 6 4 9 8 15 3 0 11 1 2 12 5 10 14 7 ; 1 10 13
0 6 9 8 7 4 15 14 3 11 5 2 12];
S4=[7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15 ; 13 8 11 5 6 15 0 3 4 7
2 12 1 10 14 9 ; 10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4 ; 3 15 0 6
10 1 13 8 9 4 5 11 12 7 2 14];
S5=[2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9 ; 14 11 2 12 4 7 13 1 5
0 15 10 3 9 8 15 ; 4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14 ; 11 8 12
7 1 14 2 13 6 15 0 9 10 4 5 3];
S6=[12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11 ; 10 15 4 2 7 12 9 5 6 1
13 14 0 11 3 8 ; 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6 ; 4 3 2 12
9 5 15 10 11 14 1 7 6 0 8 13];
S7=[4 11 2 14 15 0 8 13 3 12 9 7 5 10 6 1 ; 13 0 11 7 4 9 1 10 14
3 5 12 2 15 8 6 ; 1 4 11 13 12 3 7 14 10 15 6 8 0 5 9 2 ; 6 11 13
8 1 4 10 7 9 5 0 15 14 2 3 12];
S8=[13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7 ; 1 15 13 8 10 3 7 4 12
5 6 11 0 14 9 2 ; 7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8 ; 2 1 14 7
4 10 8 13 15 12 9 0 3 5 6 11];
% tabel P-Box
p_box=[16 7 20 21 29 12 28 17 ; 1 15 23 26 5 18 31 10 ; 2 8 24 14
32 27 3 9 ; 19 13 30 6 22 11 4 25];
```

```
[mp_box np_box]=size(p_box);
trPBox=transpose(p_box);
%Tabel Indek Permutation (IP)
IP=[58 50 42 34 26 18 10 2 ; 60 52 44 36 28 20 12 4 ; 62 54 46 38
30 22 14 6 ; 64 56 48 40 32 24 16 8 ; 57 49 41 33 25 17 9 1 ; 59
51 43 35 27 19 11 3 ; 61 53 45 37 29 21 13 5 ; 63 55 47 39 31 23
15 7];
[m n]=size(IP);
k=0;
p2b=transpose(plain2bin); %transpose dr plain2bin
for a=1:m
    for b=1:n
        k=k+1;
        IPpesan(k)=p2b(IP(a,b));
    end
end

% Pecah bit pada IP(x) menjadi 2 bagian yaitu:
L(1,:)=IPpesan(1:length(IPpesan)/2);
R(1,:)=IPpesan((length(IPpesan)/2)+1:length(IPpesan));
k2b=transpose(key2bin); %transpose dr key2bin
PCmin1=[57 49 41 33 25 17 9 ; 1 58 50 42 34 26 18 ; 10 2 59 51 43
35 27 ; 19 11 3 60 52 44 36 ; 63 55 47 39 31 23 15 ; 7 62 54 45 38
30 22 ; 14 6 61 53 45 37 29 ; 21 13 5 28 20 12 4];
[m n]=size(PCmin1);
k=0;
for c=1:m
    for d=1:n
        k=k+1;
        CD(k)=k2b(PCmin1(c,d));
    end
end
%Pecah CD(k) menjadi dua bagian kiri dan kanan, sehingga menjadi
C(1,:)=CD(1:length(CD)/2);
D(1,:)=CD((length(CD)/2)+1:length(CD));
% Lakukan Pergeseran sesuai pada tabel left shift
left_shift=[1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1];
[m n]=size(C);
for e=1:length(left_shift)
    if left_shift(e)==1
        C(e+1,1:m*n)=[C(e,2:m*n) C(e,1)];
        D(e+1,1:m*n)=[D(e,2:m*n) D(e,1)];
    elseif left_shift(e)==2
        C(e+1,1:m*n)=[C(e,3:m*n) C(e,1:2)];
        D(e+1,1:m*n)=[D(e,3:m*n) D(e,1:2)];
    end
end
% C dan D digabungkan kembali
for f=2:17
    CD(f-1,:)=[C(f,:) D(f,:)];
end

% Setiap hasil putaran digabungkan kembali menjadi CiDi dan
diinput kedalam tabel Permutation Compression 2 (PC-2)
% dan terjadi kompresi data CiDi 56 bit menjadi CiDi 48 bit.
PCmin2=[14 17 11 24 1 5 ; 3 28 15 6 21 10 ; 23 19 12 4 26 8 ; 16 7
```

```
27 20 13 2 ; 41 52 31 37 47 55 ; 30 40 51 45 33 48 ; 44 49 39 56
34 53 ; 46 42 50 36 29 32];
trPCmin2=transpose(PCmin2); % Transpose dr PCmin2
[m n]=size(PCmin2);
k=0;

for p=1:16
    for q=1:m*n
        K(p,q)=CD(p,trPCmin2(q));
    end
end
tabel_ekspansi=[32 1 2 3 4 5 ; 4 5 6 7 8 9 ; 8 9 10 11 12 13 ; 12
13 14 15 16 17 ; 16 17 18 19 20 21 ; 20 21 22 23 24 25 ; 24 25 26
27 28 29 ; 28 29 30 31 32 1];
trTabel_eks=transpose(tabel_ekspansi);
[m n]=size(tabel_ekspansi);
iter=0;

for i=2:17
    for j=1:48
        ER(i,j)=R(i-1,trTabel_eks(j));
        A(i-1,j)=num2str(xor(str2num(ER(i,j)),str2num(K(i-1,j))));
    end
        for k=1:8
            brs(k,:)=bin2dec([A(i-1,(6*(k)-5)) A(i-1,(6*(k)))])+1;
            klm(k,:)=bin2dec([A(i-1,(6*(k)-4):(6*(k)-1))])+1;
        end
        B(i-1,1:4)=dec2bin([S1(brs(1),klm(1))],4);
        B(i-1,5:8)=dec2bin([S2(brs(2),klm(2))],4);
        B(i-1,9:12)=dec2bin([S3(brs(3),klm(3))],4);
        B(i-1,13:16)=dec2bin([S4(brs(4),klm(4))],4);
        B(i-1,17:20)=dec2bin([S5(brs(5),klm(5))],4);
        B(i-1,21:24)=dec2bin([S6(brs(6),klm(6))],4);
        B(i-1,25:28)=dec2bin([S7(brs(7),klm(7))],4);
        B(i-1,29:32)=dec2bin([S8(brs(8),klm(8))],4);
        for m=1:32
            PB(i,m)=B(i-1,trPBox(m));
        end
        L(i,:)=R(i-1,:);
    for n=1:32
        R(i,n)=num2str(xor(str2num(PB(i,n)),str2num(L(i-1,n))));
    end
end

tabel_IP_inv=[40 8 48 16 56 24 64 32 ; 39 7 47 15 55 23 63 31 ; 38
6 46 14 54 22 62 30 ; 37 5 45 13 53 21 61 29 ; 36 4 44 12 52 20 60
28 ; 35 3 43 11 51 19 59 27 ; 34 2 42 10 50 18 58 26 ; 33 1 41 9
49 17 57 25];

R16L16=[R(17,:) R(16,:)];
[m n]=size(tabel_IP_inv);
for i=1:m
    for j=1:n
        cipher(i,j)=R16L16(tabel_IP_inv(i,j));
    end
```

```
end
psn_dec=dec2hex(bin2dec(cipher));
[m n]=size(psn_dec);
k=0;
for i=1:m
    for j=1:n
        k=k+1;
        psn(k)=psn_dec(i,j);
    end
end
s=psn;
```

Lampiran C. *Script* Algoritma Dekripsi DES

```
function s=cip2plain(psn, key);

kk=0;
for aa=1:8
    for bb=1:2
        kk=kk+1;
        pesan(aa,bb)=psn(kk);
    end
end

plain2bin=dec2bin(hex2dec(pesan),8); % Ubahlah plaintext kedalam
bentuk biner
key2bin=dec2bin(double(key),8); %Ubahlah key kedalam bentuk biner
% Tabel S-Box
S1=[14 4 13 1 2 15 11 8 3 10 6 12 5 9 0 7 ; 0 15 7 4 14 2 13 1 10
6 12 11 9 5 3 8 ; 4 1 14 8 13 6 2 11 15 12 9 7 3 10 5 0 ; 15 12 8
2 4 9 1 7 5 11 3 14 10 0 6 13];
S2=[15 1 8 14 6 11 3 4 9 7 2 13 12 0 5 10 ; 3 13 4 7 15 2 8 14 12
0 1 10 6 9 11 5 ; 0 14 7 11 10 4 13 1 5 8 12 6 9 3 2 15 ; 13 8 10
1 3 15 4 2 11 6 7 12 0 5 14 9];
S3=[10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8 ; 13 7 0 9 3 4 6 10 2 8
5 14 12 11 15 1 ; 13 6 4 9 8 15 3 0 11 1 2 12 5 10 14 7 ; 1 10 13
0 6 9 8 7 4 15 14 3 11 5 2 12];
S4=[7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15 ; 13 8 11 5 6 15 0 3 4 7
2 12 1 10 14 9 ; 10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4 ; 3 15 0 6
10 1 13 8 9 4 5 11 12 7 2 14];
S5=[2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9 ; 14 11 2 12 4 7 13 1 5
0 15 10 3 9 8 15 ; 4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14 ; 11 8 12
7 1 14 2 13 6 15 0 9 10 4 5 3];
S6=[12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11 ; 10 15 4 2 7 12 9 5 6 1
13 14 0 11 3 8 ; 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6 ; 4 3 2 12
9 5 15 10 11 14 1 7 6 0 8 13];
S7=[4 11 2 14 15 0 8 13 3 12 9 7 5 10 6 1 ; 13 0 11 7 4 9 1 10 14
3 5 12 2 15 8 6 ; 1 4 11 13 12 3 7 14 10 15 6 8 0 5 9 2 ; 6 11 13
8 1 4 10 7 9 5 0 15 14 2 3 12];
S8=[13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7 ; 1 15 13 8 10 3 7 4 12
5 6 11 0 14 9 2 ; 7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8 ; 2 1 14 7
4 10 8 13 15 12 9 0 3 5 6 11];
% tabel P-Box
p_box=[16 7 20 21 29 12 28 17 ; 1 15 23 26 5 18 31 10 ; 2 8 24 14
32 27 3 9 ; 19 13 30 6 22 11 4 25];
[mp_box np_box]=size(p_box);
trPBox=transpose(p_box);
%Tabel Indek Permutation (IP)
IP=[58 50 42 34 26 18 10 2 ; 60 52 44 36 28 20 12 4 ; 62 54 46 38
30 22 14 6 ; 64 56 48 40 32 24 16 8 ; 57 49 41 33 25 17 9 1 ; 59
51 43 35 27 19 11 3 ; 61 53 45 37 29 21 13 5 ; 63 55 47 39 31 23
15 7];
[m n]=size(IP);
k=0;
p2b=transpose(plain2bin); %transpose dr plain2bin
for a=1:m
    for b=1:n
        k=k+1;
```

```
        IPpesan(k)=p2b(IP(a,b));
    end
end

% Pecah bit pada IP(x) menjadi 2 bagian yaitu:
L(1,:)=IPpesan(1:length(IPpesan)/2);
R(1,:)=IPpesan((length(IPpesan)/2)+1:length(IPpesan));
k2b=transpose(key2bin); %transpose dr key2bin
PCmin1=[57 49 41 33 25 17 9 ; 1 58 50 42 34 26 18 ; 10 2 59 51 43
35 27 ; 19 11 3 60 52 44 36 ; 63 55 47 39 31 23 15 ; 7 62 54 45 38
30 22 ; 14 6 61 53 45 37 29 ; 21 13 5 28 20 12 4];
[m n]=size(PCmin1);
k=0;
for c=1:m
    for d=1:n
        k=k+1;
        CD(k)=k2b(PCmin1(c,d));
    end
end
%Pecah CD(k) menjadi dua bagian kiri dan kanan, sehingga menjadi
C(1,:)=CD(1:length(CD)/2);
D(1,:)=CD((length(CD)/2)+1:length(CD));
% Lakukan Pergeseran sesuai pada tabel left shift
left_shift=[1 2 2 2 2 2 2 1 2 2 2 2 2 2 1 1];
[m n]=size(C);
for e=1:length(left_shift)
    if left_shift(e)==1
        C(e+1,1:m*n)=[C(e,m*n) C(e,1:m*n-1)];
        D(e+1,1:m*n)=[D(e,m*n) D(e,1:m*n-1)];
    elseif left_shift(e)==2
        C(e+1,1:m*n)=[C(e,m*n-1:m*n) C(e,1:m*n-2)];
        D(e+1,1:m*n)=[D(e,m*n-1:m*n) D(e,1:m*n-2)];
    end
end
% C dan D digabungkan kembali
for f=1:16
    CD(f,:)=[C(f,:) D(f,:)];
end

% Setiap hasil putaran digabungkan kembali menjadi CiDi dan
diinput kedalam tabel Permutation Compression 2 (PC-2)
% dan terjadi kompresi data CiDi 56 bit menjadi CiDi 48 bit.
PCmin2=[14 17 11 24 1 5 ; 3 28 15 6 21 10 ; 23 19 12 4 26 8 ; 16 7
27 20 13 2 ; 41 52 31 37 47 55 ; 30 40 51 45 33 48 ; 44 49 39 56
34 53 ; 46 42 50 36 29 32];
trPCmin2=transpose(PCmin2); % Transpose dr PCmin2
[m n]=size(PCmin2);
k=0;

for p=1:16
    for q=1:m*n
        K(p,q)=CD(p,trPCmin2(q));
    end
end
tabel_ekspansi=[32 1 2 3 4 5 ; 4 5 6 7 8 9 ; 8 9 10 11 12 13 ; 12
13 14 15 16 17 ; 16 17 18 19 20 21 ; 20 21 22 23 24 25 ; 24 25 26
```

```
27 28 29 ; 28 29 30 31 32 1];
trTabel_eks=transpose(tabel_ekspansi);
[m n]=size(tabel_ekspansi);
iter=0;

for i=2:17
    for j=1:48
        ER(i,j)=R(i-1,trTabel_eks(j));
        A(i-1,j)=num2str(xor(str2num(ER(i,j)),str2num(K(i-1,j))));
    end
        for k=1:8
            brs(k,:)=bin2dec([A(i-1,(6*(k)-5)) A(i-1,(6*(k)))])+1;
            klm(k,:)=bin2dec([A(i-1,(6*(k)-4):(6*(k)-1))])+1;
        end
        B(i-1,1:4)=dec2bin([S1(brs(1),klm(1))],4);
        B(i-1,5:8)=dec2bin([S2(brs(2),klm(2))],4);
        B(i-1,9:12)=dec2bin([S3(brs(3),klm(3))],4);
        B(i-1,13:16)=dec2bin([S4(brs(4),klm(4))],4);
        B(i-1,17:20)=dec2bin([S5(brs(5),klm(5))],4);
        B(i-1,21:24)=dec2bin([S6(brs(6),klm(6))],4);
        B(i-1,25:28)=dec2bin([S7(brs(7),klm(7))],4);
        B(i-1,29:32)=dec2bin([S8(brs(8),klm(8))],4);
        for m=1:32
            PB(i,m)=B(i-1,trPBox(m));
        end
        L(i,:)=R(i-1,:);
    for n=1:32
        R(i,n)=num2str(xor(str2num(PB(i,n)),str2num(L(i-1,n))));
    end
end

tabel_IP_inv=[40 8 48 16 56 24 64 32 ; 39 7 47 15 55 23 63 31 ; 38
6 46 14 54 22 62 30 ; 37 5 45 13 53 21 61 29 ; 36 4 44 12 52 20 60
28 ; 35 3 43 11 51 19 59 27 ; 34 2 42 10 50 18 58 26 ; 33 1 41 9
49 17 57 25];

R16L16=[R(17,:) R(16,:)];
[m n]=size(tabel_IP_inv);

for i=1:m
    for j=1:n
        plaintext(i,j)=R16L16(tabel_IP_inv(i,j));
    end
end
psn_dec=dec2hex(bin2dec(plaintext));
pesann=transpose(char(hex2dec(psn_dec))); %Plaintext

% Proses pengacakan plaintext dengan cara menggeser biner --------
---------
for i=1:8
    psngeser(8*i-7:8*i)=[dec2bin(double(pesann(i)),8)];
end
for i=1:8
    keygeser(i)=mod(char(key(i)),64);
end
```

```
pesanawal_geser=keygeser;
for i=1:length(pesanawal_geser)
    psngeser(i+1,1:64)=[psngeser(i,64-(pesanawal_geser(i)-1):64)
psngeser(i,1:64-pesanawal_geser(i))];
end
hsl_gsr=psngeser(length(pesanawal_geser)+1,:);
for i=1:8
    text(i)=[char(bin2dec(hsl_gsr(8*i-7:8*i)))];
end
% ------------------------------------------------------------------
---------
s=text;
```

Lampiran D. *Script* MLSB 5 bit

```
function s=mlsb5(psn)
for i=1:length(psn)
    if psn(i)>=65 && psn(i)<=90 %Capital Letter
        con_sym(i,:)='1C';
        limabit(i,:)=double(psn(i))-64;
    elseif psn(i)>=97 && psn(i)<=122 %Small Letter
        con_sym(i,:)='1B';
        limabit(i,:)=double(psn(i))-96;
    elseif psn(i)>=48 && psn(i)<=57 %Number Letter
        con_sym(i,:)='1E';
        limabit(i,:)=double(psn(i))-48;
    elseif psn(i)==' ' %Space
        con_sym(i,:)='1D';
    end
end
limabit=dec2hex(limabit,2);
A=con_sym;
[m n]=size(con_sym);
for i=1:m-1
    if (con_sym(i,:)=='1B' & con_sym(i+1,:)=='1B') |
(con_sym(i,:)=='1C' & con_sym(i+1,:)=='1C') |(con_sym(i,:)=='1E' &
con_sym(i+1,:)=='1E')
        A(i+1,:)='  '; %menjadikan 1 kontrol simbol yang sama,
kecuali '1D'
    end
end

[m n]=size(con_sym);
for i=1:m
    C(2*i-1,:)=A(i,:);
    C(2*i,:)=limabit(i,:);
end

[m n]=size(C);
k=0;
for i=1:m
    if C(i,:)~='  '
        k=k+1;
        D(k,:)=C(i,:); %menghlangkan Spasi
    end
end

[m n]=size(D);

for i=1:m-1
    if D(i,:)=='1D' & D(i+1,:)=='00'
        D(i+1,:)='  ';
    end
end

[m n]=size(D);
k=0;
for i=1:m
```

```
    if D(i,:) ~= '  '
        k=k+1;
        E(k,:)=D(i,:);
    end
end

[m n]=size(E);
E(m+1,:)='1F';

biner=dec2bin(double(hex2dec(E)));
[m n]=size(biner);
k=0;
for i=1:m
    for j=1:n
        k=k+1;
        biner1(k)=biner(i,j);
    end
end
s=biner1;
```

Lampiran E. *Script* MLSB 8 bit

```
function s=mlsb8(biner)
kotak(1:5,1:length(biner)/5)=' ';
for i=1:length(biner)
    kotak(i)=biner(i);
end
kotak=transpose(kotak);
A=dec2hex(bin2dec(kotak));
[m n]=size(A);
B(1:m,1:2)=' ';
for i=1:m
    if
A(i,:)=='1B'|A(i,:)=='1C'|A(i,:)=='1D'|A(i,:)=='1E'|A(i,:)=='1F'
        B(i,:)=A(i,:);
    end
end
[m n]=size(B);
for i=1:m-1
    if B(i,:)=='1B' & B(i+1,:)=='  ';
        B(i+1,:)='1B';
    elseif B(i,:)=='1C' & B(i+1,:)=='  ';
        B(i+1,:)='1C';
    elseif B(i,:)=='1D' & B(i+1,:)=='1D';
        B(i,:)='20';
        B(i+1,:)='20';
    elseif B(i,:)=='1D'
        B(i,:)='20';
    elseif B(i,:)=='1E' & B(i+1,:)=='  ';
        B(i+1,:)='1E';
    elseif B(i,:)=='1F' & B(i+1,:)=='  ';
        B(i+1,:)='1F';
    end
end
k=0;
for i=1:m
    if A(i,:)~=B(i,:) | (A(i,1)==B(i,1) & A(i,2)~=B(i,2)) |
(A(i,1)~=B(i,1) & A(i,2)==B(i,2))
        k=k+1;
        C(k,:)=A(i,:);
        D(k,:)=B(i,:);
    end
end
[m n]=size(C);
for i=1:m
    if D(i,:)=='1B'
        teks(i)=hex2dec(C(i,:))+96;
    elseif D(i,:)=='1C'
        teks(i)=hex2dec(C(i,:))+64;
    elseif D(i,:)=='1E'
        teks(i)=hex2dec(C(i,:))+48;
    end
end
s=char(teks);
```