IMPLEMENTASI ALGORITMA TWOFISH DENGAN KEY 128-BIT UNTUK ENKRIPSI DAN DEKRIPSI DATA TEKS

LAPORAN TUGAS AKHIR
Diajukan Sebagai Persyaratan Untuk Menyelesaikan
Program Pendidikan Diploma III

Pada

Program Studi : Teknik Informatika
Jurusan : Teknologi Informasi

Oleh:
M.Firly Feisal Abdigusna
NIM : 11615042

POLITEKNIK NEGERI SAMARINDA
2014

HALAMAN PENGESAHAN

IMPLEMENTASI ALGORITMA TWOFISH DENGAN KEY 128-BIT UNTUK ENKRIPSI DAN DEKRIPSI DATA TEKS

Disusun Oleh :

Nama : M.Firly Feisal Abdigusna
NIM : 11615042
Program Studi : Teknik Informatika
Jurusan : Teknologi Informasi

Laporan Tugas Akhir ini telah diterima dan disahkan untuk memenuhi salah satu syarat memperoleh ijazah Diploma III
Jurusan Teknologi Informasi
Politeknik Negeri Samarinda

Mengesahkan :

Pembimbing I	Pembimbing II
Abdul Najib, S.Kom., M.Cs	**Bedi Suprapty, S.Kom., M.Kom**
NIP. 19711121 200112 1 002	**NIP. 19781210 200212 1 002**

Mengetahui,
Direktur Politeknik Negeri Samarinda

Ir. H. Ibayasid, M.Sc
NIP. 19590303 198903 1 002

HALAMAN PERSETUJUAN

IMPLEMENTASI ALGORITMA TWOFISH DENGAN KEY 128-BIT UNTUK ENKRIPSI DAN DEKRIPSI DATA TEKS

Disusun Oleh :

Nama : M.Firly Feisal Abdigusna
NIM : 11615042
Program Studi : Teknik Informatika
Jurusan : Teknologi Informasi

Laporan Tugas Akhir ini telah disidangkan pada tanggal 07 Agustus 2014 di Jurusan Teknologi Informasi Politeknik Negeri Samarinda

Menyetujui :

Ketua Penguji / Penguji I	Pendamping
Didi Susilo Budi Utomo, ST., M.Sc	**Abdul Najib, S.Kom., M.Cs**
NIP. 19661109 199103 1 004	**NIP. 19711121 200112 1 002**
Penguji III	**Penguji II**
Rihartanto, ST	**Farandika Metandi, BCompSc., MM., M.Cs**
NIP. 19711205 200312 1 001	**NIP. 19870308 200801 1 002**

Mengetahui,
Ketua Jurusan Teknologi Informasi

M.F. Andrijasa, S.Kom., M.Kom
NIP. 19760116 200112 1 003

ABSTRAK

MUHAMMAD FIRLY FEISAL ABDIGUSNA. NIM 11615042 Konsentrasi Teknik Informatika – Jurusan Teknologi Informasi – Politeknik Negeri Samarinda, Samarinda, tahun 2013. (Impementasi Algoritma Twofish dengan Key 128-bit untuk Enkripsi dan Dekripsi Data Teks).

Pada saat ini, Penyimpanan data pada media digital sudah banyak digunakan. Dari data dapat diperoleh informasi yang bermanfaat, Tetapi tidak semua pihak berhak untuk mengetahui informasi yang bersifat rahasia. Oleh karena itu, penyimpanan data yang bersifat rahasia tersebut harus dilakukan dengan cara khusus. Salah satu caranya adalah menggunakan Kriptografi. Pada penelitian ini akan di usulkan Kriptografi dengan Algoritma Twofish 128 bit.

Penelitian ini bertujuan untuk membangun sistem enkripsi dan dekripsi untuk data text dengan menggunakan algoritma twofish.

Kata Kunci : Algoritma, Twofish, Ekripsi. Dekripsi.

KATA PENGANTAR

Seribu juta ucapan terindah yang pernah ada di bumi ini tak akan pernah bisa mewakili rasa syukur ke hadirat Allah SWT atas segala curahan Rahman dan Rahim-Nya yang selalu mengiringi di setiap perjalanan makhluk-Nya bernama manusia. Terselesaikannya Laporan Tugas Akhir ini hanyalah satu tapakan kecil perjalanan panjang yang masih harus ditempuh dengan tetap berpegang pada pertolongan-Nya.

Laporan Tugas Akhir ini adalah salah satu prasyarat pengerjaan Tugas Akhir sebagai muara akhir Program D3 Konsentrasi Teknik Informatika Jurusan Teknologi Informasi Politeknik Negeri Samarinda Judul Tugas Akhir yang akan dikerjakan adalah *Impementasi Algoritma Twofish dengan Key 128-bit untuk Enkripsi dan Dekripsi Data Teks.*

Ucapan terima kasih disampaikan kepada semua pihak yang secara langsung maupun tidak langsung memberikan dukungan dan bantuannya hingga terselesaikannya Tugas Akhir ini. Ucapan terima kasih khusus disampaikan kepada :

1. Bapak M.F. Andrijasa, S.Kom.,M.Kom, selaku Ketua Jurusan Teknologi Informasi Politeknik Negeri Samarinda.
2. Bapak Abdul Najib, S.Kom., M.CS selaku Pembimbing 1
3. Bu Bedi Suprapty S.Kom., M.Kom selaku Pembimbing 2
3. Bapak Karyo Budi Utomo, ST., MT., selaku Koordinator Kelompok Riset Dosen.
4. Semua rekan-rekan sesama mahasiswa Jurusan Teknologi Informasi.

Saran yang bersifat membangun sangat diharapkan untuk kesempurnaan Laporan Tugas Akhir ini.

Semoga tapakan perjalanan kecil ini menjadi sumber inspirasi positif bagi perjalanan berikutnya untuk dapat lebih menebar manfaat bagi semua makhluk di muka bumi ini.

Samarinda, 2014

Penulis

DAFTAR ISI

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi dalam bidang telekomunikasi menjadikan orang semakin sering melakukan pengiriman data melalui internet. Kegiatan tersebut sangat beresiko, karena internet merupakan media umum yang rentan akan terjadinya penyusupan dan pencurian informasi terhadap aliran data oleh pihak yang tidak berhak. Saat ini hampir setiap sistem komputer terkoneksi dengan jaringan internet. Sistem *sharing* data dan akses jarak jauh menyebabkan masalah keamanan menjadi salah satu kelemahan komunikasi data.

Berdasarkan kenyataan tersebut, perlu ada suatu pengamanan informasi pada saat pengiriman informasi. Untuk melakukan ini ada suatu cara yang biasa disebut *kriptografi* / penyandian data. Dalam penelitian ini akan mencoba mengimplementasikan suatu cabang ilmu matematika yang disebut dengan kriptografi. Dengan adanya sebuah kriptografi yang meliputi proses enkripsi dan dekripsi maka pesan, data, maupun informasi yang telah di enkripsi tidak dapat dibaca oleh orang lain kecuali orang yang mengetahui kunci untuk mendeskripsikannya.

1.2. Perumusan Masalah

Dari uraian di atas maka perumusan masalah dapat disusun sebagai berikut:

1. Bagaimana langkah-langkah dalam proses enkripsi dan dekripsi pada algoritma *Twofish.*
2. Bagaimana merancang dan membuat aplikasi enkripsi dan dekripsi menggunakan algoritma twofish

1.3. Batasan Masalah

Ruang Lingkup

1. Implementasi aplikasi hanya untuk data teks.
2. Data Text yang digunakan meliputi ASCII Printable Character.
3. Panjang kunci sebesar 128-bit.

1.4. Tujuan Penelitian

Penelitian dalam Laporan Tugas Akhir ini bertujuan untuk membangun aplikasi enkripsi dan dekripsi untuk data text dengan Algoritma Twofish.

1.5. Manfaat Penelitian

Diharapkan hasil dari penelitian ini memiliki manfaat sebagai berikut :

1. Menjaga kerahasiaan data text dari pihak yang tidak berkepentingan.

2. Mengetahui lebih dalam tentang keamanan data dan bisa mencegah pencurian data-data penting.
3. Mempermudah pengembangan lebih lanjut agar kedepannya tidak hanya data text saja yang dapat di enkripsi maupun di dekripsi.

BAB II

TINJAUAN PUSTAKA

2.1. Kajian Ilmiah

Informasi merupakan aset yang memiliki nilai seperti aset lainnya. Sebagai aset, informasi perlu diamankan dari serangan. Sampai dengan saat ini, jaringan komputer menciptakan sebuah revolusi dalam penggunaan informasi. Orang yang berwenang dapat mengirim dan menerima informasi dari jarak menggunakan jaringan komputer. Agar aman maka informasi perlu disembunyikan dari akses yang tidak sah (kerahasiaan), dilindungi dari perubahan yang tidak sah (integritas), dan tersedia bagi pihak yang berwenang, bila diperlukan (ketersediaan). Selain aman pada saat disimpan dalam komputer, informasi juga harus terjamin kerahasiaannya ketika dikirimkan dari satu komputer ke komputer lain (Schneier, B., 1998).

Keamanan dan kerahasiaan sebuah data atau informasi dalam komunikasi dan pertukaran informasi sangatlah penting. Seringkali data atau informasi yang penting, dalam komunikasi dan pertukaran informasi kadang tidak sampai kepada penerima atau tidak hanya diterima oleh penerima tetapi juga oleh pihak lain yang melakukan pembajakan atau penyadapan. Hal ini membuat data atau informasi tersebut menjadi tidak berguna lagi dan lebih parahnya lagi kadang data atau

informasi tersebut oleh para pembajak digunakan untuk menjatuhkan pihak lain. Oleh karena itu kriptografi sangat dibutuhkan dalam menjaga kerahasiaan data atau informasi. (Prayudi dan Halik, 2005)

2.2. Konsep Dasar Kriptografi

Kata Kriptografi berasal dari bahasa Yunani Kryptos (tersembunyi) dan Graphien (menulis). Kriptografi merupakan seni dan ilmu untuk menjaga berita. Dimana kriptografi mempunyai dua bagian penting, yaitu enkripsi dan deskripsi. Enkripsi adalah proses penyandian dari pesan asli (plaintext) menjadi pesan yang tidak dapat diartikan (ciphertext). Sedangkan deskripsi sendiri berarti merubah pesan yang sudah disandikan (ciphertext) menjadi pesan aslinya (plaintext). Adapun algoritma matematis yang digunakan pada proses enkripsi yakin disebut cipher dan sistem yang memanfaatkan kriptografi untuk mengamankan sistem informasi disebut kriptosistem.

2.3. Tujuan Kriptografi

Kriptografi adalah studi teknis yang digunakan untuk mencapai beberapa tujuan antar lain : kerahasiaan (*confidentiality*), integritas data (*data integrity*), otentikasi (*authentication*), dan non-repudiation.

Kerahasiaan mengacu pada perlindungan informasi dari akses yang tidak sah. Pihak yang tidak diinginkan, yang disebut musuh harus tidak dapat mengakses materi komunikasi.

Integritas data menjamin bahwa informasi tidak dimanipulasi dengan cara yang tidak sah. Jika informasi diubah, semua pihak yang berkomunikasi dapat mendeteksi perubahan ini.

Metode otentikasi yang dipelajari dalam dua kelompok; otentikasi entitas dan otentikasi pesan. Otentikasi entitas adalah proses dimana dua pihak yang terlibat dalam komunikasi telah terjamin identitasnya.

Non-repudiation berarti bahwa penerima dapat membuktikan kepada semua orang bahwa pengirim memang mengirim pesan, dan pengirim tidak dapat mengklaim bahwa ia tidak mengenkripsi pesan.

2.4. Enkripsi dan Dekripsi

Ide dasar enkripsi adalah memodifikasi pesan dengan berbagai cara dan hanya penerima yang sah yang dapat merekonstruksi isinya. Sistem kripto diskrit dinyatakan dengan :

Plaintext, P. Merupakan pesan yang akan dienkripsi.

Ciphertext, *C*. Merupakan pesan yang sudah terenkripsi.

Cipher Key, *K*. Merupakan *passkey* yang digunakan dalam enkripsi dan dekripsi.

Transformasi enkripsi dan dekripsi, *E* dan *D*.

Prosedur enkripsi dinyatakan dengan :

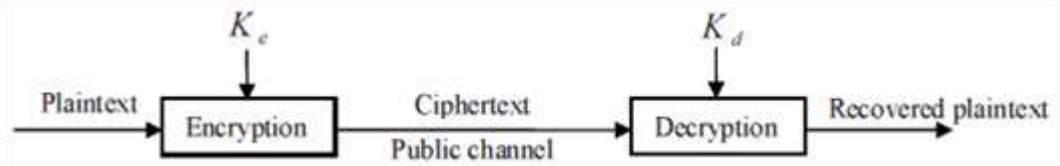
$$C = E_{Ke}(P) \qquad (2.1)$$

dimana *Ke* adalah *passkey*/kunci enkripsi dan *E* adalah fungsi enkripsi. Prosedur dekripsi dinyatakan dengan :

$$P = D_{Kd}(P) \qquad (2.2)$$

dimana K_d adalah *passkey* dekripsi dan *D* adalah fungsi dekripsi.

Keamanan *cipher* tergantung pada kunci dekripsi K_d karena musuh dapat memulihkan *plaintext* dari *ciphertext* jika dapat memperoleh K_d. Blok diagram proses enkripsi/dekripsi suatu *cipher* ditunjukkan dalam gambar berikut.



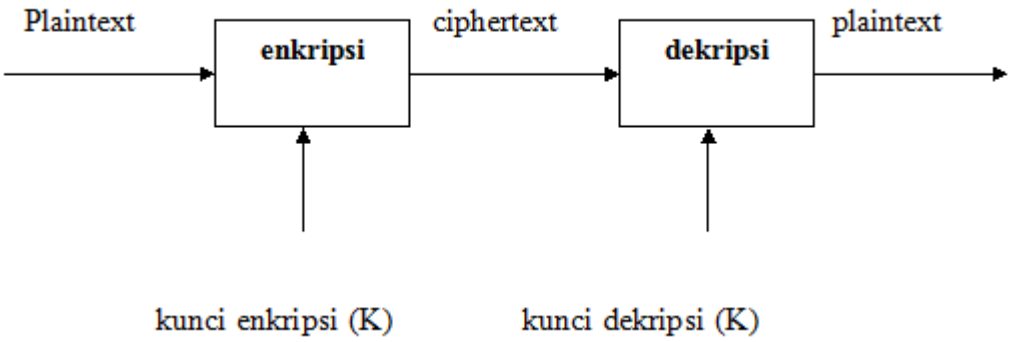
Gambar 1. Proses enkripsi dan dekripsi

2.5. Algoritma Kunci Simetris dan Asimetris

Algoritma kriptografi terbagi menjadi dua berdasarkan kuncinya yaitu algoritma kunci simetris dan algoritma kunci asimetris.

2.5.1 Algoritma Kunci Simetris

Algoritma simetris *(symmetric algorithm)* adalah suatu algoritma dimana kunci enkripsi yang digunakan sama dengan kunci dekripsi sehingga algoritma ini disebut juga sebagai *single-key algorithm.*



Gambar 2. Proses enkripsi dan dekripsi dengan algoritma kunci simetris

Sebelum melakukan pengiriman pesan, pengirim dan penerima harus memili suatu kunci tertentu yang sama untuk dipakai bersama, dan kunci ini haruslah rahasia bagi pihak yang tidak berkepentingan sehingga algoritma ini disebut juga algoritma kunci rahasia *(secret-key algorithm).*

Kelebihan :

1. Kecepatan operasi lebih tinggi bila dibandingkan dengan algoritma asimetrik.
2. Karena kecepatannya yang cukup tinggi, maka dapat digunakan pada sistem real time

Kelemahan :

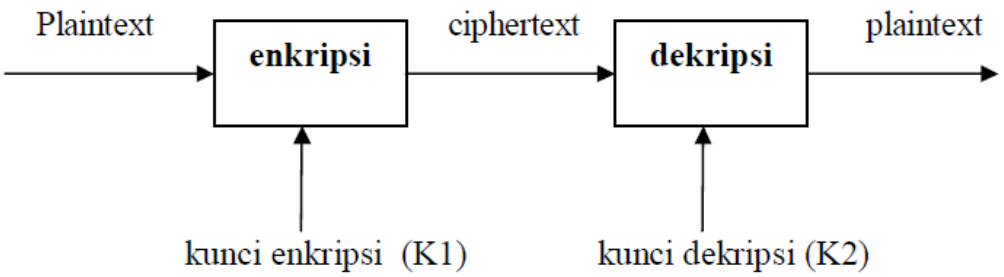
1. Untuk tiap pengiriman pesan dengan pengguna yang berbeda dibutuhkan kunci yang berbeda juga, sehingga akan terjadi kesulitan dalam manajemen kunci tersebut.
2. Permasalahan dalam pengiriman kunci itu sendiri yang disebut *"key distribution problem"*

Contoh algoritma : TwoFish, Rijndael

2.5.2. Algoritma Kunci Asimetris

Algoritma asimetris adalah suatu algoritma kriptografi dimana kunci untuk enkripsi yang digunakan berbeda dengan kunci dekripsi. Kunci enkripsi dinamakan sebagai kunci public yaitu kunci yang bebas diketahui oleh siapapun, sedangkan kunci dekripsi dinamakan kunci kunci privat yaitu kunci yang hanya boleh diketahui oleh penerima pesan. Contoh dari algoritma kriptografi asimetris adalah RSA, Elgamal, dan lain-lain.

Berikut adalah skema dari algoritma asimetris:



Gambar 3. Proses enkripsi dan dekripsi dengan algoritma kunci asimetris

Pada **Gambar 3.** merupakan diagram proses enkripsi dan dekripsi pada algoritma asimetris. Pada proses tersebut plainteks dienkripsi menggunakan kunci enkripsi (K1) sehingga menghasilkan chiperteks. Chiperteks didekripsikan kembali menggunakan kunci dekripsi (K2), artinya kunci pada saat pendekripsian berbeda dengan kunci ketika pengenkripsian dilakukan sehingga menghasilkan plainteks.

Kelebihan :

1. Masalah keamanan pada distribusi kunci dapat lebih baik
2. Masalah manajemen kunci yang lebih baik karena jumlah kunci yang lebih sedikit

Kelemahan :

1. Kecepatan yang lebih rendah bila dibandingkan dengan algoritma simetris Untuk tingkat keamanan sama,
2. Kunci yang digunakan lebih panjang dibandingkan dengan algoritma simetris.

2.6. Block Cipher dan Stream Cipher

Jika kita melihat berdasarkan ukuran serta format data yang akan diproses, maka algoritma kriptografi dapat dibagi menjadi dua bagian yang utama yaitu:

1. **Block Cipher,** algoritma kriptografi ini bekerja pada suatu data yang berbentuk blok/kelompok data dengan panjang data tertentu (dalam beberapa byte), jadi dalam sekali proses enkripsi atau dekripsi data yang masuk mempunyai ukuran yang sama (Twofish).
2. **Stream cipher,** algoritma yang dalam operasinya bekerja dalam suatu pesan berupa bit tunggal atau terkadang dalam suatu byte, jadi format data berupa aliran dari bit untuk kemudian mengalami proses enkripsi dan dekripsi.

Pada algoritma penyandian blok *(block cipher)*, plainteks yang masuk akan diproses dengan panjang blok yang tetap yaitu *n*, namun terkadang jika ukuran data ini terlalu panjang maka dilakukan pemecahan dalam bentuk blok yang lebih kecil. Jika dalam pemecahan dihasilkan blok data yang kurang dari jumlah data dalam blok maka akan dilakukan proses *padding* (penambahan beberapa bit).

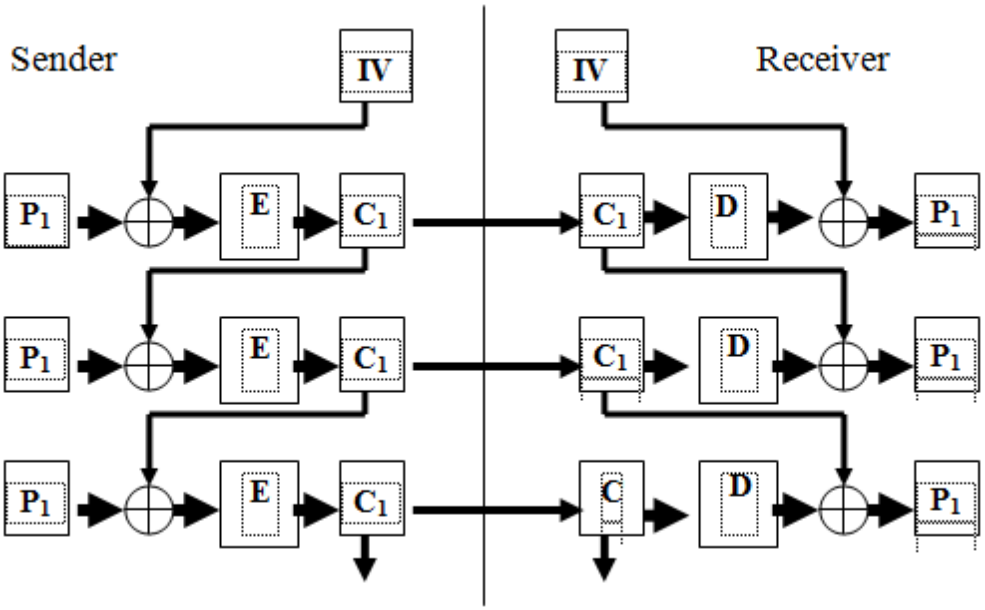
2.7. Mode Operasi Cipher Block Chaining (CBC)

Mode ini menerapkan mekanisme umpan-balik (*feedback*) pada sebuah blok, yang dalam hal ini hasil enkripsi blok sebelumnya di-umpan-balikkan ke dalam enkripsi blok yang *current*.

Caranya, blok plainteks yang *current* di-XOR-kan terlebih dahulu dengan blok cipherteks hasil enkripsi sebelumnya, selanjutnya hasil peng-XOR-an ini masuk ke dalam fungsi enkripsi.

Dengan mode *CBC*, setiap blok cipherteks bergantung tidak hanya pada blok plainteksnya tetapi juga pada seluruh blok plainteks sebelumnya.

Dekripsi dilakukan dengan memasukkan blok cipherteks yang *current* ke fungsi dekripsi, kemudian meng-XOR-kan hasilnya dengan blok cipherteks sebelumnya. Dalam hal ini, blok cipherteks sebelumnya berfungsi sebagai umpan-maju (*feedforward*) pada akhir proses dekripsi.



Gambar 4. Skema enkripsi dan dekripsi dengan mode CBC

Secara matematis, enkripsi dengan mode *CBC* dinyatakan sebagai

$$C_i = E_K(P_i \oplus C_{i-1}) \quad (2.3)$$

dan dekripsi sebagai

$$P_i = D_K(C_i) \oplus C_{i-1} \quad (2.4)$$

Blok plainteks pertama menggunakan C_0 sebagai vektor awal (*initialization vector* atau IV). IV tidak perlu rahasia.

Langkah Enkripsi dengan mode CBC

a. P(i) akan di Xor kan dengan C(i – 1)
b. Selanjutnya hasil dari P(i) Xor C(i – 1) akan memasuki fungsi enkripsi
c. Hasil dari enkripsi ini adalah C(i)

Langkah Dekripsi dengan mode CBC

a. C(i) memasuki fungsi dekripsi
b. Setelah melewati fungsi dekripsi hasil tersebut akan di Xor kan dengan C(i-1)
c. Hasil dari dekripsi ini adalah P(i)

2.8. Twofish

Algoritma twofish diciptakan oleh Bruce Schneier, sebelumnya ia menciptakan algoritma blowfish dengan 64 bit *block chiper* dan kunci 128 bit. Twofish merupakan algoritma kunci simetris *block chiper* dengan blok masukan 128 bit dan kunci 128 bit, 192

bit, dan 256 bit. Schneier (1998: 3) menjelaskan bahwa Pada tahun 1972 dan 1974, *National of Standart* (yang sekarang bernama NIST) mengumumkan adanya standar enkripsi, yaitu DES yang sangat beralasan karena penggunaannya yang luas dan merupakan algoritma yang sangat sukses di dunia. Dalam proses perkembangannya ternyata key-key dalam DES dirasa terlalu pendek bagi keamanan komersial sehingga membuat gusar para kriptografer yang menginginkan proses algoritma yang "*closed door*". Akhirnya, NIST mengumumkan *Advanced Encryption Standard* (AES) pada tahun 1997 [NIST97a]. Salah satu kandidat AES adalah Twofish. Twofish memenuhi semua kriteria yang dibutuhkan NIST, yaitu 128-bit block, 128 bit, 192 bit dan 256 bit key (kata kunci), efisien pada *platform* manapun dan lain-lain, serta beberapa desain berat lainnya Twofish dapat melakukan:

1. Melakukan enkripsi data pada 285 siklus per block di atas Pentium Pro setelah menjalankan key setup 12700 siklus clock.
2. Melakukan enkripsi data pada 860 siklus per blok sdi atas Pentium Pro setelah menjalankan key setup 1250 siklus clock.
3. Melakukan enkripsi data pada 26500 siklus per block di atas sebuah 6805 smart card setelah mejalankan key setup 1750 siklus clock.

2.9. Algoritma Twofish

Algoritma Twofish menggunakan struktur sejenis Feistel dalam 16 putaran dengan tambahan teknik whitening terhadap input dan output. Teknik whitening sendiri adalah teknik melakukan operasi XOR terhadap materi kunci sebelum putaran pertama dan sesudah putaran akhir. Elemen di luar jaringan feistel normal yang terdapat dalam algoritma twofish adalah rotasi 1 bit. Plaintext dipecah menjadi empat kata 32-bit. Pada langkah input whitening terdapat *xored* dengan empat kata kunci. Selanjutnya diikuti oleh enam belas putaran. Pada setiap putaran, dua kata-kata pada sisi kiri digunakan sebagai masukan kepada fungsi g (Salah satu darinya diputar pada 8 bit pertama). Fungsi g terdiri dari empat *byte-wide S-Box keydependent*, yang diikuti oleh suatu langkah pencampuran linier berdasar pada suatu matriks *MDS*. Hasil kedua fungsi g dikombinasikan menggunakan suatu *Pseudo Hadamard Transform (PHT)*, dan ditambahkan dua kata kunci. Kedua hasil ini kemudian di-*XOR* ke dalam kata-kata pada sisi kanan (salah satunya diputar ke kanan 1 bit pertama, yang lainnya diputar ke kanan setelahnya). Yang kiri dan kanan dibelah dua kemudian ditukar untuk putaran yang berikutnya, pertukaran yang terakhir(putaran 16) dilakukan *undo swap*, dan yang empat kata di-*XOR* dengan lebih dari empat kata kunci untuk menghasilkan ciphertext. Secara formal,

16 byte *plaintext* p0,… p15 yang yang pertama dipecah menjadi 4 kata P0,…P3 dari 32 bit masing-masing menggunakan konvensi *little-endian.*

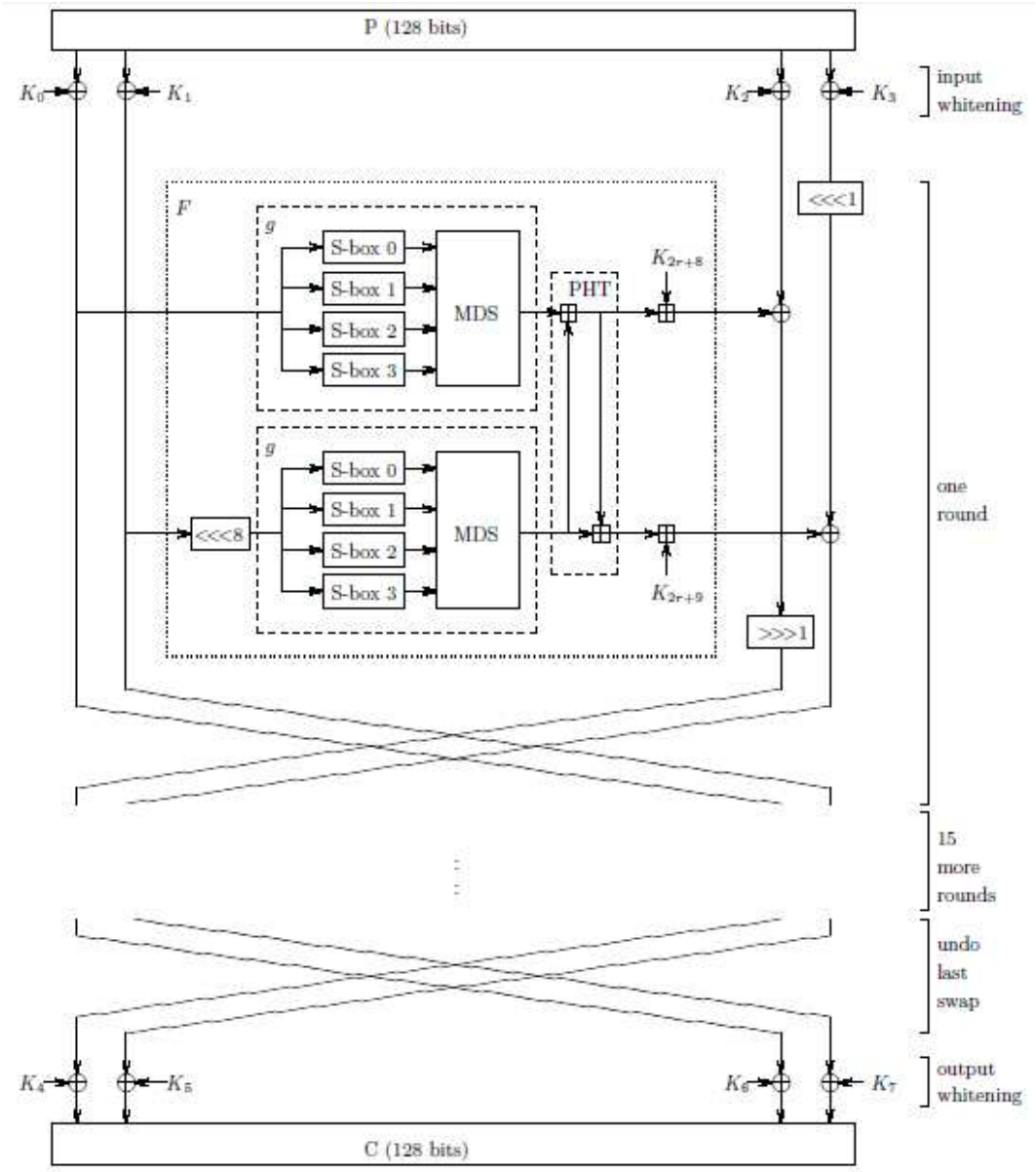
$$\mathbf{P(i)} = \sum_{j=0}^{3} P(4i+j).2^{8j} \quad (2.5)$$

i = 0,…,3

Di dalam langkah w*hitening*, kata-kata ini di-*XOR* dengan 4 kata dari kunci yang diperluas.

$$R_{0,i}=P \oplus K_i \quad (2.6)$$

i = 0,…,3



Gambar 5. Algoritma Twofish

(Schneier dkk, 1998: 6).

Pada setiap 16 putaran, dua kata pertama digunakan sebagai masukan kepada fungsi *F*, yang juga mengambil angka bulat itu sebagai masukan. Kata yang ketiga di-*XOR* dengan keluaran pertama F dan kemudian diputar ke kanan satu bit. Kata keempat

diputar ke kiri satu bit kemudian di-*XOR* dengan kata keluaran *F* Yang kedua . Akhirnya, keduanya saling ditukar menghasilkan persamaan :

$$(F_{r,0},F_{r,1}) = F(F_{r,0},F_{r,1},r) \quad (2.7)$$

$$R_{r+1,0} = ROR(R_{r,2} \oplus F_{r,0},1) \quad (2.8)$$

$$R_{r+1,1} = ROL(R_{r,3},1) \oplus F_{r,1} \quad (2.9)$$

$$R_{r+1,2} = R_{r,0} \quad (2.10)$$

$$R_{r+1,3} = R_{r,1} \quad (2.11)$$

untuk r = 0,…,15(putaran).

di mana *ROR* dan *ROL* adalah berfungsi memutar argumentasi pertama (32-bit kata) ke kanan / ke kiri dengan angka bit-bit diindikasikan dengan argumentasi keduanya. Langkah whitening keluaran membatalkan `pertukaran' putaran terakhir dan meng *XOR* kata-kata dengan 4 kata dari kunci yang diperluas.

$$C_i = R_{16,(i+2) \bmod 4} \oplus K_{i+4} \quad (2.12)$$

i = 0,…,3

Empat kata dari ciphertext kemudian ditulis sebagai 16 byte $C_0,\ldots,C_{15}$ menggunakan konversi little-endian untuk plaintext.

$$c_i = \left\lfloor \frac{C\lfloor i/4 \rfloor}{2^{8(i \bmod 4)}} \right\rfloor \bmod 2^8 \quad (2.13)$$

i = 0,…,15

(Schneier dkk, 1998: 7).

2.9.1. Whitening

Whitening merupakan teknik mengXORkan key material sebelum ronde pertama dan sesudah ronde terakhir. Dalam serangan terhadap Twofish, terbukti bahwa whitening secara substansial meningkatkan kesulitan menyerang chipper, dengan jalan menyembunyikan input spesifik untuk awal dan akhir ronde dari Twofish.

2.9.2. Fiestel Network

Sebuah Fietsel Network adalah metoda umum untuk mentransformasi suatu fungsi menjadi bentuk permutasi. Bagian paling fundamental dari Jaringan Fietsel adalah fungsi F: sebuah pemetaan key-dependent dari suatu input string menjadi output string. Dalam Twofish dilakukan Fietsel Network sebanyak 16 kali. Procedure Fietsel Network sebenarnya terdiri dari Input Whitening, S-boxes, Transformasi Pseudo Hadamard, Output dan Output Whitening.

2.9.3. S-Boxes

Sebuah S-box adalah operasi subsitusi table-driven non linear yang digunakan dalam block chipper. S-boxes bervariasi antara setiap ukuran input dan ukuran outputnya, dan bisa diciptakan secara random atau dengan algoritma.

Twofish menggunakan empat bijective, key-dependent dan 8-by-8-bit S-boxes. S-boxes ini dibuat menggunakan dua permutasi 8-by-8-bit dan material key.

2.9.4. MDS Matrix

Code Maximum Distance Separable(MDS) pada sebuah field adalah pemetaan linear dari a elemen field ke b elemen field, dan menghasilkan vector komposit a + b elemen, dengan property ketentuan bahwa jumlah minimum dari elemen bukan nol, pada setiap vector bukan nol paling sedikit b + 1. Dengan kata lain “Distance” adalah jumlah elemen yang berbeda antara dua vector yang berbeda yang dihasilkan oleh MDS paling sedikit adalah b + 1. Pemetaan MDS bisa direpresentasikan oleh matriks MDS yang terdiri dari a x b element. Twofish menggunakan matriks MDS 4 x 4 tunggal.

Berikut Matrix MDS Twofish :

$$\text{MDS} = \begin{pmatrix} 01 & EF & 5B & 5B \\ 5B & EF & EF & 01 \\ EF & 5B & 01 & EF \\ EF & 01 & EF & 5B \end{pmatrix}$$

Gambar 6. Matrix MDS

2.9.5. Transformasi Pseudo-Hadamard

Transformasi Pseudo-Hadamard (PHT) adalah operasi sederhana yang bekerja dengan cepat dalam software. Diberikan dua input, a dan b, dan PHT 32 bit didefinisikan sebagai :

$$A_0 = a + b \bmod 2^{32} \quad (2.14)$$

$$B_0 = a + 2b \bmod 2^{32} \quad (2.15)$$

Twofish menggunakan PHT 32 bit untuk melakukan mixing terhadap outputnya dari dua buah fungsi g 32 bit parallel. PHT ini dapat dieksekusi dalam dua opcode diatas kebanyakan microprocessor modern, termasuk keluarga Pentium.

2.9.6. Fungsi F

Pondasi dasar dari jaringan Feistel adalah fungsi *F*, yaitu suatu permutasi yang key-dependent terhadap nilai 64-bit. Fungsi *F* memerlukan tiga buah argument, dua input word R_0 dan R_1, dan bilangan bulat *r* yang digunakan untuk memilih subkey yang besesuaian. R_0 dilewatkan fungsi *g*, yang menghasilkan T_0. R_1 dirotasikan dalam sebuah PHT dan dua word dari key yang di-expand kemudian ditambahkan kepadanya.

$$T_0 = g(R_0) \quad (2.16)$$

$$T_1 = g(ROL(R_1,8)) \quad (2.17)$$

$$F_0 = (T_0 + T_1 + K_{2r+8}) \bmod 2^{32} \quad (2.18)$$

$$F_1 = (T_0 + 2T_1 + K_{2r+9}) \bmod 2^{32} \quad (2.19)$$

Dimana (F_0, F_1) merupakan hasil dari *F*, *ROL* adalah rotasi ke kiri terhadap R_1 sejauh 8 bit.

Fungsi *F* selalu non linear dan kemungkinan non surjektif, yaitu bahwa tidak semua output yang dimungkinkan berada dalam ruang output dapat terjadi semua.

2.9.7. Fungsi g

Fungsi g membentuk jantungnya Twofish. Kata masukan *X* dipecah menjadi empat byte. Masing-masing byte dijalankan melewati *S-box key-dependent.* Masing-masing S-box adalah *bijective,* mengambil 8 bit masukan, dan menghasilkan 8 bit keluaran. Ke empat hasil diinterpretasikan sebagai vektor yang panjangnya 4 di atas *GF* (2^8), dan dikalikan dengan yang matriks *MDS* 4x8 (menggunakan bidang *GF* (2^8) untuk perhitungannya). Untuk menghasilkan vektor diinterpretasikan sebagai 32-bit kata sebagai adalah hasil dari g :

$$x_i = \left\lfloor X / 2^{8i} \right\rfloor \bmod 2^8 \quad (2.20)$$

i = 0,…,3

$$y_i = s_i[x_i] \quad (2.21)$$

i = 0,…,3

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} \cdot & \cdots & \cdot \\ \vdots & \text{MDS} & \vdots \\ \cdot & \cdots & \cdot \end{pmatrix} \cdot \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} \quad (2.23)$$

$$Z = \sum_{i=0}^{3} z_i \cdot 2^{8i} \quad (2.24)$$

Di mana s_i adalah *S-Box key-dependent* dan *Z* adalah hasil dari *g*. Untuk merumuskan dengan baik, kita harus menetapkan koresponden antara nilai-nilai byte dan elemen-elemen bidang *GF* (2^8). Kita merepresentasikan *GF* (2^8) sebagai *GF*(2)[x]/v(x) di mana $v(x)=x^8+x^6+x^5+x^3+1 = 333$ adalah suatu polynomial primitif

dari 8 tingkat di atas GF(2). Unsur Bidang $a=\sum_{i=0}^{7} a_i x^i$ dengan $a_i \in GF(2)$ adalah dikenal dengan nilai byte $\sum_{i=0}^{7} a_i x^i$. Ini adalah beberapa pengertian pemetaan "alamiah"; penambahan di dalam $GF(2^8)$ berkorespondensi dengan suatu *XOR* dari bytes. dari rumus 2.23 maka di dapatkan :

Z0 = 01.y0 + EF.y1 + 5B.y2 + 5B.y3
Z1 = 5B.y0 + EF.y1 + EF.y2 + 01.y3
Z2 = EF.y0 + 5B.y1 + 01.y2 + EF.y3
Z3 = EF.y0 + 01.y1 + EF.y2 + 5B.y3

2.9.8. Key Schedule

Jadwal kunci harus menyediakan 40 kata dari kunci yang diperluas *K0,…,K39*, dan 4 buah *S-Box key-dependent* yang digunakan di dalam fungsi g. *Twofish* didefinisikan untuk kunci-kunci dengan panjang N= 128, N= 192, dan N= 256. Beberapa kunci yang lebih pendek dari 256 bit dapat digunakan oleh lapisannya dari nol hingga yang lebih besar yang didefinisikan sebagai panjang kunci. Kita mendefinisikan k=N/64. Kunci M terdiri dari 8k byte m0,…,m8k-1. Byte-byte yang pertama diubah ke dalam 2k kata dari 32 bit masing-masing :

$$M_i = \sum_{j=0}^{3} m_{(4i+j)} . 2^{8j} \qquad (2.24)$$

$i = 0,\ldots 2k\text{-}1$

dan kemudian ke dalam dua vektor kata dari panjang k :

$$M_e = (M_0, M_2, \ldots, M_{2k-2}) \tag{2.25}$$

$$M_o = (M_1, M_3, \ldots, M_{2k-2}) \tag{2.26}$$

Sepertiga vektor kata dari panjang *k* a juga diperoleh dari kunci itu. Hal ini dilakukan dengan mengambil byte-byte kunci di dalam kelompok 8, menginterpretasikannya sebagai vektor di atas *GF* (28), dan mengalikannya dengan matrils 4x8 yang diperoleh dari suatu kode *R*. Masing-masing hasil dari 4 byte kemudian diinterpretasikan sebagai suatu kata 32 bit.. Kata-kata ini menyusun vektor yang ketiga :

$$\begin{pmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \end{pmatrix} = \begin{pmatrix} \cdot \\ RS \\ \cdot \end{pmatrix} \cdot \begin{pmatrix} m_{Si} \\ m_{Si+1} \\ m_{Si+2} \\ m_{Si+3} \\ m_{Si+4} \\ m_{Si+5} \\ m_{Si+6} \\ m_{Si+7} \end{pmatrix} \tag{2.27}$$

$$S_i = \sum_{j=0}^{3} S_{i,j} \cdot 2^{8j} \tag{2.28}$$

untuk $i = 0, \ldots, k-1$, dan

$$S = (S_k\text{-}1, S_{k-2}, \ldots, S_0) \tag{2.29}$$

Catat bahwa daftar S kata-kata itu di dalam order “terbalik". Karena Untuk perkalian matriks RS, *GF* (2^8) diwakili oleh *GF* (2)[x]/w(x), di mana w(x)= x8+x6+x3+x2+1 adalah polynomial primitif derajat dari 8 tingkat di atas GF(2). Pemetaan antara nilai-nilai byte dan elemen-elemen *GF* (2^8) menggunakan defnisi yang sama sebagaimana yang digunakan untuk perkalian matriks *MDS*. Dalam pemetaan ini, matriks RS ditunjukkan sebagai berikut :

$$\text{RS} = \begin{pmatrix} 01 & A4 & 55 & 87\,5A & 58 & DB & 9E \\ A4 & 56 & 82 & F31E & C6 & 68 & E5 \\ 02 & A1 & FC & C1\,47 & AE & 3D & 19 \\ A4 & 55 & 87 & 5A\,58 & DB & 9E & 03 \end{pmatrix} \quad (2.30)$$

Ke tiga vektor *Me*, *Mo*, dan *S* ini membentuk basis dari jadwal kunci.

(Schneier dkk, 1998: 8).

2.9.9. Fungsi h

Pada **Gambar 7.** menunjukkan suatu ikhtisar fungsi h. Ini adalah suatu fungsi yang mengambil dua input yaitu 32-bit kata *X* dan sebuah daftar *L = (L0,…,Lk- 1*) dari kata-kata *X* 32-bit dengan panjang k dan menghasilkan satu kata pada outputnya. Ini adalah pekerjaan fungsi di dalam langkah-langkah k. Pada setiap langkah, empat byte ini masing-masing melintasi suatu *fixed S-Box*, dan di-*XOR* dengan sebuah byte yang diperoleh dari daftar. Akhirnya, byte-byte sekali lagi

digeser melewati sebuah fixed S-box dan empat byte itu dikalikan dengan matriks *MDS* seperti halnya dalam g. Lebih formal kita memisah-misahkan kata-kata itu ke dalam bytes :

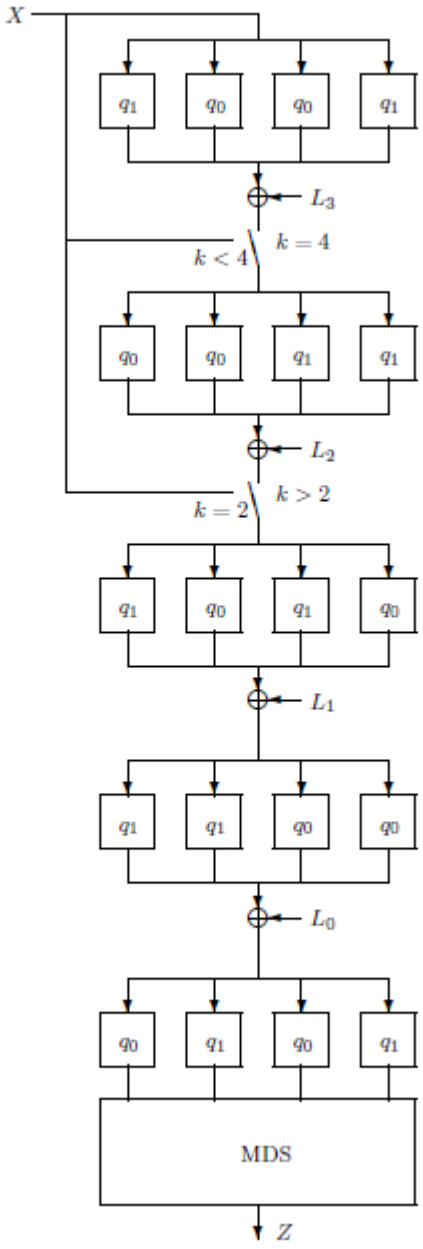
$$l_{i,j} = \lfloor L_i / 2^{8j} \rfloor \bmod 2^8 \quad (2.31)$$

$$x_j = \lfloor X / 2^{8j} \rfloor \bmod 2^8 \quad (2.32)$$

untuk i=0,…,k-1 dan j=0,…,3. Kemudian urutan subtitusi dan *XOR* diterapkan :

$$y_{k,j} = x_j \quad (2.33)$$

$j = 0,\ldots,3$



Gambar 7. Skema Fungsi h

(Schneier, 1998 : 9)

Untuk K = 2 Maka didapatkan

$$y_{k,0} = q0(q1(q1(x(0)) \text{ Xor } l(0, 0)) \text{ Xor } l(1, 0)) \quad (2.34)$$

$$y_{k,1} = q1(q1(q0(x(1)) \text{ Xor } l(0, 1)) \text{ Xor } l(1, 1)) \quad (2.35)$$

$$y_{k,2} = q0(q0(q1(x(2)) \text{ Xor } l(0, 2)) \text{ Xor } l(1, 2)) \quad (2.36)$$

$$y_{k,3} = q1(q0(q0(x(3)) \text{ Xor } l(0, 3)) \text{ Xor } l(1, 3)) \quad (2.37)$$

Di sini, *q0* dan *q1* ditetapkan permutasi di atas nilai 8-bit yang akan didefinisikan segera. Menghasilkan vektor yang merupakan perkalian matriks *MDS*, seperti halnya dalam fungsi *g* :

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} . \\ MDS \\ . \end{pmatrix} . \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} \quad (2.38)$$

$$Z = \sum_{i=0}^{3} z_i . 2^{8i} \quad (2.39)$$

di mana *Z* adalah hasil dari h. (Schneier dkk, 1998: 8).

2.9.10. S-Box Key-dependent

Sekarang kita dapat mendefinisikan *S-Box* dalam fungsi *g* dengan :

$$g(x) = h(X; S) \quad (2.40)$$

Hal itu untuk i= 0,…,3, *S-Box Key-Dependent si* dibentuk oleh pemetaan dari *xi* ke *yi* di dalam fungsi *h*, di mana daftar L sama dengan vektor *S* yang diperoleh dari kunci itu. (Schneier dkk, 1998: 10).

2.9.11. Kata-kata Kunci yang Diperluas K_j

Kata-kata dari kunci yang diperluas didefinisikan menggunakan fungsi *h* :

$$\rho = 2^{24} + 2^{16} + 2^8 + 2^0 = \&H01010101 \quad (2.41)$$

$$Ai = h(2_{i\,\rho}, M_e) \quad (2.42)$$

$$Bi = ROL(h((2_i + 1)\,\rho, Mo), 8) \quad (2.43)$$

$$K2i = (A_i + B_i) \bmod 2^{32} \quad (2.44)$$

$$K2i+1 = ROL((A_i + 2B_i) \bmod 2^{32}, 9) \quad (2.45)$$

Konstanta ρ digunakan untuk menduplikat byte yang mempunyai properti untuk i = 0,…,255, kata itu jika terdiri dari empat byte yang sama, masing-masing dengan nilai *i*. Fungsi h diberlakukan bagi kata-kata jenis ini. Untuk *Ai* nilai-nilai byte nya adalah 2i, dan argumentasi yang kedua dari h adalah *Me. Bi* dihitung dengan cara yang sama menggunakan 2i+1 sebagai byte nilai dan Mo sebagai argumentasi yang kedua, dengan suatu putaran ekstra di atas 8 bit. Nilai-nilai *Ai* dan *Bi* Dua dikombinasikan dalam *PHT*. Salah satu dari hasil itu selanjutnya diputar dengan 9 bit. Kedua hasil tersebut membentuk dua kata kunci yang diperluas. (Schneier dkk, 1998: 10).

2.9.12. Permutasi q_0 dan q_1

Permutasi q0 dan q1 adalah permutasi yang telah di tetapkan di atas nilai-nilai 8 bit. Mereka dibangun dari empat 4-bit permutasi yang masing-masing berbeda. Untuk nilai input didefinisikan sebagai x dan nilai output sebagai y:

$$a_0, b_0 = \lfloor x/16 \rfloor, x \bmod 16 \quad (2.46)$$

$$a_1 = a_0 \oplus b_0 \quad (2.47)$$

$$b_1 = a_0 \oplus \mathrm{ROR}_4(b_0, 1) \oplus 8a_0 \bmod 16 \quad (2.48)$$

$$a_2, b_2 = t_0[a_1], t_1[b_1] \quad (2.49)$$

$$a_3 = a_2 \oplus b_2 \quad (2.50)$$

$$b_3 = a_2 \oplus \mathrm{ROR}_4(b_2, 1) \oplus 8a_2 \bmod 16 \quad (2.51)$$

$$a_4, b_4 = t_2[a_3], t_3[b_3] \quad (2.52)$$

$$y = 16\, b_4 + a_4 \quad (2.53)$$

di mana *ROR4* adalah suatu fungsi yang serupa dengan *ROR*, yang merupakan putaran nilai 4-bit.

Pertama, byte dipecah menjadi dua bagian a_0 dan b_0. Masing-masing bagian kemudian melintasi *4-bitfixed S-Box*. Ini diikuti oleh yang lain. Akhirnya, dua bagian dikombinasikan kembali ke dalam satu byte. Untuk permutasi *q0 S-Box* 4-bit :

$$t_0 = [8\ 1\ 7\ D\ 6\ F\ 3\ 2\ 0\ B\ 5\ 9\ E\ C\ A\ 4] \quad (2.54)$$

$$t_1 = [E\ C\ B\ 8\ 1\ 2\ 3\ 5\ F\ 4\ A\ 6\ 7\ 0\ 9\ D] \quad (2.55)$$

$$t_2 = [B\ A\ 5\ E\ 6\ D\ 9\ 0\ C\ 8\ F\ 3\ 2\ 4\ 7\ 1] \quad (2.56)$$

$$t_3 = [D\ 7\ F\ 4\ 1\ 2\ 6\ E\ 9\ B\ 3\ 0\ 8\ 5\ C\ A] \quad (2.57)$$

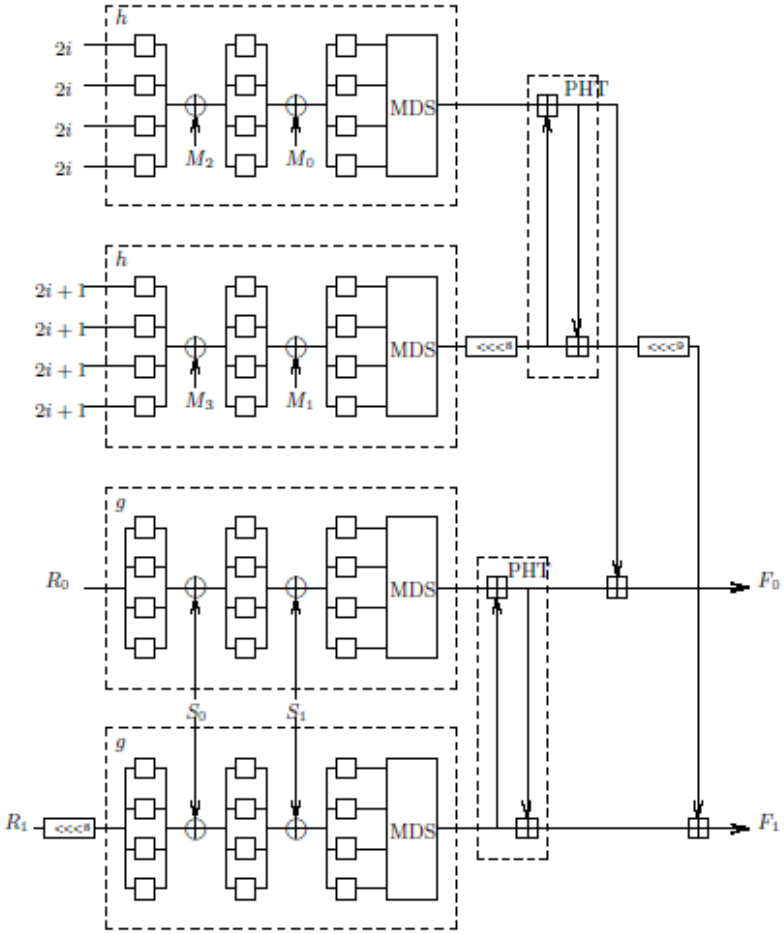
di mana masing-masing *S-Box* 4-Bit diwakili oleh daftar masukan yang menggunakan notasi hexadecimal. (untuk masukan 0,…,15 didaftarkan dalam order.) Dengan cara yang sama, untuk q1 *S-Box* 4-bit :

$$t_0 = [2\ 8\ B\ D\ F\ 7\ 6\ E\ 3\ 1\ 9\ 4\ 0\ A\ C\ 5] \quad (2.58)$$

$$t_1 = [1\ E\ 2\ B\ 4\ C\ 3\ 7\ 6\ D\ A\ 5\ F\ 9\ 0\ 8] \quad (2.59)$$

$$t_2 = [4\ C\ 7\ 5\ 1\ 6\ 9\ A\ 0\ E\ D\ 8\ 2\ B\ 3\ F] \quad (2.60)$$

$$t_3 = [B\ 9\ 5\ 1\ C\ 3\ D\ E\ 6\ 4\ 7\ F\ 2\ 0\ 8\ A] \quad (2.61)$$



Gambar 8. Satu Putaran Fungsi F (Kunci 128-bit)

(Schneier dkk, 1998: 11).

2.9.13. Padding dan Unpadding

Pada cipher block, input plainttext yang akan dienkripsi dibagi kedalam blok-blok. Tiap blok memiliki panjang yang sama sebesar n-bit. Untuk ukuran blok 128 bit, pembagian akan menyebabkan blok terakhir memiliki ukuran lebih kecil

atau sama dengan 128 bit. Cara mengatasi masalah tersebut adalah melakukan mekanisme padding.

Padding adalah proses penambahan bit-bit isian pada blok terakhir input plaintext yang akan dienkripsi.

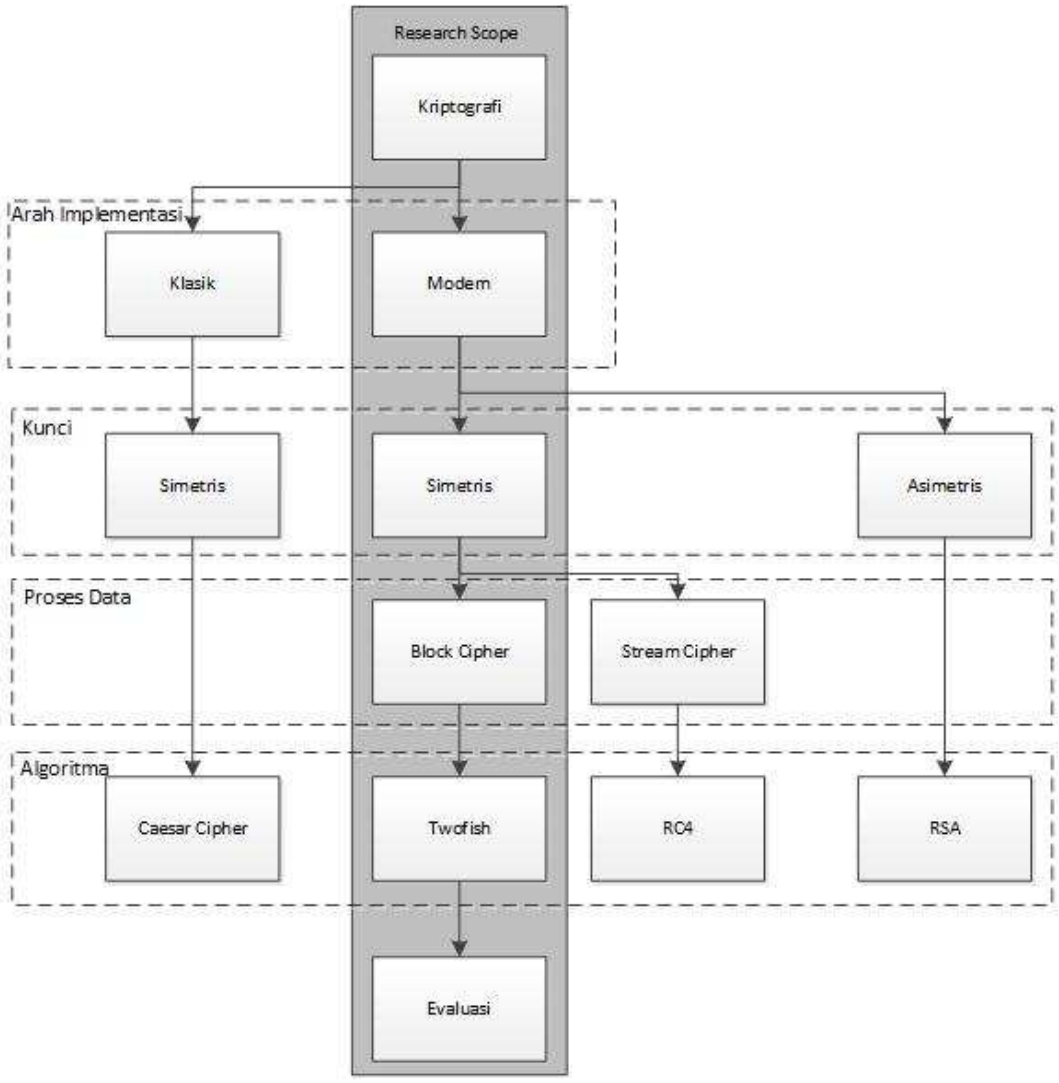
Apabila pada proses enkripsi dilakukan padding maka untuk proses dekripsi akan dilakukan proses unpadding, yaitu menghilangkan kembal bit-bit isian yang ditambahkan pada proses padding

BAB III

KERANGKA KONSEP PENELITIAN

3.1. Kerangka Konsep Penelitian

Diagram alir kerangka konsep penelitian ditunjukkan dalam gambar berikut.



Gambar 9. Diagram Alir Kerangka Konsep Penelitian

Data yang akan di enkripsi dan dekripsi terlebih dahulu melalui tahap penentuan arah implementasi Kriptografi. Dalam tahap ini, Kriptografi Modern yang dipilih dengan menggunakan kunci(*Passkey*) simetris dengan proses pengolahan data Block Cipher. Kemudian menentukan algoritma yang akan dipakai. Dalam TA ini Algoritma Twofish yang dipilih. Selanjutnya Algoritma tersebut di implementasikan dan di evaluasi.

3.2. Variabel Penelitian

Variabel penelitian yang akan dianalisis adalah Data Teks dan File (.txt) yang telah di enkripsi dan di dekripsi menggunakan Algoritma Twofish dengan kunci 128-bit.

3.3. Hipotesis

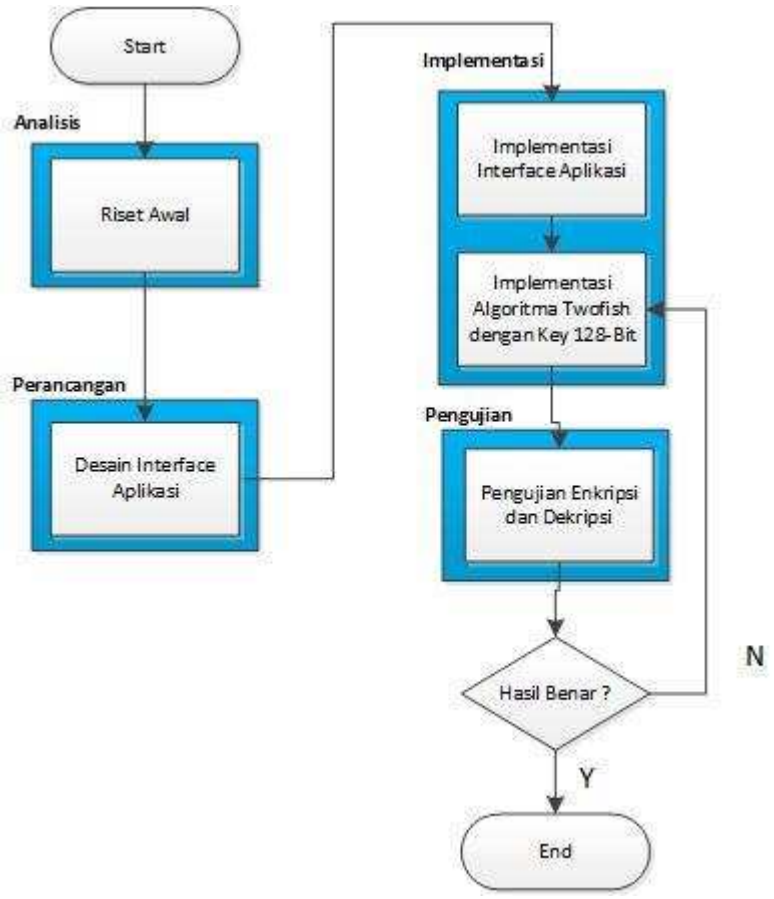
Dari kerangka konsep penelitian yang telah diuraikan maka hipotesis penelitian adalah dengan menggunakan Algoritma Twofish maka dapat menciptakan sistem enkripsi dan dekripsi.

BAB IV

METODOLOGI PENELITIAN

4.1. Metodologi Penelitian

Diagram alir metodologi penelitian ditunjukkan dalam gambar berikut.



Gambar 10. Diagram Alir Metodologi Penelitian

4.1.1. Riset Awal

Sebelum melakukan penelitian terlebih dahulu mempelajari segala hal yang terkait dengan topik penelitian. Bagian utama yang perlu dipelajari dan didalami adalah:

1. Konsep dasar Kriptografi.

   Mata kuliah yang terkait adalah Keamanan Jaringan

2. Algoritma Twofish.

   Mata kuliah yang terkait adalah Keamanan Jaringan

3. Algoritma dan Pemrograman.

   Mata kuliah yang terkait adalah Pemrograman Tingkat Lanjut

4.1.2. Desain Interface Aplikasi

Pada tahap ini Penulis akan membuat desain interface aplikasi secara abstrak.

4.1.3. Implementasi Interface Aplikasi

Pada tahap ini aplikasi yang telah di desain tersebut kemudian di implementasikan menggunakan.

4.1.4. Implementasi Algoritma Twofish dengan Key 128-bit

Di tahap ini algoritma tersebut di implementasikan menggunakan bahasa visual basic.

4.1.5. Pengujian Enkripsi dan Dekripsi

Di tahap ini aplikasi yang di buat selanjutnya di uji berdasarkan hasil enkripsi dan dekripsi. Jika hasil benar maka selesai tetapi jika hasil salah maka akan kembali lagi ke proses implementasi.

BAB V

ANALISIS DAN PERANCANGAN

Bab ini membahas tentang analisis dan perancangan sistem tentang pembuatan aplikasi enkripsi serta dekripsi data text dengan algoritma twofish. Analisis dan perancangan ini meliputi analisis sistem, perancangan proses, tahap pembuatan system dan perancangan antarmuka.

5.1. Analisis Sistem

Masalah yang diselesaikan dalam skripsi ini antara lain adalah menerapkan algoritma twofish digunakan untuk enkripsi dan dekripsi data text. Pada subbab ini dilakukan beberapa analisis yaitu deskripsi sistem dan perancangan proses sistem yang akan dibangun.

5.1.2. Deskripsi Sistem

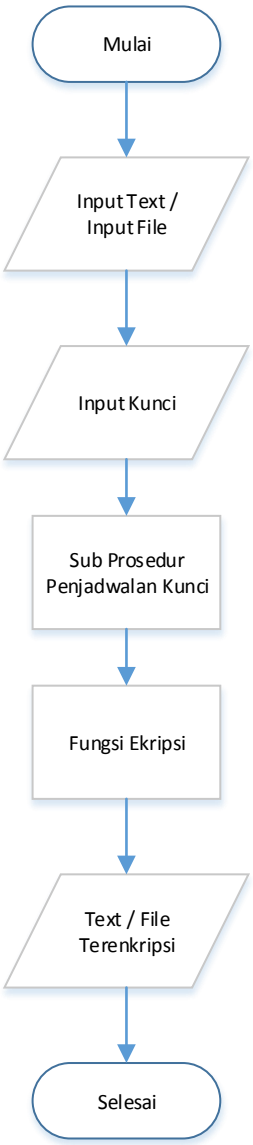
Subbab ini akan membahas mengenai deskripsi sistem yang dikerjakan pada tugas akhir ini. Tujuan pembuatan sistem ini adalah menerapkan algoritma untuk mengamankan data text sehingga data tersebut menjadi tidak dapat terbaca. Proses

utama pada aplikasi perangkat lunak ini adalah melakukan proses enkripsi dan dekripsi. Adapun proses dalam perangkat lunak ini sebagai berikut :

A. Melakukan enkripsi data text.

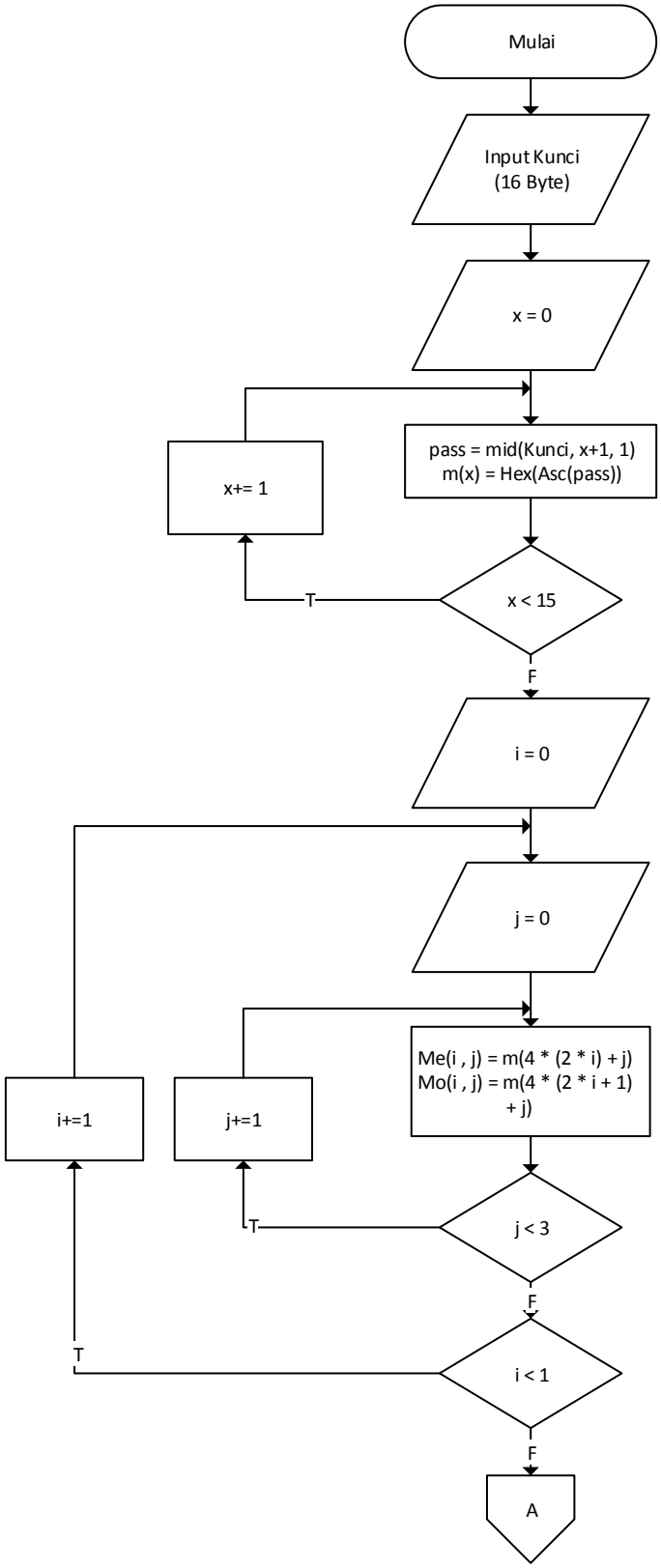
1. Pengguna memasukkan input berupa file atau text. File yang akan diinputkan berupa file ber ekstensi .txt.
2. Masukkan kunci untuk mengenkripsi.
3. Lakukan enkripsi file atau text yang telah diinputkan.
4. File atau text yang telah terenkripsi menjadi file atau text yang tidak bermakna.

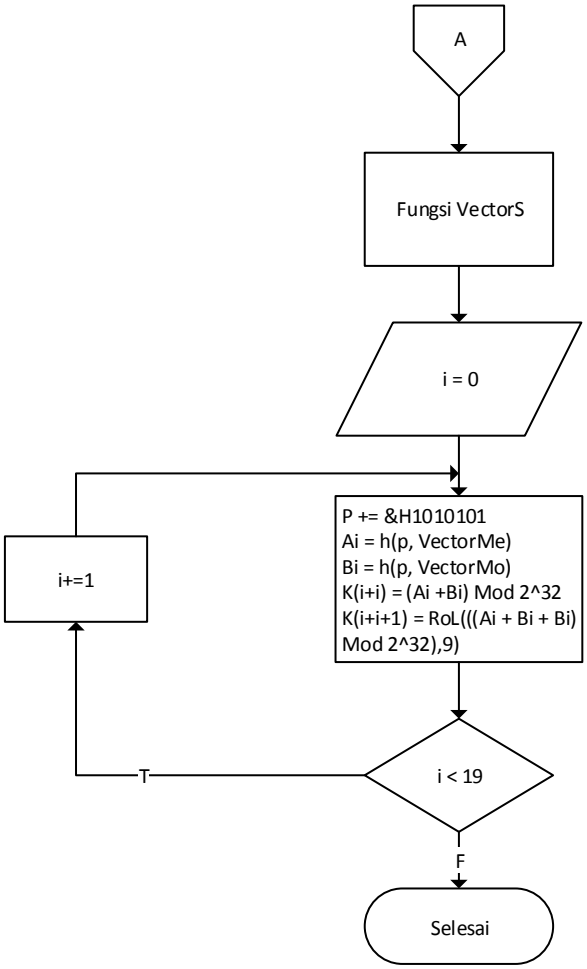
Flowchart untuk enkripsi text dan file adalah sebagai berikut:



Gambar 11. Flowchart Enkripsi File dan Text

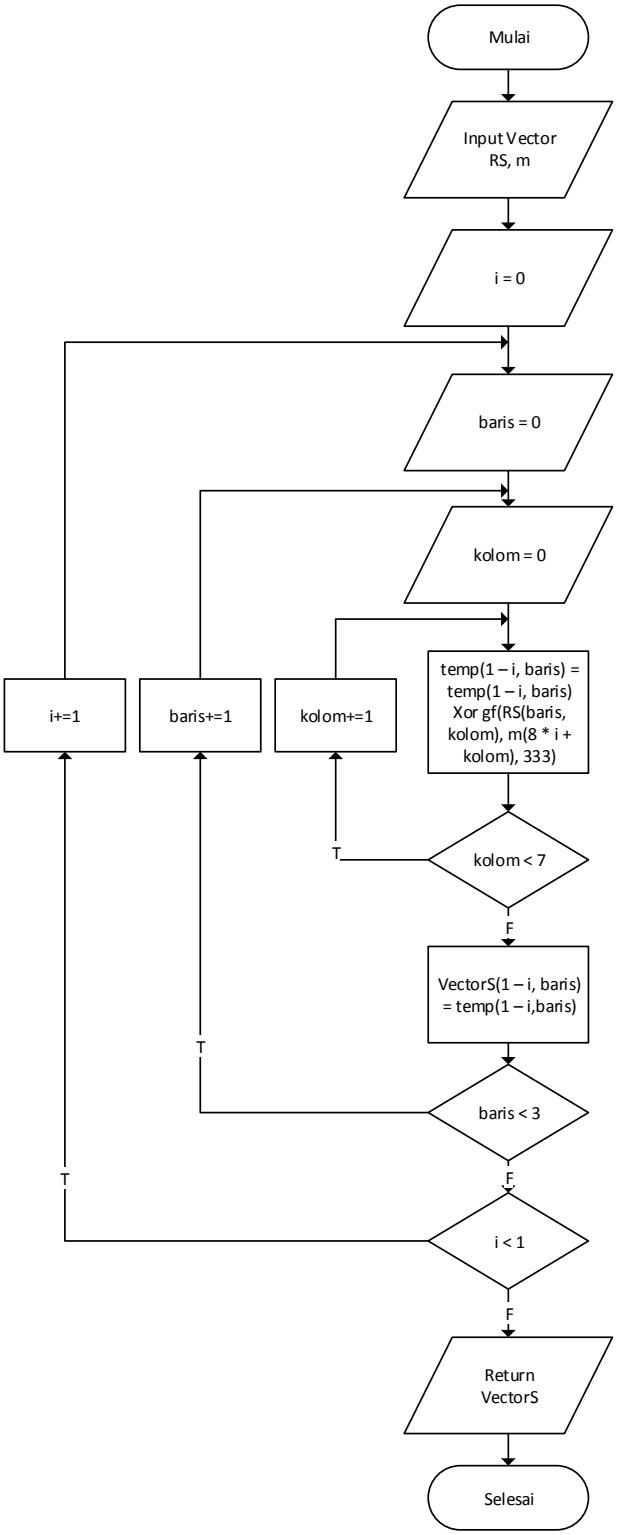
Flowchart sub procedur penjadwalan kunci adalah sebagai berikut :





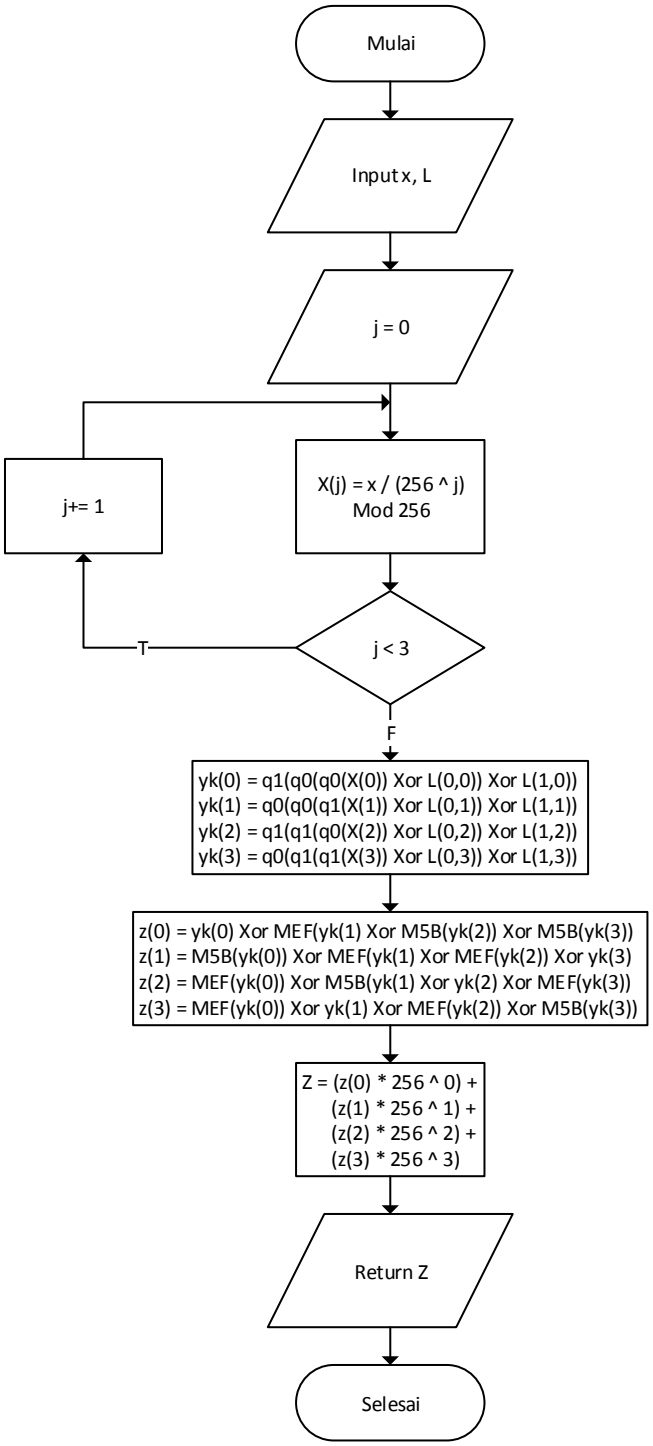
Gambar 12. Flowchart Sub Procedur Penjadwalan Kunci

Flowchart Fungsi Vector S adalah sebagai berikut :



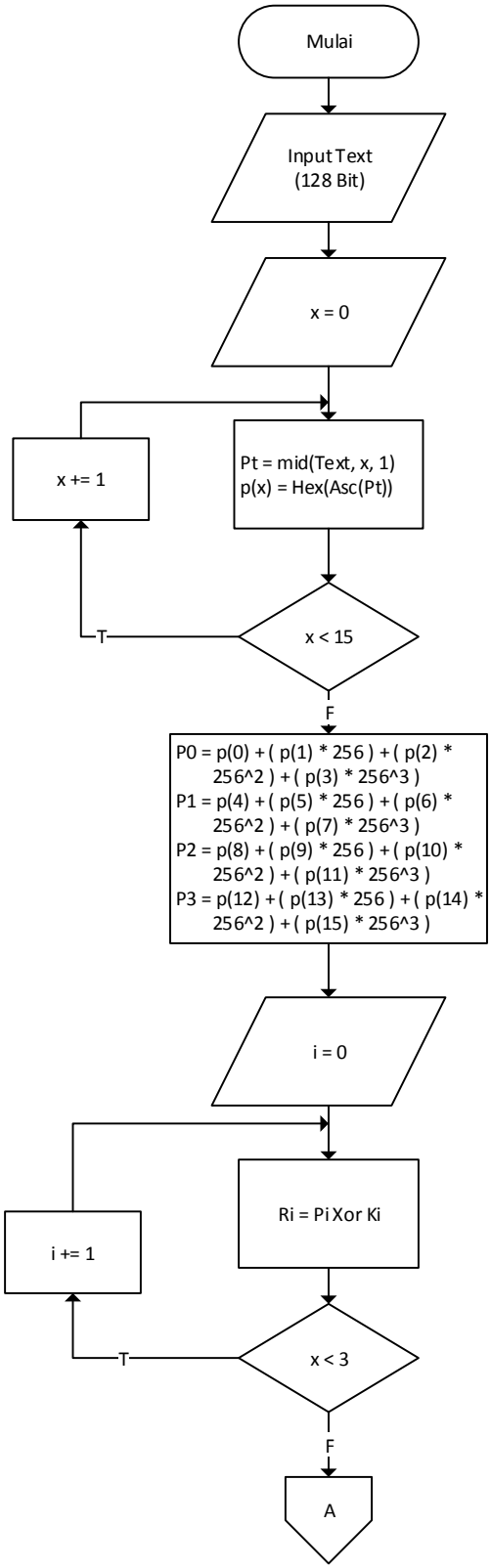
Gambar 13. Flowchart Fungsi Vector S

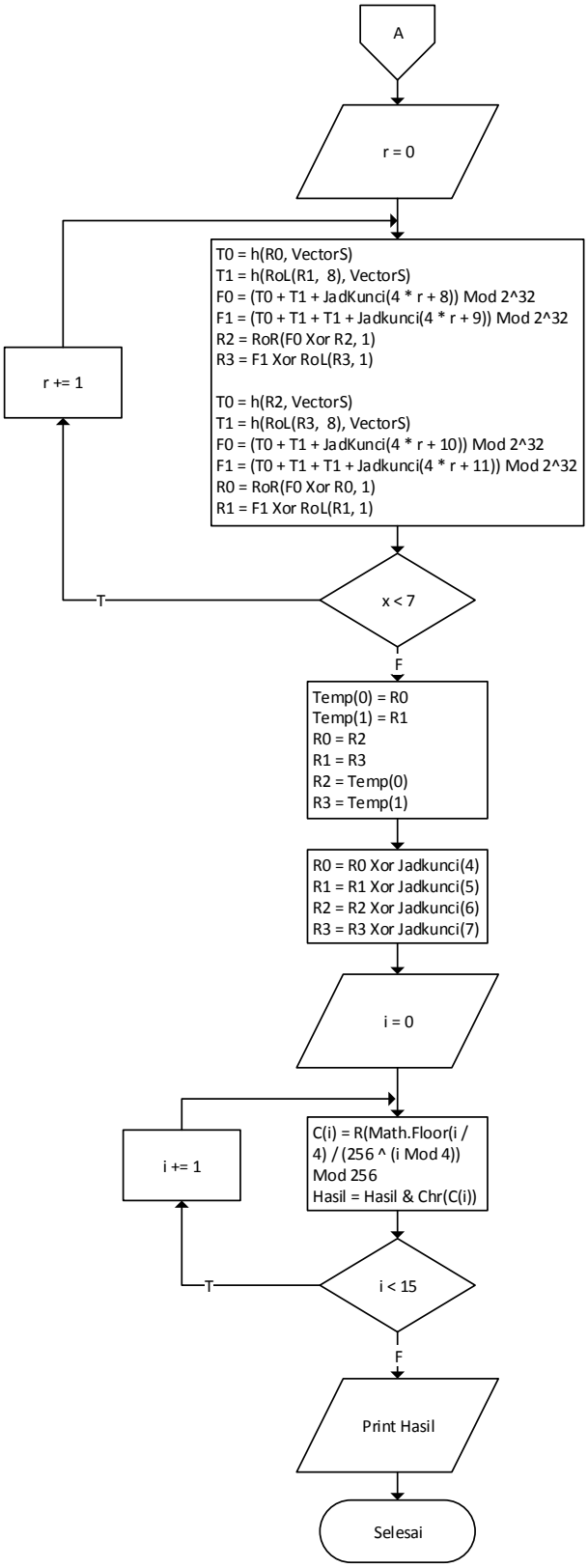
Flowchart fungsi h dan g adalah sebagai berikut :



Gambar 14. Flowchart Fungsi h dan g

Flowchart Fungsi Enkripsi adalah sebagai berikut :





Gambar 15. Flowchart Fungsi Enkripsi

B. Melakukan dekripsi data text.

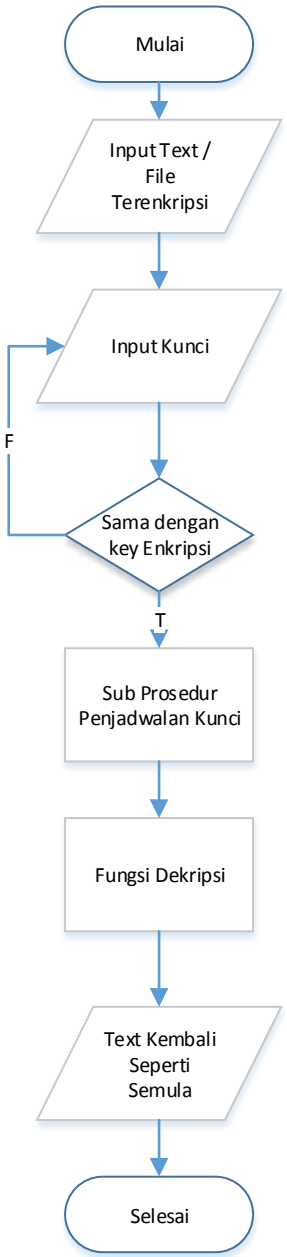
1. Masukkan file atau text yang sudah terenkripsi.

2. Masukkan kunci yang sama ketika file atau text dienkripsi.

3. Lakukan dekripsi untuk file atau text yang telah diinputkan.

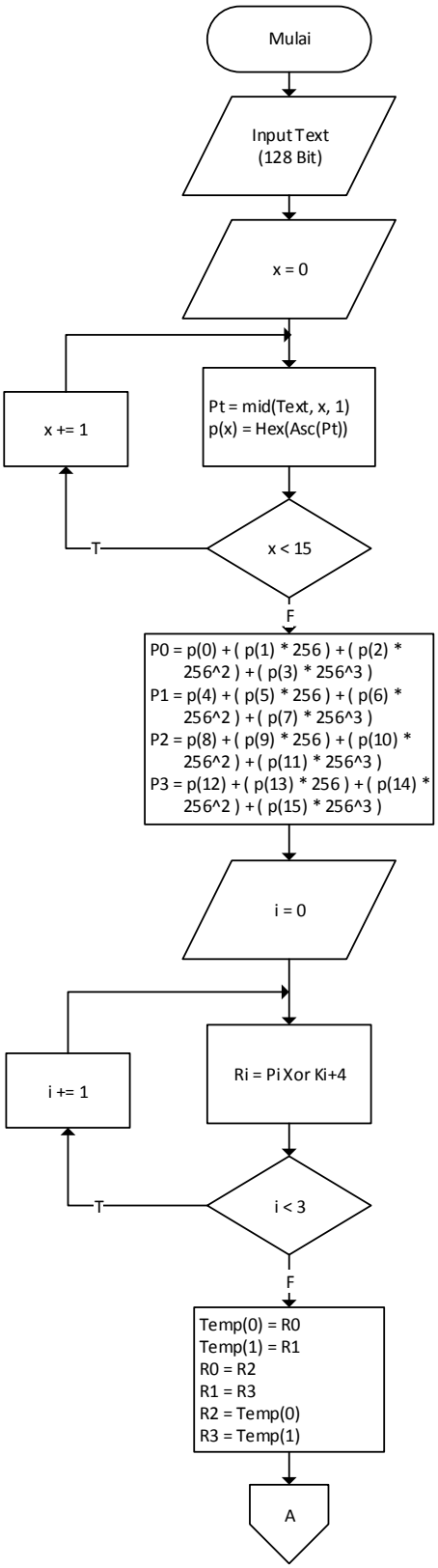
4. File atau text akan menjadi seperti keadaan semula.

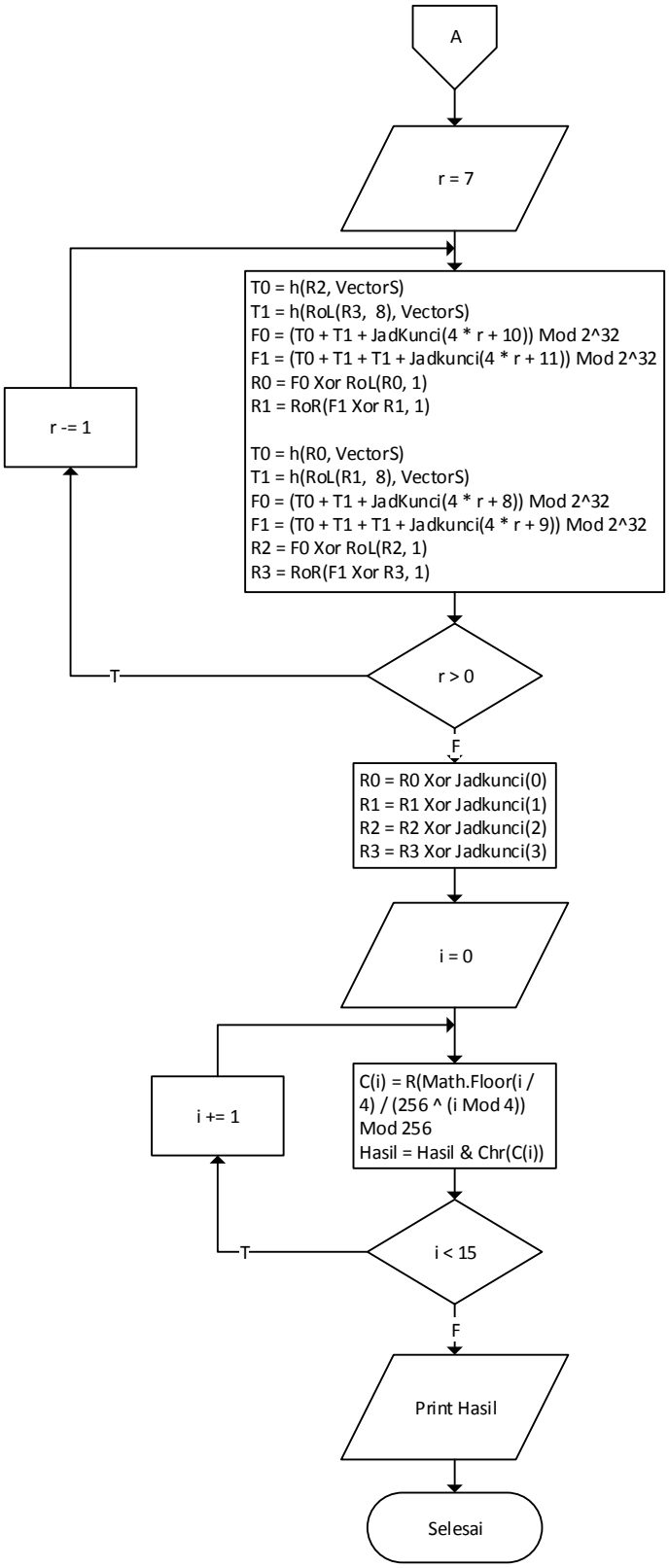
Flowchart dekripsi text dan file adalah sebagai berikut :



Gambar 16. Flowchart Dekripsi File dan Text

Flowchart fungsi dekripsi adalah sebagai berikut :





Gambar 17. Flowchart Fungsi Dekripsi

5.1.3. Proses Enkripsi dan Dekripsi

Proses yang dirancang pada perangkat lunak ini berdasarkan pada algoritma yang digunakan yaitu algoritma twofish. Algoritma twofish sendiri merupakan algoritma yang menggunakan kunci simetrik yaitu kunci yang digunakan untuk mengenkripsi sama dengan kunci yang digunakan pada saat mendekripsi. Twofish menggunakan 128 bit untuk setiap blok yang akan dienkripsi. Kunci yang digunakan dengan panjang 128 bit. Pada bab teori dasar telah dijelaskan tentang unsur pembangun twofish beserta algoritmanya. bagan algoritma twofish bisa dilihat pada **Gambar 5.** Secara lebih jelas tahapan-tahapan algoritma twofish akan dijabarkan sebagai berikut.

1. Langkah-Langkah Penjadwalan Kunci :

1. Sebelum melalui tahapan enkripsi, maka harus melalui penjadwalan kunci. Panjang kunci yang didefinisikan sepanjang 128 bit.
2. Setelah itu kunci dibagi menjadi vector Me, Mo, dan S. Vector Me dan Mo digunakan pada fungsi h sebagai *list*, sedangkan vector S digunakan untuk tahap enkripsi pada fungsi g.

3. Masukkan masing-masing kata kunci yang diekspansi yaitu 2i dan 2i+1 (i = 0,..,19) ke dalam fungsi h melalui permutasi q0 dan q1 dilanjutkan dengan matrik MDS.
4. Hasil dari *word* 2i melalui proses PHT, sedangkan *word* 2i+1 sebelum melalui proses PHT dilakukan rotasi kekiri sejauh 8 bit. Maka hasil dari proses tersebut menjadi kunci yang sudah terjadwal.

2. Langkah-Langkah Enkripsi Algoritma Twofish :

1. Masukan satu blok plainteks 128 bit. Satu blok tersebut dibagi menjadi 4 buah subblok yang masing-masing sepanjang 32 bit (A, B, C dan D).
2. Masing-masing subblok tersebut akan melalui input whitening yaitu meng-xor-kan subblok A, B, C dan D dengan K0, K1, K2 dan K3.

Langkah 1 putaran dalam Twofish adalah sebagai berikut :

3. Dua buah 32 bit yang kiri (A dan B) merupakan input dari fungsi g (yang merupakan bagian dari fungsi f), yang satu (B) di geser ke kiri sejauh 8 bit dahulu sebelum memasuki fungsi g.
4. Fungsi g memiliki 4 buah kotak subtitusi yang dibangkitkan oleh kunci.
5. Keluaran fungsi kotak subtitusi dilakukan pencampuran linear menggunakan kotak Most Distance Separable.

6. Keluaran fungsi g. dimasukan ke fungsi transformasi pseudo-Hadamard, kemudian ditambahkan dengan 2 buah 32 bit kunci K_{2r+8} dan K_{2r+9}.
7. Dua buah 32 bit hasil kemudian di xor-kan dengan C dan D. Hasil xor dengan C digeser ke kanan sejauh 1 bit. Dan untuk D sebelum di-xor-kan digeser ke kiri sejauh 1 bit
8. Dua buah 32 bit kiri dan kanan dipertukarkan (A dan B ditukar dengan C dan D).

Langkah di atas dilakukan hingga 16 kali putaran. Kemudian langkah selanjutnya :

9. Hasil keluaran setelah diputar 16 kali, di tukar lagi (A dan B di tukar dengan C dan D).
10. Hasil dari pertukaran tersebut di xor kan dengan empat buah 32 bit kunci K_4, K_5, K_6 dan K_7 untuk menghasilkan cipher teks.

Pada proses dekripsi, cara yang dilalui sama saja dengan proses enkripsi tetapi hanya arahnya saja yang berlawanan. Proses yang dilalui secara berurutan yaitu : *output whitening*, swap blok trakhir, 16 iterasi dekripsi, dan *input whitening*. Inputnya adalah chiperteks dan kunci untuk memperoleh plainteks. Kunci untuk

dekripsi sama saja dengan kunci enkripsi, begitupun juga panjang maksimal kunci yaitu 128 bit atau 16 karakter.

5.2. Tahapan Penelitian

Dalam pembuatan proses sistem yang dibangun melalui beberapa tahapan. Secara garis besar tahapan pembutan sistem yang akan dibangun adalah sebagai berikut:

a. Melakukan studi literatur

   Melakukan studi literatur yang berhungan dengan penelitian yang dikutip dari buku teks, paper, dan browsing internet. Bahasan yang ambil mengenai teori kriptografi dan algoritma twofish. Melalui studi literatur, teori tersebut dianalisis untuk mengimplementasikannya ke dalam bentuk perangkat lunak.

b. Analisis dan perancangan perangkat lunak.

   Menganalisis dan merancang perangkat lunak enkripsi dan dekripsi text dan file (.txt) untuk memperoleh perangkat lunak yang sesuai dengan teori dan rancangan yang telah dilakukan.

c. Pembuatan perangkat lunak

Pembuatan perangkat lunak berdasarkan analisis dan perancangan yang dilakukan sehingga diperoleh hasil yang optimal.

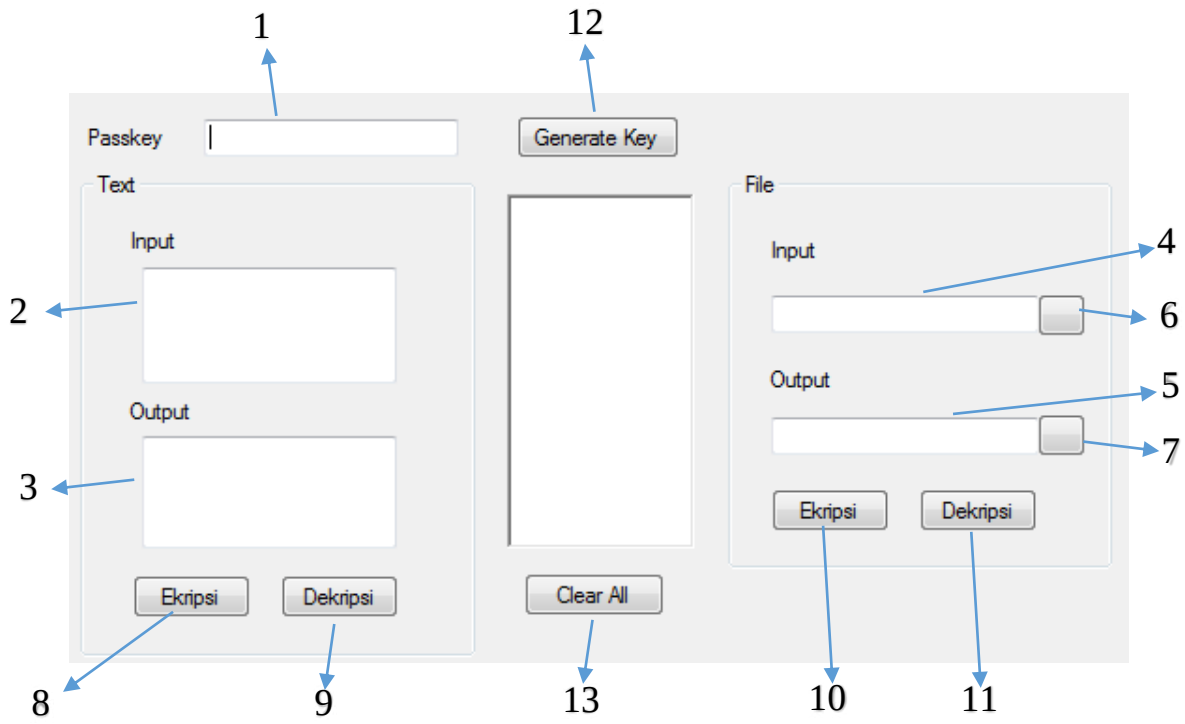
d. Uji coba dan evaluasi sistem.

Melakukan uji coba terhadap perangkat lunak. Uji coba yang dilakukan yaitu menguji keberhasilan enkripsi text serta mendekripsikannya kembali sehingga menjadi file atau text semula. Bahan yang akan diuji berupa file yang mempunyai kapasitas yang beragam. Selain itu mengevaluasi sistem yang telah diselesaikan dengan menganalisa kinerja perangkat lunak yang telah dibangun.

5.3. Perancangan Antarmuka

Rancangan antarmuka yang akan dibangun dengan satu *form* yang didalamnya terdapat enkripsi dan dekripsi data text.

Berikut adalah gambar *form* yang telah dibangun.



Gambar 18. Form Antarmuka

Keterangan Gambar :

1. *Text Field* Kunci : Isinya berupa Printable Character yang tidak lebih dari 128-byte atau 16 character
2. *Text Field* Input Text : Isinya berupa Printable Character
3. *Text Field* Output Text : Isinya berupa text yang telah di eksekusi enkripsi / dekripsi
4. *Text Field* Input File : Isinya berupa lokasi file yang akan dieksekusi enkripsi / dekripsi
5. *Text Field* Output File : Isinya berupa lokasi file yang telah dieksekusi enkripsi / dekripsi

6. *Button* Open File : Tombol untuk memilih file yang akan dieksekusi
7. *Button* Destinasi File : Tombol untuk memilih lokasi untuk menyimpan file yang telah dieksekusi
8. *Button* Enkripsi Text : Tombol untuk mengeksekusi program enkripsi text
9. *Button* Dekripsi Text : Tombol untuk mengeksekusi program dekripsi text
10. *Button* Enkripsi File : Tombol untuk mengeksekusi program enkripsi file
11. *Button* Dekripsi File : Tombol untuk mengeksekusi program dekripsi file
12. *Button* Generate Key : Tombol untuk membuat kunci K0, K1, …, K39
13. *Button* Clear All : Tombol untuk membersihkan semua isi *Text Field*

BAB VI

IMPLEMENTASI DAN EVALUASI

Bab ini akan membahas tentang pengujian dan analisa hasil program yang telah dibuat. Tujuan dari pengujian ini adalah untuk mengetahui apakah aplikasi yang telah dibuat sesuai dengan perancangannya.

6.1. Ruang Lingkup Pendukung Implementasi

Untuk menjalankan aplikasi yang telah dibangun maka dibutuhkan beberapa ruang lingkup pendukung implementasi yaitu berupa perangkat keras maupun perangkat lunak. Berikut adalah ruang lingkup untuk menjalankan aplikasi yang telah dibangun.

6.1.1. Ruang Lingkup Perangkat Keras

Dalam pembuatan aplikasi enkripsi menggunakan perangkat keras Notebook Acer Aspire 5315. Adapun spesifikasi yang perangkat keras yang digunakan adalah :

a. Prosesor Intel Core I5-4200U 1,6Ghz (4 CPUs).

b. RAM DDR II 4096 MB.

c. Hardisk dengan Kapasitas 500 GB.

d. Graphic Card Geforce 720M 1500 MB.

6.1.2. Ruang Lingkup Perangkat Lunak

Perangkat lunak yang digunakan untuk pengembangan aplikasi ini antara lain :

1. Sistem Operasi Windows 7 Ultimate 32 Bit.
2. Visual Studio 2012.

6.2. Implementasi Sistem

Pada subbab ini akan memaparkan implementasi sistem berdasarkan rancangan program. Rancangan yang telah dirancang akan diimplentasikan ke dalam bentuk *sourcecode* dalam bahasa pemrograman Visual Basic 2010. Berikut adalah paparan implementasi dari perangkat lunak yang telah dibangun.

6.2.1. Penjadwalan Kunci

Sebelum melalukan proses enkripsi pada twofish maka diperlukan penjadwalan kunci. Sebelum penjadwalan kunci, vector Me, Mo, dan S, dipersiapkan terlebih dahulu. Setelah itu vector Me dan Mo menjadi bagian dari penjadwalan kunci sebagai masukan, sedangkan vector S akan digunakan pada saat enkripsi. Lalu proses berikutnya akan dilakukan perhitungan fungsi *h*. Berikut adalah *source code* dari penjadwalan kunci :

```
Sub Hitung_JadKunci()
    Dim p As Integer = 0
    Dim Ai, Bi, K2i, K2il As Long
    Dim i As Integer
    For i = 0 To 19
        p = p + &H1010101
        Ai = h(p, VectorMe)
```

```
        Bi = RoL64(h(p, VectorMo), 8)
        JadKunci(i + i) = Hex((Ai + Bi) Mod 2 ^ 32)
        JadKunci(i + i + 1) = Hex(RoL64(((Ai + Bi + Bi) Mod 2 ^ 32), 9))
    Next
End Sub
```

6.2.2. Proses Enkripsi

Proses enkripsi pada algoritma twofish melalui beberapa tahapan. Pertama masukan dari plainteks sebesar 128 bit dipecah menjadi empat bagian yang masing-masing sebesar 32 bit lalu di XOR-kan dengan kunci yang sudah terjadwal yang masing-masing juga dibagi menjadi empat bagian. Hal ini disebut dengan proses *whitening*. Hasil *whitening* akan akan melalui fungsi *F* yang mempunyai output F0 dan F1 dan masing-masing di-XORkan dengan R2 dan R3 (melalui rotasi kekiri 1 bit). Setelah itu R0 dan R1 akan di swap dengan R2 dan R3 kemudian dilakukan iterasi sebanyak 16 kali iterasi, Setelah iterasi ke 16 maka dilakukan swap blok akhir yaitu dengan meng-*undo* swap blok paling akhir tersebut. Terakhir dilakukan peng-XORan terhadap beberapa kunci yang terjadwal yaitu K4, K5, K6, dan K7. Demikian proses enkripsi yang dilakukan. Berikut adalah *source code* untuk proses enkripsi:

```
Function Enkripsi(block As Array) As Long
    Dim i As Byte
    Dim T0, T1 As Long
    Dim F0, F1 As Long
    Dim hasil As Long
    Input_Whitening()
    For i = 0 To 7
        T0 = h(block(0), VectorS)
        T1 = h(RoL64(block(1), 8), VectorS)
        F0 = T0 + T1 + JadKunci(4 * i + 8)
        F1 = T0 + T1 + T1 + JadKunci(4 * i + 9)
        block(2) = RoR64(F0 Xor block(2), 1)
        block(3) = F1 Xor RoL64(block(3), 1)
```

```
        T0 = h(block(2), VectorS)
        T1 = h(RoL64(block(3), 8), VectorS)
        F0 = T0 + T1 + JadKunci(4 * i + 10)
        F1 = T0 + T1 + T1 + JadKunci(4 * i + 11)
        block(2) = RoR64(F0 Xor block(0), 1)
        block(3) = F1 Xor RoL64(block(1), 1)
    Next
    Swap_Block()
    Output_Whitening()
    hasil = Asc(block(0)) & Asc(block(1)) & Asc(block(2)) & Asc(block(3))
    Return hasil
End Function
```

6.2.3. Proses Dekripsi

Proses yang dilalui pada saat dekripsi sama saja dengan proses enkripsi tetapi hanya arahnya saja yang berlawanan. Proses yang dilalui secara berurutan yaitu : *output whitening*, swap blok trakhir, 16 iterasi dekripsi, dan *input whitening*. Inputnya adalah chiperteks dan kunci untuk memperoleh plainteks. Kunci untuk dekripsi sama saja dengan kunci enkripsi, begitupun juga panjang maksimal kunci yaitu 128 bit atau 16 karakter. Berikut adalah *source code* untuk mendekripsi file :

```
Function Dekripsi(block As Array) As Long
    Dim i As Byte
    Dim T0, T1 As Long
    Dim F0, F1 As Long
    Dim hasil As Long
    Output_Whitening()
    For i = 7 To 0 Step -1
        T0 = h(block(0), VectorS)
        T1 = h(RoL64(block(1), 8), VectorS)
        F0 = T0 + T1 + JadKunci(4 * i + 10)
        F1 = T0 + T1 + T1 + JadKunci(4 * i + 11)
        block(2) = RoL64(T0 Xor block(2), 1)
        block(3) = T1 Xor RoR64(block(3), 1)

        T0 = h(block(2), VectorS)
```

```
        T1 = h(RoL64(block(3), 8), VectorS)
        F0 = T0 + T1 + JadKunci(4 * i + 8)
        F1 = T0 + T1 + T1 + JadKunci(4 * i + 9)
        block(2) = RoL64(T0 Xor block(0), 1)
        block(3) = T1 Xor RoR64(block(1), 1)
    Next
    Swap_Block()
    Input_Whitening()
    hasil = Asc(block(0)) & Asc(block(1)) & Asc(block(2)) & Asc(block(3))
    Return hasil
End Function
```

6.2.4. Vector Me, Mo Dan S

Vector Me dan Mo akan digunakan didalam fungsi h sedangkan Vector S digunakan didalam fungsi g. Berikut *source code* Vector M dan S

```
Sub Persiapan_VectorM()
    Dim i, j As Integer
    For i = 0 To 1
        For j = 0 To 3
            VectorMe(i, j) = m(4 * (2 * i) + j)
            VectorMo(i, j) = m(4 * (2 * i + 1) + j)
        Next
    Next
        End Sub

        Sub Hitung_VectorS()
    Dim i, baris, kolom As Byte
    For i = 0 To 1
        For baris = 0 To 3
            For kolom = 0 To 7
                        VectorS(1 - i, baris) = VectorS(1 - i, baris) Xor     gf(RS(baris
                        kolom), m(8 * i + kolom), 333)
            Next
        Next
    Next
        End Sub
```

6.2.5. Fungsi h dan g

Fungsi h akan digunakan didalam penjadwalan kunci untuk mencari K0,…,K39 sedangkan fungsi g akan digunakan untuk proses enkripsi dan dekripsi. Berikut *source code* fungsi h dan g.

```
Function h(x As Long, L As Array) As Long
    Dim Output(3) As Long
    Dim OutputL As Long
    Dim j, i As Integer
    Dim xi(3) As Double
    Dim yk(3) As Integer
    Dim q0(3), q1(3) As Integer

    For j = 0 To 3
       xi(j) = ((x / (256 ^ j)) Mod 256)

       q0(j) = q0f(xi(j))
       q1(j) = q1f(xi(j))
    Next
    yk(0) = q0f(q1f(q1(0) Xor L(0, 0)) Xor L(1, 0))
    yk(1) = q1f(q1f(q0(1) Xor L(0, 1)) Xor L(1, 1))
    yk(2) = q0f(q0f(q1(2) Xor L(0, 2)) Xor L(1, 2))
    yk(3) = q1f(q0f(q0(3) Xor L(0, 3)) Xor L(1, 3))

    Output(0) = yk(0) Xor MEF(yk(1)) Xor M5B(yk(2)) Xor M5B(yk(3))
    Output(1) = M5B(yk(0)) Xor MEF(yk(1)) Xor MEF(yk(2)) Xor yk(3)
    Output(2) = MEF(yk(0)) Xor M5B(yk(1)) Xor yk(2) Xor MEF(yk(3))
    Output(3) = MEF(yk(0)) Xor yk(1) Xor MEF(yk(2)) Xor M5B(yk(3))
    OutputL = Output(0) + (Output(1) * 256) + (Output(2) * 256 * 256) +
       (Output(3) * 256 * 256 * 256)

    Return OutputL
End Function
```

6.2.6. Permutasi q0 dan q1

Permutasi q0 dan q1 akan digunakan didalam fungsi h dan g untuk melakukan permutasi dengan input 1 byte dan mengeluarkan output 1 byte. Berikut *source code* q0 dan q1.

```
 Function q0f(xi As Double) As Byte
    Dim a(5), b(5) As Double
    a(0) = Pembulatan(xi / 16)
    b(0) = xi Mod 16
    a(1) = a(0) Xor b(0)
    b(1) = a(0) Xor RoR4(b(0), 1) Xor 8 * a(0) Mod 16
    a(2) = t0q0(a(1))
    b(2) = t1q0(b(1))
    a(3) = a(2) Xor b(2)
    b(3) = a(2) Xor RoR4(b(2), 1) Xor 8 * a(2) Mod 16
    a(4) = t2q0(a(3))
    b(4) = t3q0(b(3))

    Return 16 * b(4) + a(4)
End Function

Function q1f(xi As Double) As Byte
    Dim a(5), b(5) As Double
    a(0) = Pembulatan(xi / 16)
    b(0) = xi Mod 16
    a(1) = a(0) Xor b(0)
    b(1) = a(0) Xor RoR4(b(0), 1) Xor 8 * a(0) Mod 16
    a(2) = t0q1(a(1))
    b(2) = t1q1(b(1))
    a(3) = a(2) Xor b(2)
    b(3) = a(2) Xor RoR4(b(2), 1) Xor 8 * a(2) Mod 16
    a(4) = t2q1(a(3))
    b(4) = t3q1(b(3))

    Return 16 * b(4) + a(4)
End Function
```

BAB VII

PENUTUP

7.1. Kesimpulan

Dari hasil penelitian yang didapat maka dapat disimpulkan bahwa:

1. Twofish menggunakan 128 bit setiap blok yang akan dienkripsi. Kunci yang digunakan dengan panjang 128 bit dengan menggunakan kunci simetrik dimana kunci pada saat enkripsi sama dengan dekripsi.
2. Aplikasi yang dibangun berhasil mengenkripsi dan mendekripsi text maupun file.

7.2. Saran

Dalam pembuatan sistem yang dilakukan , tentunya tidak terlepas dari kekurangan yang ada, sehingga dapat disarankan untuk dapat menjadi acuan dalam pengembangan sistem atau penelitian selanjutnya sebagai berikut :

Sistem yang penulis bangun disini hanya dapat mengenkripsi dan dekripsi data text dan file (.txt) sehingga data resourse yang bisa diolah sangat terbatas. Oleh karena itu disarankan untuk pengembangan selanjutnya bisa menambahkan berbagai macam fitur enkripsi dan dekripsi.

DAFTAR PUSTAKA

Fairuzabadi, Muhammad, 2010. *Implementasi Kriptografi Klasik menggunakan Borland Delphi.* Vol. 2 , No. 2, Universitas PGRI Yogyakarta. (journal)

Hendra, Andi, 2010. *Analisis Perbandingan Kinerja Algoritma Twofish dan TEA (Tiny Encryption Algorithm) Pada Data Suara.* Vol. 1, No. 3, JIMT. (journal)

Landge, Irfan., Bharmal, Tasneem. and Narwankar Pooja. 2012. *Encryption and decryption of data using twofish algorithm.* R.C.Patel Institute of Technology Dhule, Maharashtra, India (journal)

Montaseri, Fanidzar, 2010. *Perancangan dan Pembuatan Aplikasi Enkripsi dan Dekripsi File dan Folder Menggunakan Metode Twofish*, Universitas Islam Negeri Maulana Malik Ibrahim Malang. (journal)

Prayudi, Yudi., Halik, Idham, 2005. *Studi dan Analisis Algoritma Rivers Code 6 (RC6) dalam Enkripsi/Dekripsi Data.* Universitas Islam Indonesia Yogyakarta. (journal)

Schneier, Bruce dkk, 1998. *Twofish: A 128-Bit Block Cipher.* (journal) (www.schneier.com/paper-twofish-paper.pdf)

Yanuarto, Alfath. 2006. *Analisis Algoritma dan Keamanan Sandi Blok Twofish Dalam Penyandian Pesan*. Institut Pertanian Bogor. (journal)

LAMPIRAN

Contoh Proses Penjadwalan Kunci Algoritma Twofish

1. Misalkan Kunci yang digunakan "12345678abcdefgh". Masing-masing karakter dari kunci tersebut di rubah ke kode ASCII 1=49, 2=50, 3=51, 4=52, 5=53, 6=54, 7=55, 8=56, a=97, b=98, c=99, d=100, e=101, f=102, g=103, h=104.
2. Selanjutnya kode ASCII tersebut kemudian dirubah kedalam hexadecimal 128-bit, yaitu: &H31323334353637386162636465666768.
3. Kemudian Key Tersebut di bagi menjadi 16 byte m0….,m15 yaitu m0=31, m1=32, m2=33, m3=34, m4=35, m5=36, m6=37, m7=38, m8=61, m9=62, m10=63, m11=64, m12=65, m13=66, m14=67, m15=68.
4. Selanjutnya di kelompokan kedalam 4 kelompok menggunakan rumus.

$$M_i = \sum_{j=0}^{3} m_{(4i+j)}.2^{8j} \quad (2.24)$$

$i = 0,\ldots 2k\text{-}1$

5. Maka didapatkan lah M0 sampai M3 dalam bentuk hexadecimal:

   M0 = m12 + m8 + m4 + m0 = 65613531

   M1 = m13 + m9 + m5 + m1 = 66623632

   M2 = m14 + m10 + m6 + m2 = 67633733

   M3 = m15 + m11 + m7 + m3 = 68643834

6. Kemudian di kelompokan lagi ke dalam Vector Me dan Vector Mo

   Me = M0, M2 (2.25) Mo = M1, M3 (2.26)

7. Berikutnya Mencari Vector S dengan cara membagi kunci 128-bit menjadi 8 bagian kemudian 8 bagian tersebut dipresentasikan di atas vector GF(2^8) dan mengkalikannya dengan matrik 4x8.

   Maka didapatkan lah :

   $S_{0,0}$ = E5

   $S_{0,1}$ = 60

   $S_{0,2}$ = D1

   $S_{0,3}$ = 67

   $S_{1,0}$ = B9

   $S_{1,1}$ = C4

   $S_{1,2}$ = C5

   $S_{1,3}$ = 9B

$$\begin{pmatrix} S_{i,0} \\ S_{i,1} \\ S_{i,2} \\ S_{i,3} \end{pmatrix} = \begin{pmatrix} . \\ .RS \\ . \end{pmatrix} . \begin{pmatrix} m_{Si} \\ m_{Si+1} \\ m_{Si+2} \\ m_{Si+3} \\ m_{Si+4} \\ m_{Si+5} \\ m_{Si+6} \\ m_{Si+7} \end{pmatrix}$$

$$S_i = \sum_{j=0}^{3} S_{i,j} . 2^{8.j}$$

8. Setelah ketiga Vector diatas telah di dapatkan maka Vector Me, Mo akan digunakan di dalam fungsi h untuk mencari K0,….,K39 sedangkan S akan di gunakan di fungsi g untuk proses enkripsi.

9. Fungsi h membutuhkan 2 Input yaitu p = &H10101010 dan Vector Me/Mo

   p += $2^{24} + 2^{16} + 2^8 + 2^0$ = &H1010101 (2.41)

   Ai = h(p, VectorMe) (2.42)

   Bi = RoL(h(p, VectorMo), 8) (2.43)

   K(i + i) = (Ai + Bi) Mod 2 ^ 32 (2.44)

   K(i + i + 1) = RoL(((Ai + Bi + Bi) Mod 2 ^ 32), 9) (2.45)

   untuk i = 0,…,19

Maka didapatkan lah :

K0 = 9D471D28	K1 = 7872E6A8	K2 = B406CAF6
K3 = 59AB81E3	K4 = AFE0BB53	K5 = 5E9DD194
K6 = AE59E2A0	K7 = 19D4908A	K8 = D99B73CE
K9 = D3669972	K10 = BA1B8CB2	K11 = 388AAE30
K12 = 6DC307E6	K13 = 7D5F3952	K14 = 3EE0A1C1
K15 = BEA3B001	K16 = CF0F5F4	K17 = 19858111
K18 = 36F21E16	K19 = F70D9D08	K20 = ECF24E2F
K21 = 92DB8BBA	K22 = 9011E75B	K23 = F7C2DAAA
K24 = 72AFAEDE	K25 = 27A0AD53	K26 = 6EBBD888
K27 = A7463AA2	K28 = 9F49B9BB	K29 = E71EAF2C
K30 = 5E5616BB	K31 = 6126BE8C	K32 = F03F7DF4
K33 = 7B10D585	K34 = 9417C47F	K35 = 898A907C
K36 = B840D7E2	K37 = 67AB898C	K38 = F06EBB39

K39 = 82098B2E

Contoh Proses Enkripsi Algoritma Twofish

1. Misalkan pesan yang digunakan "12345678abcdefgh". Masing-masing karakter dari kunci tersebut di rubah ke kode ASCII 1=49, 2=50, 3=51, 4=52, 5=53, 6=54, 7=55, 8=56, a=97, b=98, c=99, d=100, e=101, f=102, g=103, h=104.
2. Kemudian pesan tersebut dibagi menjadi empat bagian masing-masing 32 bit menggunakan rumus :

   P(i) = $\sum_{j=0}^{3} P(4i+j).2^{8j}$ (2.5)

   i = 0, …,3

   Maka didapatkanlah P0 sampai P3 dalam bentuk hexadecimal:

   P0 = 1345E15F P1 = EBB43A62

   P2 = 4427B884 P3 = D19012F8
3. Masing-masing subblok tersebut akan melalui input whitening yaitu meng-xor-kan subblok P0, P1, P2 dan P3 dengan K0, K1, K2 dan K3. (2.6)

   R0 = 8E02FC77 R1 = 93C6DCCA

   R2 = F0217272 R3 = 883B931B
4. Setelah di xor kan dengan kunci R0 dan R1 menjadi bagian kiri sedangkan R2 dan R3 menjadi bagian kanan.
5. Selanjutnya R0 dan R1 akan memasuki fungsi g yang merupakan bagian awal dari fungsi f, R1 di geser ke kiri sejauh 8 bit sebelum memasuki fungsi g.
6. Fungsi g membutuhkan 2 input yaitu R(i) dan Vector S, R(i) akan dibagi menjadi 4 bagian masing-masing 8 bit dan masing-masing 8 bit melewati S-Box. Setelah melewati 4 buah S-Box masing-masing 8 bit akan digabung lagi menjadi 32-bit dengan menggunakan MDS.

Maka didapatkan :

T0 = 8C9507E6 (2.16) T1 = EA9C54C6 (2.17)

7. Selanjutnya T0 dan T1 akan melewati PHT dan ditambahkan dengan K8 dan K9 dan di modulo 2^{32} untuk putaran pertama.

   F0 = (T0 + T1 + JadKunci(2 * i + 8)) Mod 2 ^ 32 (2.18)

   F1 = (T0 + T1 + T1 + JadKunci(2 * i + 9)) Mod 2 ^ 32 (2.19)

   Maka didapatkan :

   F0 = 51FFEEC4 F1 = 3667692E

8. F0 dan F1 kemudian di xor-kan dengan R2 dan R3. Hasil xor dengan R2 digeser ke kanan sejauh 1 bit. Dan untuk R3 sebelum di-xor-kan digeser ke kiri sejauh 1 bit.

   R2 = RoR(F0 Xor R2, 1) R3 = F1 Xor RoL(R3, 1)

   Maka didapatkan :

   R2 = 3E5A3ACA R3 = FAEB72F1

9. Selanjutnya R0, R1 akan di swap dengan R2, R3
10. Setelah itu akan dilakukan iterasi sebanyak 16 kali dari langkah 5 sampai 9.
11. Setelah melewati 16 iterasi R0, R1 akan di swap lagi dengan R2,R3.

    Maka didapatkan :

    R0 = BCA55A0C R1 = B529EBF6

    R2 = EA7E5A24 R3 = C8448272

12. Hasil dari pertukaran tersebut kemudian di xor kan dengan K4, K5, K6 dan K7.

    C0 = R0 xor K4 C1 = R1 xor K5

    C2 = R2 xor K6 C3 = R3 xor K7

    Maka didapatkan :

C0 = 1345E15F C1 = EBB43A62

C2 = 4427B884 C3 = D19012F8

13. 4 kata C0, C1, C2 dan C3 dipisah menjadi 16 byte c0,…,c15 untuk menghasilkan Ciphertext 128 bit menggunakan rumus :

$$c_i = \left\lfloor \frac{C\lfloor i/4 \rfloor}{2^{8(i \bmod 4)}} \right\rfloor \bmod 2^8$$

i = 0,…,15

c0 = 5F c1 = E1 c2 = 45 c3 = 13

c4 = 62 c5 = 3A c6 = B5 c7 = EB

c8 = 84 c9 = B8 c10 = 27 c11 = 44

c12 = F8 c13 = 12 c14 = 90 c15 = D1

Setelah didapatkan maka masing-masing byte hexadecimal c0,…,c15 di rubah ke dalam ASCII Character. Maka didapatkan lah :

Cipherteks = _áEb:´ë,,¸'Dø• Ñ