SISTEM PENGKODEAN DATA FILE TEKS PADA KEAMANAN INFORMASI MENGGUNAKAN METODE ALGORITMA CAST-128

Tacbir Hendro Pudjiantoro

Fakultas Matematika dan Ilmu Pengetahuan Alam
UNIVERSITAS JENDERAL ACHMAD YANI

Tacbir23501027@yahoo.com

Abstrak

Kemajuan di bidang teknologi informasi telah memungkinkan instansi – instansi pendidikan atau lainnya melakukan interaksi dengan konsumen melalui jaringan komputer. Kegiatan-kegiatan tersebut tentu saja akan menimbulkan resiko bilamana informasi yang sensitif dan berharga tersebut diakses oleh orang-orang yang tidak berhak.

Untuk proteksi data yang cukup penting tidak ada jalan lain selain menggunakan program khusus proteksi atau enkripsi dan dekripsi data. Salah satu metode enkripsi dan dekripsi adalah kriptografi kunci simetrik. Salah satu algoritma kunci simetrik yaitu CAST-128, yang menggunakan kunci yang sama pada saat proses enkripsi dan dekripsi, dan menggunakan kunci sampai 128 bit. CAST-128 menggunakan enam belas putaran jaringan feistel sebagai salah satu kekuatannya, dan juga menggunakan panjang blok 64 bit.

Tujuan dari penelitian ini adalah menganalisis dan membuat sistem keamanan data enkripsi dan dekripsi yang berformat txt (teks), dan panjang kunci maksimal 128 bit. Di dalam Skripsi ini akan dibahas lebih lanjut tentang sistem keamanan data dengan metode CAST-128. Dan juga pengujian – pengujian yang dilakukan pada perangkat lunak dengan menggunakan metoda *black box* yaitu pengujian dengan fokus pada persyaratan fungsional perangkat lunak yang telah dibuat.

Kata-kata Kunci : informasi berharga , CAST-128, enkripsi dan dekripsi.

1. PENDAHULUAN

Pada era global seperti sekarang ini, keamanan sistem informasi komputer dan interkoneksinya melalui jaringan menjadi suatu keharusan untuk diperhatikan, karena jaringan komputer yang sifatnya publik dan global pada dasarnya tidak aman. Pada saat data terkirim dari suatu komputer ke komputer yang lain, data itu akan melewati sejumlah komputer yang lain yang berarti akan memberi kesempatan pada user lain untuk menyadap atau mengubah data tersebut. Dalam hal ini keamanan data merupakan permasalahan yang sangat penting. Untuk memenuhi hal tersebut, dilakukan proses enkripsi dan dekripsi terhadap informasi yang akan dikirimkan. Permasalahan utama dalam proses enkripsi dan dekripsi adalah penentuan algoritma yang tepat dan efisien. Algoritma yang digunakan untuk

proses enkripsi dan dekripsi data pada skripsi ini yaitu algoritma CAST-128. CAST-128 suatu algoritma simetrik, beroperasi dalam mode blok (*block cipher*), menggunakan tiga tipe fungsi *round,* dan yang bekerja dengan enam belas putaran jaringan feistel, dengan panjang kunci bervariasi antara 40 sampai 128 dalam kelipatan delapan.

Dalam penelitian ini akan dibahas tentang enkripsi dan dekripsi yang digunakan dengan metoda CAST dengan panjang kunci maksimal 128 bit atau lebih dikenal dengan nama CAST-128 dan hanya pada file yang berformat txt (berisi teks).

2. KRIPTOGRAFI

Kriptografi (*Cryptography*) berasal dari bahasa Yunani: "*Cryptos"*artinya *"secret"* (rahasia), sedangkan "g*raphein"* artinya *"writing"* (tulisan). Jadi, kriptografi berarti *"secret writing"* (tulisan rahasia). Yaitu suatu ilmu yang mempelajari penulisan secara rahasia. Kriptografi merupakan bagian dari suatu cabang ilmu matematika yang disebut *Cryptology*. Kriptografi bertujuan menjaga kerahasiaan informasi yang terkandung dalam data sehingga informasi tersebut tidak dapat diketahui oleh pihak yang tidak sah.

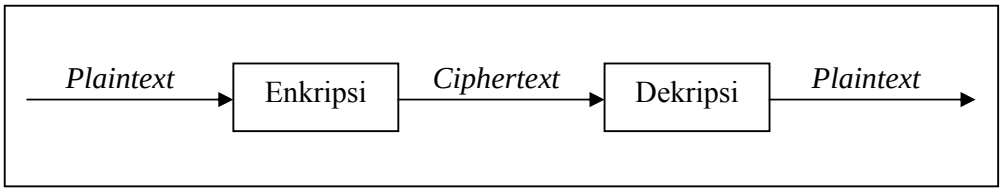
Kriptografi adalah ilmu yang mempelajari bagaimana supaya pesan atau dokumen kita aman, tidak bisa dibaca oleh pihak yang tidak berhak. Dalam perkembangannya, kriptografi juga digunakan untuk identifikasi pengirim pesan dengan tanda tangan digital dan keaslian pesan dengan sidik jari digital (*fingerprint*). Tugas utama kriptografi adalah untuk menjaga agar baik plainteks maupun kunci ataupun keduanya tetap terjaga kerahasiaannya dari penyadap (disebut juga sebagai lawan, penyerang, pencegat, penyelundup pesan, musuh attacker dan sebagainya). Pencegat pesan rahasia mempunyai akses yang lengkap ke dalam saluran komunikasi antara pengirim dan penerima. Ini sangat mudah terjadi pada jalur internet dan saluran telepon.

Dalam menjaga kerahasiaan data, kriptografi mentransformasikan data jelas *(plaintext)* ke dalam bentuk data sandi *(ciphertext)* yang tidak dapat dikenali. Cipherteks inilah yang kemudian dikirimkan oleh pengirim (*sender*) kepada penerima (*receiver*). Setelah sampai di penerima, cipherteks tersebut ditransformasikan kembali ke dalam bentuk plainteks agar dapat dikenali.

Proses transformasi dari plainteks menjadi cipherteks disebut proses *Encipherment* atau enkripsi (*encryption*). Sedangkan proses mentransformasikan kembali cipherteks menjadi plainteks disebut dekripsi (*decryption*).

Untuk mengenkripsi dan mendekripsi data, kriptografi menggunakan suatu algoritma (*cipher*) dan kunci (*key*). *Cipher* adalah fungsi matematika yang digunakan untuk

mengenkripsi dan mendekripsi data. Sedangkan kunci merupakan sederetan bit yang diperlukan untuk mengenkripsi dan mendekripsi data.

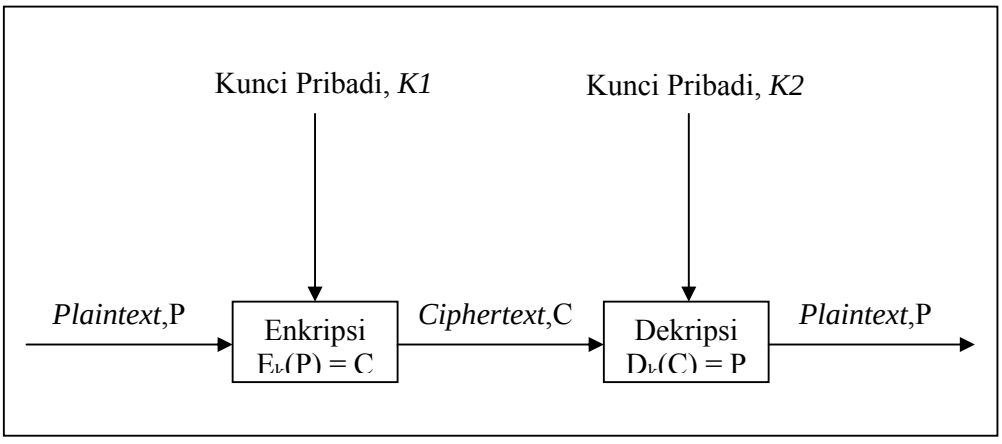


Gambar 1. Kriptografi dengan Proses Enkripsi dan Dekripsi

3. ALGORITMA SIMETRI

Algoritma simetri disebut juga sebagai algoritma konvensional adalah algoritma yang menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Algoritma simetri sering juga disebut algoritma kunci rahasia, algoritma kunci tunggal atau algoritma satu kunci, dan mengharuskan pengirim dan penerima menyetujui suatu kunci tertentu sebelum berkomunikasi dengan aman. Keamanan algoritma simetri tergantung pada kunci, membocorkan kunci berarti bahwa orang lain dapat mengenkrip dan mendekrip pesan agar komunikasi tetap aman, kunci harus tetap dirahasiakan.

Secara umum, *cipher* yang termasuk ke dalam kriptografi simetri beroperasi dalam mode blok (*block cipher*), yaitu setiap kali enkripsi/dekripsi dilakukan terhadap satu blok data (yang berukuran tertentu), atau beroperasi dalam mode aliran (*stream cipher*), yaitu setiap kali enkripsi/dekripsi dilakukan terhadap satu bit atau *byte* data.



Gambar 2. Kriptografi Simetri

Untuk mempermudah penulisan dan analisis dalam kriptografi seringkali digunakan notasi matematika. Notasi matematika dari proses enkripsi dan proses dekripsi algoritma simetri digambarkan sebagai berikut :

Ek(P) = C

Dk(C) = P

Dk(Ek(P)) = P

dengan P adalah plainteks, C adalah cipherteks, E_k adalah proses enkripsi menggunakan kunci, dan D_k adalah proses dekripsi menggunakan kunci.

4. JARINGAN FEISTEL

Hampir semua algoritma cipher blok bekerja dalam model jaringan Feistel. Jaringan *Feistel* ditemukan oleh Horst Feistel tahun 1970. Model jaringan *Feistel* adalah sebagai berikut:

a. Bagi blok yang panjangnya *n* bit menjadi dua bagian, kiri (*L*) dan kanan (*R*), yang masing-masing panjangnya *n*/2 (hal ini mensyaratkan *n* harus genap).

b. Definisikan *cipher* blok berulang dimana hasil dari putaran ke-*i* ditentukan dari hasil putaran sebelumnya.

$$L_i = R_{i-1};$$

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

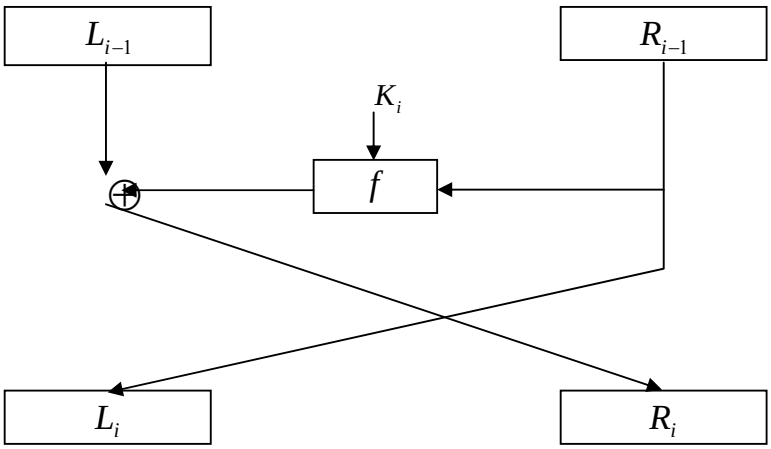
Yang dalam hal ini,

i = 1,2,.....,r (r adalah jumlah putaran).

K_i = upa-kunci (*subkey*) pada putaran ke-*i*

f = fungsi transformasi (di dalamnya terdapat fungsi substitusi, permutasi, dan/atau ekspansi, kompresi).

Plainteks adalah gabungan *L* dan *R* awal, atau secara formal dinyatakan dengan $\left(L_0, R_0\right)$, sedangkan cipherteks didapatkan dari *L* dan *R* hasil dari putaran terakhir setelah terlebih dahulu dipertukarkan, atau secara formal dinyatakan sebagai $\left(R_r, L_r\right)$.



Gambar 3. Jaringan *Feistel*

5. KOTAK-S (*S-BOX*)

Kotak-S adalah matriks yang berisi subtitusi sederhana yang memetakan satu atau lebih bit dengan satu atau lebih bit lainya. Pada kebanyakan algoritma cipher blok, kotak-S memetakan m bit masukan menjadi n bit keluaran, sehingga kotak-S tersebut dinamakan kotak m x n S-*box*. Kotak-S merupakan satu- satunya langkah nirlanjar di dalam algoritma, karena operasinya adalah *look-up table*. Masukan dari operasi *look-up table* dijadikan sebagai indeks kotak-S dan keluarannya adalah *entry* di dalam kotak-S. Perancangan kotak-S menjadi isu penting karena kotak-S harus dirancang sedemikian sehingga kekuatan kriptografinya bagus dan mudah diimplementasikan.

6. SUBTITUTION BOX

CAST-128 menggunakan delapan box (8x32 S-*Box*) yang digunakan untuk fungsi putaran (*round)* dan pengolahan kunci: s-box S1, S2, S3, dan S4 digunakan untuk fungsi putaran (*round)*. S-*box* S5, S6, S7, and S8 digunakan untuk fungsi pengolahan kunci. Karena S-*box* adalah kotak 8 x 32, kotak ini akan menerima 8 bit masukan dan mengeluarkan 32 *bit* keluaran. Meskipun 8 S-*box* ini memerlukan total 8 *Kbyte* memori, tetapi pada proses sebenarnya hanya memerlukan 4 *Kbyte* memori karena pengolahan kunci telah di proses di awal untuk semua jenis data input.

7. PASANGAN KUNCI *ROUND*

CAST-128 menggunakan sepasang kunci setiap *round* yaitu 32-bit kunci Km yang digunakan untuk kunci *masking* yaitu untuk menyamarkan bit dan 5-*bit* kunci Kr yang digunakan untuk kunci rotasi (memutar bit).

8. PENGOLAHAN KUNCI (PEMBANGKITAN KUNCI INTERNAL)

Karena CAST-128 menggunakan enambelas putaran feistel, dimana setiap putarannya, CAST-128 membutuhkan sepasang kunci internal, tentunya dibutuhkan kunci sebanyak 32 buah, yaitu enam belas buah (Km1..Km16) dan enam belas buah (Kr1..Kr16). Enam belas kunci Km1..Km16 digunakan untuk *masking* di setiap putaran, sedangkan Kr1..Kr16 digunakan untuk kunci rotasi. Kunci internal ini dibangkitkan dari kunci eksternal yang diberikan oleh pengguna. Jadi, dari kunci eksternal yang panjangnya 128 bit, dibentuk enam belas pasang kunci internal, yaitu pasangan 32 bit kunci *masking* dan 5 *bit* kunci rotasi.

Algoritma pembangkitan kunci internal diberikan dengan asumsi sebagai berikut, misalkan kunci eksternal sepanjang 128 bit dibagi menjadi 16 *bytes* upakunci, yaitu: x0x1x2x3x4x5x6x7x8x9xAxBxCxDxExF, Dimana x0 menunjukkan *most significant byte* (MSB) dan xF menunjukkan *least significant byte* (LSB).

Selain itu, digunakan z0..zF yang merupakan temporary *byte* untuk penyimpanan *byte* sementara. Digunakan juga Si[] yang merepresentasikan s-*box* ke-i.

Algoritma pembangkitan kunci internal diberikan sebagai berikut:

z0z1z2z3 = x0x1x2x3 ^ S5[xD] ^ S6[xF] ^ S7[xC] ^ S8[xE] ^ S7[x8]
z4z5z6z7 = x8x9xAxB ^ S5[z0] ^ S6[z2] ^ S7[z1] ^ S8[z3] ^ S8[xA]
z8z9zAzB = xCxDxExF ^ S5[z7] ^ S6[z6] ^ S7[z5] ^ S8[z4] ^ S5[x9]
zCzDzEzF = x4x5x6x7 ^ S5[zA] ^ S6[z9] ^ S7[zB] ^ S8[z8] ^ S6[xB]
K1 = S5[z8] ^ S6[z9] ^ S7[z7] ^ S8[z6] ^ S5[z2]
K2 = S5[zA] ^ S6[zB] ^ S7[z5] ^ S8[z4] ^ S6[z6]
K3 = S5[zC] ^ S6[zD] ^ S7[z3] ^ S8[z2] ^ S7[z9]
K4 = S5[zE] ^ S6[zF] ^ S7[z1] ^ S8[z0] ^ S8[zC]
x0x1x2x3 = z8z9zAzB ^ S5[z5] ^ S6[z7] ^ S7[z4] ^ S8[z6] ^ S7[z0]
x4x5x6x7 = z0z1z2z3 ^ S5[x0] ^ S6[x2] ^ S7[x1] ^ S8[x3] ^ S8[z2]
x8x9xAxB = z4z5z6z7 ^ S5[x7] ^ S6[x6] ^ S7[x5] ^ S8[x4] ^ S5[z1]
xCxDxExF = zCzDzEzF ^ S5[xA] ^ S6[x9] ^ S7[xB] ^ S8[x8] ^ S6[z3]

K5 = S5[x3] ^ S6[x2] ^ S7[xC] ^ S8[xD] ^ S5[x8]

K6 = S5[x1] ^ S6[x0] ^ S7[xE] ^ S8[xF] ^ S6[xD]

K7 = S5[x7] ^ S6[x6] ^ S7[x8] ^ S8[x9] ^ S7[x3]

K8 = S5[x5] ^ S6[x4] ^ S7[xA] ^ S8[xB] ^ S8[x7]

z0z1z2z3 = x0x1x2x3 ^ S5[xD] ^ S6[xF] ^ S7[xC] ^ S8[xE] ^ S7[x8]

z4z5z6z7 = x8x9xAxB ^ S5[z0] ^ S6[z2] ^ S7[z1] ^ S8[z3] ^ S8[xA]

z8z9zAzB = xCxDxExF ^ S5[z7] ^ S6[z6] ^ S7[z5] ^ S8[z4] ^ S5[x9]

zCzDzEzF = x4x5x6x7 ^ S5[zA] ^ S6[z9] ^ S7[zB] ^ S8[z8] ^ S6[xB]

K9 = S5[z3] ^ S6[z2] ^ S7[zC] ^ S8[zD] ^ S5[z9]

K10 = S5[z1] ^ S6[z0] ^ S7[zE] ^ S8[zF] ^ S6[zC]

K11 = S5[z7] ^ S6[z6] ^ S7[z8] ^ S8[z9] ^ S7[z2]

K12 = S5[z5] ^ S6[z4] ^ S7[zA] ^ S8[zB] ^ S8[z6]

x0x1x2x3 = z8z9zAzB ^ S5[z5] ^ S6[z7] ^ S7[z4] ^ S8[z6] ^ S7[z0]

x4x5x6x7 = z0z1z2z3 ^ S5[x0] ^ S6[x2] ^ S7[x1] ^ S8[x3] ^ S8[z2]

x8x9xAxB = z4z5z6z7 ^ S5[x7] ^ S6[x6] ^ S7[x5] ^ S8[x4] ^ S5[z1]

xCxDxExF = zCzDzEzF ^ S5[xA] ^ S6[x9] ^ S7[xB] ^ S8[x8] ^ S6[z3]

K13 = S5[x8] ^ S6[x9] ^ S7[x7] ^ S8[x6] ^ S5[x3]

K14 = S5[xA] ^ S6[xB] ^ S7[x5] ^ S8[x4] ^ S6[x7]

K15 = S5[xC] ^ S6[xD] ^ S7[x3] ^ S8[x2] ^ S7[x8]

K16 = S5[xE] ^ S6[xF] ^ S7[x1] ^ S8[x0] ^ S8[xD]

[Setengah bagian yang lain untuk membentuk K17 – K32 prosesnya identik dengan proses di atas, tergantung dari x0..xF yang terakhir dibentuk]

Dapat dilihat dari algoritma di atas bahwa pembangkitan kunci internal menggunakan operator XOR dan empat buah S-*box* yaitu S5, S6, S7, dan S8. Dengan algoritma ini didapat K1..K32.

9. *MASKING* DAN ROTASI SUB KUNCI

Dari K1..K32 yang dibangkitkan sebelumnya, enam belas kunci pertama Km1,..., Km16 adalah 32-bit sub kunci *masking* yang akan digunakan untuk kunci *masking* (satu kunci per putaran). Dan enam belas sisanya Kr1,..., Kr16 adalah 32-bit sub kunci rotasi yang digunakan untuk kunci rotasi. Untuk kunci rotasi, hanya 5 bit dari *least significant byte* yang digunakan. Algoritma penjadwalan kunci dberikan sebagai berikut: for (i=1; i<=16; i++) { Kmi = Ki; Kri = K16+i; }.

10. FUNGSI ENKRIPSI ALGORITMA CAST-128

Dari enam belas putaran feistel yang digunakan, CAST-128 menggunakan tiga jenis putaran yang berbeda. Tiga jenis putaran tersebut dibedakan menurut tiga tipe fungsi enkripsi yang berbeda.

Secara matematis ketiga tipe tersebut dinyatakan dengan:

Tipe 1: I = ((Kmi + D) <<< Kri)
f = ((S1[Ia] ^ S2[Ib]) - S3[Ic]) + S4[Id]

Tipe 2: I = ((Kmi ^ D) <<< Kri)
f = ((S1[Ia] - S2[Ib]) + S3[Ic]) ^ S4[Id]

Tipe 3: I = ((Kmi - D) <<< Kri)
f = ((S1[Ia] + S2[Ib]) ^ S3[Ic]) - S4[Id]

Jadwal pemakaian ketiga tipe fungsi tersebut dalam jaringan feistel diberikan di bawah ini:

Rounds 1, 4, 7, 10, 13, dan 16 menggunakan fungsi *round* tipe 1.

Rounds 2, 5, 8, 11, dan 14 menggunakan fungsi *round* tipe 2.

Rounds 3, 6, 9, 12, dan 15 menggunakan fungsi *round* tipe 3.

Dimana D adalah 32 bit data input, I adalah hasil operasi a terhadap D. I dibagi menjadi empat bagian sepanjang 8 bit Ia, Ib, Ic, dan Id terurut mulai dari *most significant byte* (MSB) sampai *least significant byte* (LSB). f adalah hasil fungsi enkripsi. Sebagai catatan, "+" dan "-"adalah penjumlahan dan pengurangan modulo 2^{32}, "^" adalah XOR, dan "<<<" adalah penggeseran bit ke kiri (*circular left-shift operation*).

11. ENKRIPSI DENGAN ALGORITMA CAST-128

CAST-128 memiliki blok data masukan sebagai plainteks berukuran 64-bit. Untuk data yang berukuran lebih besar dari 64-bit, data tersebut harus diubah menjadi kumpulan blok-blok data yang berukuran 64-bit dan akan diproses dengan kunci yang sama. CAST-128 menggunakan kunci dengan ukuran yang bervariasi dari 8-bit sampai 128-bit.

Algoritma lengkap dari CAST-128 terbagi menjadi 4 langkah yaitu :

Masukan : plainteks m1...mn; key K = k1...k128. kunci sepanjang 128 bit

Keluaran : cipherteks c1...cn.

a. Pengolahan kunci : Menghitung 16 pasang {Kmi, Kri} dari K

b. $(L_0, R_0) \leftarrow$ (m1...m64). (Membagi plainteks menjadi 2 yaitu 32 bit kiri L_0 = m1...m32 dan 32 bit kanan R_0 = m33...m64)

c. Menghitung Li dan Ri sebanyak 16 langkah yaitu :

$L_i = R_{i-1}$;

$R_i = L_{i-1} \wedge f(R_{i-1}, K_i)$, dimana f adalah 3 macam tipe perhitungan sebagai berikut:

Type 1 : I = ((Kmi + D) <<< Kri)
f = ((S1[Ia] ^ S2[Ib]) - S3[Ic]) + S4[Id]

Type 2 : I = ((Kmi ^ D) <<< Kri)
f = ((S1[Ia] - S2[Ib]) + S3[Ic]) ^ S4[Id]

Type 3 : I = ((Kmi - D) <<< Kri)
f = ((S1[Ia] + S2[Ib]) ^ S3[Ic]) - S4[Id]

d. c1...c64$\leftarrow$ (R_{16}, L_{16}) . (Menukarkan dan menggabungkan blok final L_{16}, R_{16} menjadi bentuk cipherteks).

Proses dari algoritma dekripsi hampir sama dengan algoritma enkripsi yang disebutkan di atas, kecuali langkah (termasuk sub kunci) yang digunakan dengan urutan terbalik (R_{16}, L_{16}) menuju (L_0, R_0)

12. KEUNGGULAN DAN KELEMAHAN ALGORITMA CAST-128

Keunggulan Algoritma *CAST* memiliki tingkat keamanan yang tinggi dikarenakan hal-hal sebagai berikut :

a. Adanya komponen s-box yang dirancang dengan prosedur matematik yang menghasilkan kotak substitusi dengan komponen kriptogtafi yang penting, rendahnya distribusi perbedaan XOR, pengurutan yang baik dengan menggunakan kriteria bit yang terpisah.

b. Penggunaan kunci "*masking*" dan kunci "*rotation*" menaikkan entropi kunci dibandingkan dengan entropi data pada setiap putaran. Hal ini menyebabkan sulit dilalukannya serangan statistic yang bersifat iteratif.

c. Gabungan operasi dari kelompok aljabar yang berbeda (penambahan modulo 2 dan penambahan/pengurangan modulo 2^{32}) yang muncul mengakibatkan algoritma ini

efektif tidak hanya dalam mengurangi probabilitas diferensial antar putaran tetapi juga mengurangi kemungkinan untuk serangan dengan deferensial tingkat tinggi

d. Jaringan feistel yang digunakan dan variasi fungsi untuk tiap putaran mengakibatkan susahnya konstruksi yang bersifat iteratif.

Disamping keunggulan yang dimiliki oleh Algoritma CAST-128, algoritma ini juga dicurigai memiliki jalan belakang karena menggunakan s-*box*.

13. ANALISIS SISTEM

Tahap analisis sistem dilakukan setelah tahap perencanaan sistem *(system planning)* dan sebelum tahap desain sistem *(system design)*. Analisis sistem berfungsi untuk mempelajari secara seksama suatu sistem yang sedang dijalankan.

13.1. CARA KERJA PROSES ENKRIPSI DAN DESKRIPSI

Cara kerja proses pengolahan enkripsi adalah :

1. 64-bit plainteks dibagi menjadi 2 bagian double word. (*L* dan *R*)
2. Jika panjang dari kunci kurang dari atau sama dengan 10 karakter (80-bit) maka plainteks akan diproses sebanyak 16 langkah.
3. Fungsi yang digunakan dalam proses enkripsi ada 3 macam yaitu : tipe 1, tipe 2, tipe 3. tiap langkah menggunakan fungsi yang ditentukan sebagai berikut:
   a. Langkah 1,4,7,10,13, dan 16 menggunakan fungsi tipe 1
   b. Langkah 2, 5, 8, 11, dan 14 menggunakan fungsi tipe 2
   c. Langkah 3, 6, 9, 12, dan 15 menggunakan fungsi tipe 3
4. Langkah terkahir dari proses enkripsi adalah menukar 2 bagian blok akhir (*R,L*), lalu digabungkan kembali menjadi cipherteks.

13.2. CARA KERJA PROSES PENGOLAHAN CIPHERTEKS

Cara kerja proses pengolahan cipherteks adalah :

a. Cipherteks dibagi-bagi menjadi cipherteks 64 bit.
b. Cipherteks dibagi menjadi 2 bagian yaitu 32-bit kiri dan 32-bit kanan.
c. Sebelum masuk ke dalam proses dekripsi ke 2 bagian blok ditukar dahulu.

13.3. CARA KERJA PROSES ENKRIPSI

Setelah proses pengolahan plainteks maka proses selanjutnya adalah proses enkripsi. Cara kerja program enkripsi adalah :

1. 64-bit cipherteks dibagi menjadi 2 bagian double word. (L dan R).
2. Jika panjang dari kunci kurang dari atau sama dengan 10 karakter (80-bit) maka plainteks akan diproses sebanyak 12 langkah. Jika panjang kunci lebih dari 10 karekater maka plainteks akan diproses sebany ak 16 langkah.
3. Fungsi yang digunakan dalam proses dekripsi ada 3 macam yaitu : tipe 1, tipe 2, tipe 3. tiap langkah menggunakan fungsi yang ditentukan sebagai berikut:
   a. Langkah 1,4,7,10,13, dan 16 menggunakan fungsi tipe 1
   b. Langkah 2, 5, 8, 11, dan 14 menggunakan fungsi tipe 2
   c. Langkah 3, 6, 9, 12, dan 15 menggunakan fungsi tipe 3
4. Proses dekripsi identik dengan proses enkripsi hanya urutan langkah yang dilakukan mulai dari langkah ke 16.
5. Langkah terakhir dari proses dekripsi adalah menggabungkan kembali ke 2 bagian blok akhir menjadi plainteks kembali.
6. Jika pada proses enkripsi terjadi proses *padding*, maka sebelum plainteks ditampilkan karakter *padding* dihilangkan terlebih dahulu.

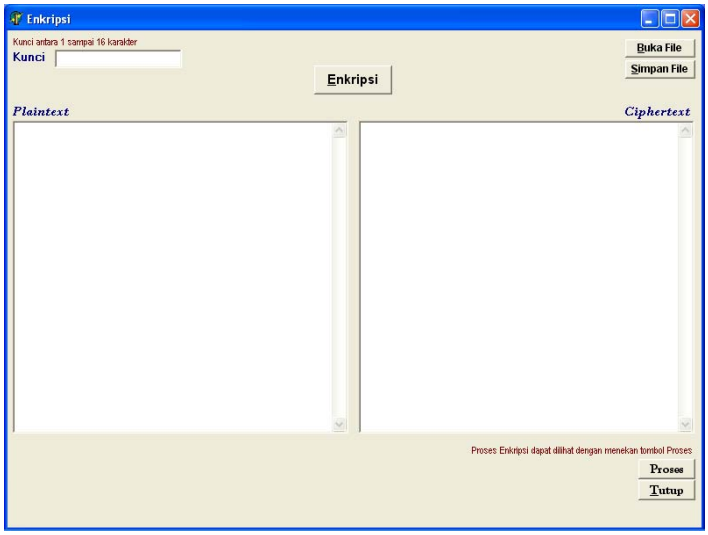
14. IMPLEMENTASI SISTEM

Setelah analisa sistem dan perancangan (desain) sistem secara detail. Tahap ini merupakan tahap untuk menerapkan sistem agar layak untuk dioperasikan.

Hasil penerapan program aplikasi ini uraian programnya sebagai berikut :

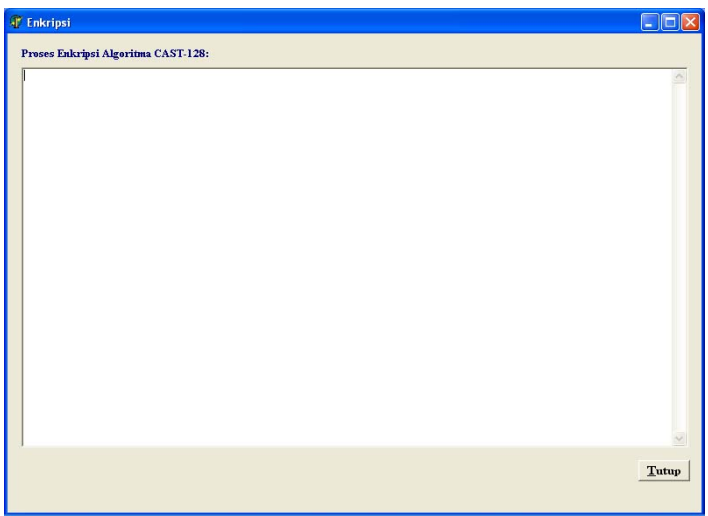
Pada Gambar 4 terlihat Menu Utama. Menu Utama adalah menu pembuka yang berisi kegiatan pengolahan data. Dalam menu Sistem Keamanan Data CAST-128 ini berisi dua menu yaitu Menu Enkripsi dan Menu Dekripsi.

Gambar 4. Implementasi Interface Menu Utama



Gambar 5. Implementasi Interface Enkripsi 1

Memo Plaintext untuk plainteks, dapat ditulis secara manual atau membuka *File* yang telah disimpan di Notepad dengan mengklik tombol Buka *File*. Penginputan kunci tidak dapat lebih dari 16 karakter, jika lebih dari 16 karakter maka karakter tersebut akan terhapus dan program akan mengeluarkan pesan "Kunci Maksimal 16 Karakter!". Setelah plainteks dan kunci terisi, maka dilanjutkan dengan meng-klik tombol Enkripsi atau menggunakan keyboard dengan menekan Alt+E. Setelah tombol Enkripsi di klik atau menekan Alt+E, maka pada *Memo Ciphertext* akan ditampilkan hasil dari Enkripsi yaitu berupa cipherteks. Setelah itu dilakukan penyimpanan hasil Enkripsi dengan mengklik tombol Simpan *File*. Tombol Proses dilakukan jika ingin melihat hasil dari langkah – langkah proses enkripsi. Tombol Tutup untuk menutup tampilan Enkripsi 1 (Gambar 5).

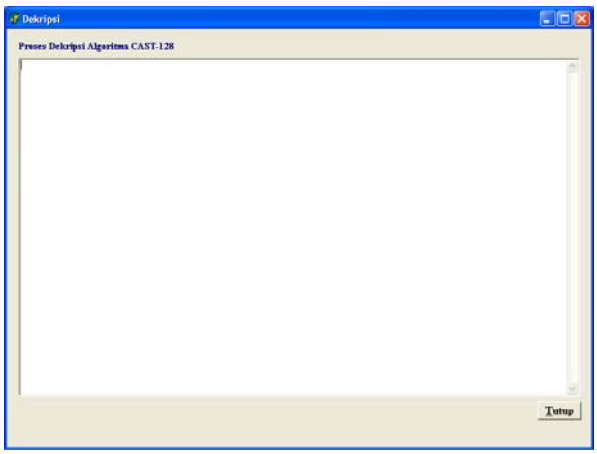


Gambar 6. Implementasi Interface Enkripsi 2

Di dalam *Memo* Proses Enkripsi berisi langkah-langkah proses enkripsi dengan algoritma CAST-128, sedangkan tombol Tutup untuk keluar dari form Enkripsi 2 (Gambar 6).

Pada Gambar 7 terlihat Interface Dekripsi 1. *Memo Ciphertext* untuk cipherteks, dilakukan dengan cara membuka *File* yang telah disimpan di Notepad dengan mengklik tombol Buka *File* (Alt+B). Penginputan kunci tidak dapat lebih dari 16 karakter, jika lebih dari 16 karakter maka karakter tersebut akan terhapus secara otomatis dan program akan mengeluarkan pesan "Kunci Maksimal 16 Karakter!". Setelah plainteks dan kunci telah terisi, maka dilanjutkan dengan meng-klik tombol Dekripsi atau menggunakan keyboard dengan menekan Alt+D. Setelah tombol Dekripsi di klik atau menekan Alt+D, maka pada *Memo Plaintext* akan ditampilkan hasil dari dekripsi yaitu berupa plainteks. Setelah itu dilakukan penyimpanan hasil dekripsi dengan mengklik tombol Simpan *File* (Alt+S). Tombol Proses dilakukan jika ingin melihat hasil langkah-langkah proses dekripsi. Tombol Tutup (Alt+T) untuk menutup tampilan dekripsi.

Gambar 7. Implementasi Interface Dekripsi 1



Gambar 8. Implementasi Interface Dekripsi 2

Di dalam *Memo* Proses Dekripsi berisi langkah-langkah proses dekripsi dengan algoritma CAST-128, sedangkan tombol Tutup untuk keluar dari form Dekripsi 2 (Gambar 8).

15. PENGUJIAN

Pengujian pada sistem ini menggunakan metoda ***black box testing*** yaitu pengujian dengan fokus pada persyaratan fungsional perangkat lunak yang telah dibangun. Pengujian dengan memberikan masukan-masukan tertentu sehingga dapat disimpulkan apakah fungsionalitas yang ada sesuai dengan yang dimaksudkan. Pengujian dilakukan dengan cara menjalankan perangkat lunak enkripsi dan dekripsi dan membuat hasil pengujian yang disajikan dalam tabel uji kualitas.

Hasil pengujian perangkat lunak enkripsi dan dekripsi dengan algoritma CAST-128, dalam lingkup interface enkripsi dan dekripsi algoritma CAST-128 dapat dilihat pada tabel 1.

Tabel Pengujian Interface Enkripsi dan Dekripsi Algoritma CAST-128

No	Materi	Hal YangDiharapkan	Valid	Tidak Valid	Hasil Nyata
1	**Menu Utama :** - Klik Menu Enkripsi - Klik Menu Dekripsi	 Masuk menu Enkripsi Masuk menu Dekripsi	 √ √		 Menu Enkripsi tampil Menu Dekripsi tampil
2	**Menu Enkripsi 1**				
	- isi Kotak Pengisian Kunci	Pengisian Kunci (Kunci antara 1 sampai dengan 16 karakter, jika lebih akan tampil pesan "Kunci Maksimal 16 Karakter!")	√		jika dimasukkan kunci lebih dari 16 karakter, dari program akan mengeluarkan pesan"Kunci Maksimal 16 Karakter!"
	- Klik Tombol Buka *File*	Untuk mengambil *File* text yang telah di simpan di dalam Notepad	√		Menu open *File* tampil
	- Klik Tombol Enkripsi	Untuk menghasilkan cipherteks	√		Hasil cipherteks tampil di *Memo Ciphertext*
	- Klik Tombol Simpan *File*	Untuk menyimpan hasil enkripsi	√		Menu *Save* tampil
	- isi *Memo Plaintext*	Tampilan text yang akan di enkripsi	√		Data plainteks tampil di *Memo Plaintext*
	- *Memo Ciphertext*	Tampilan hasil enkripsi (cipherteks)	√		Data cipherteks tampil di *Memo Ciphertext*, hasil dari proses enkripsi
	- Klik Tombol Proses	Untuk menuju Form Enkripsi 2	√		Form Enkripsi 2 tampil

	- Klik Tombol Tutup	Untuk menutup jendela Menu Form Enkripsi	√		Keluar dari Menu Form Enkripsi
	Menu Enkripsi 2				
	- Kotak *Memo* Proses Enkripsi	Untuk tampilan langkah – langkah Proses Enkripsi	√		Langkah – langkah Proses Enkripsi tampil
	- Klik Tombol Tutup	Untuk menutup jendela Menu Form Enkripsi 2	√		Keluar dari Menu Form Enkripsi 2
3	**Menu Dekripsi 1**				
	- Isi Kotak Pengisian Kunci	Untuk pengisian kunci. (kunci antara 1 sampai dengan 16 karakter, jika lebih maka akan keluar pesan "Kunci Maksimal 16 Karakter!")	√		jika dimasukkan kunci lebih dari 16 karakter, muncul pesan "Kunci Maksimal 16 Karakter!"
	- Klik Tombol Buka *File*	Untuk mengambil *File* hasil enkripsi (cipherteks)	√		Menu open *File* tampil
	- Klik Tombol Dekripsi	Untuk menghasilkan plainteks (sesuai dengan data aslinya)	√		Hasil plainteks (sesuai dengan data aslinya) tampil di *Memo Plaintext*
	- Klik Tombol Simpan *File*	Untuk menyimpan hasil dekripsi (plainteks)	√		Menu *Save* tampil
	- isi *Memo Ciphertext*	Tampilan text yang akan di dekripsi	√		Data cipherteks tampil di *Memo Ciphertext* , hasil dari buka *File*
	- *Memo Plaintext*	Tampilan hasil dekripsi (plainteks) , harus sesuai dengan data aslinya	√		Data plainteks tampil di *Memo Plaintext* sesuai dengan data aslinya, setelah proses dekripsi
	- Klik Tombol Proses	Untuk menuju Form Dekripsi 2	√		Form Dekripsi 2 tampil
	- Klik Tombol Tutup	Untuk menutup jendela Menu Form Dekripsi	√		Keluar dari Menu Form Dekripsi
	Menu Dekripsi 2				
	- Kotak *Memo* Proses Dekripsi	Untuk tampilan langkah – langkah Proses Dekripsi	√		Langkah – langkah Proses Dekripsi tampil
	- Klik Tombol Tutup	Untuk menutup jendela Menu Form Dekripsi 2	√		Keluar dari Menu Form Dekripsi 2

Setelah dilakukan pengujian interface enkripsi dan dekripsi diambil kesimpulan bahwa menu – menu , tombol – tombol yang ada di interface enkripsi dan dekripsi sudah dapat berjalan sesuai dengan harapan.

16. KESIMPULAN

Sedangkan kesimpulan yang diambil dari pengujian sistem yaitu:

a. Penggunaan kunci yang sama pada saat enkripsi dan dekripsi akan menghasilkan plainteks sesuai dengan aslinya
b. Penggunaan kunci yang tidak sama pada saat enkripsi dan dekripsi akan menghasilkan plainteks tidak sama dengan aslinya
c. Pengubahan beberapa *byte* pada cipherteks menyebabkan *byte-byte* lainnya di dalam dekripsi cipherteks menjadi rusak, sehingga isi file menjadi tidak terbaca lagi.
d. Dan di dalam CAST-128 tidak dapat menggunakan kunci lebih dari 128 bit.

17. DAFTAR PUSTAKA

1. Adams, C., *"The CAST-128 Encryption Algorithm"* **http://www.rfc-archive.org/getrfc.php?rfc=2144**, 1997.
2. Jogiyanto H.M., *Analisis dan Desain Sistem Informatika (Pendekatan Terstruktur Teori dan Praktek Aplikasi),* Edisi 2, Andi Offset, Yogyakarta, 1999.
3. M. Agus J. Alam (2001), *Borland Delphi 6.0*, Jakarta, PT. Elex Media Komputindo
4. Muh. Zaki Riyanto , "*Mengenal Kriptografi* ", **http://zaki.web.ugm.ac.id**
5. Muhammad Mursodo, "*Enkripsi Untuk Keamanan Data Pada Jaringan* ", **http://www.klik-kanan.com/fokus/enkripsi.html**
6. Onno W.Purbo, *Tony Wiharjito (2000) Keamanan Jaringan Komputer*. Penerbit PT Elex Media Komputindo, Jakarta, 2000.
7. Rinaldi Munir, *Kriptografi.* Penerbit Informatika, Bandung, 2006
8. Wikipedia (2006*), http://en.wikipedia.org/wiki/CAST-128*.