ANALISIS FEISTEL CIPHER SEBAGAI DASAR BERBAGAI ALGORITMA BLOCK CIPHER

Oleh : Alvin Susanto (13506087)
Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : alvin_punya@yahoo.co.id

Abstraksi
Makalah ini secara khusus akan membahas mengenai apa yang dimaksud dengan feistel cipher secara terperinci, mulai dari bagaimana sejarah feistel cipher yang terbagi atas sejarah bagaimana lahirnya feistel cipher beserta perkembagannya serta biografi singkat dari penemu feistel cipher, yaitu Horst Feistel. Selain itu makalah ini juga akan memaparkan bagaimana proses pembentukan feistel cipher, serangan terhadap feistel cipher, hingga perbandingan penggunaan feistel cipher pada berbagai algoritma kriptografi modern. Bagian yang disebutkan terakhir ini adalah hal utama yang akan dibahas oleh penulis dalam makalah ini, dimana algoritma kriptografi modern yang akan dibahas penggunaan feistel ciphernya di antaranya DES, Blowfish, Camellia, dan Triple DES.

Kata kunci : Feistel Cipher, DES, algoritma, kriptografi, kriptografi modern, enkripsi, dekripsi, jaringan feistel, kriptografer, kriptoanalis.

1. Pendahuluan

Ilmu kriptografi telah lahir dan berkembang sesuai dengan ditemukannya tulisan oleh peradaban manusia. Dewasa ini kriptografi telah secara luas diemplementasikan dalam berbagai bidang, terutama sebagai sarana untuk menjamin kerahasiaan suatu arsip. Selain itu kriptografi juga dapat dipergunakan sebagai sarana hiburan. Untuk mempermudah pengimplementasian kriptografi, dikembangkan berbagai algoritma kriptografi baru dengan tujuan mempermudah proses penyandian pesan. Sekarang ini telah banyak algortima kriptografi yang telah digunakan, baik yang dapat dipecahkan (breakable), maupun belum dapat dipecahkan (unbreakable). Algoritma tersebut dibagi-bagi berdasarkan golongan tertentu, seperti algoritma kriptografi klasik dan algoritma kriptografi modern. Algoritma kriptografi klasik pada umumnya hanya menggunakan penggantian huruf dengan angka, atau huruf dengan huruf, sehingga mudah untuk dipecahkan terutama dengan teknik yang disebut teknk analisis frekuensi. Untuk memperkuat proses penyandian pesan, muncullah algoritma kriptografi modern sebagai solusi keamanan penyandian pesan. Algoritma kriptografi modern pada umumnya telah menggunakan sistem bit, yang membuat cipherteks menjadi semakin sulit untuk dipecahkan. Salah satu jenis dari kriptografi modern, yaitu cipherblock, menggunakan feistel cipher atau feistel network sebagai dasar dari pembentukan berbagai algoritma yang ada, termasuk DES, Camelia, Blowfish, dan sebagainya. Feistel cipher ini banyak digunakan karena proses implementasinya yang cukup mudah, terutama dari segi proses enkripsi dan dekripsi yang hampir sama. Selain itu feistel cipher hingga saat ini masih sulit untuk dipecahkan.

2. Sejarah dan Penemu Feistel Cipher

Bagian dari makalah ini secara khusus akan membahas sisi nonteknis dari feistel cipher. Hal yang akan dibahas disini adalah mengenai bagaimana sejarah lahirnya feistel cipher dan biografi singkat dari penemu feistel cipher itu sendiri.

2.1 Sejarah dan Perkembangan Feistel Cipher

Feistel Cipher merupakan salah satu bentuk implementasi dari cipher blok, dimana feistel cipher ini dibentuk dengan menggunakan struktur yang simetris. Feistel cipher dikenal juga dengan nama feistel network atau feistel cipher. Nama feistel sendiri diambil dari kriptografer IBM, yaitu Horst Feistel, yang berkebangsaan Jerman.

Feistel cipher pertama kali digunakan secara komersial melalui algoritma Lucifer, yang didesain oleh Feistel sendiri dan Don Coppersmith.

Feistel sendiri menjadi populer setelah pemerintah AS menggunakan algoritma DES sebagai standar enkripsi. Berkat kepopuleran yang didapat feistel cipher pada saat digunakan dalam DES inilah feistel cipher menjadi banyak digunakan oleh algoritma kriptografi yang lainnya.

2.2 Biografi Horst Feistel

Horst Feistel dilahirkan pada tahun 1915 di kota Berlin, Jerman, namun pindah ke Amerika Serikat pada tahun 1934. Pada era 1950-an, Feistel bekerja pada Air Force Cambridge Research Center (AFCRC) on Identification Friend or Foe (IFF) milik pemerintah Amerika Serikat. Dia pernah bekerja di Lincoln Laboratory dalam universitas MIT, dan juga perusahaan MITRE. Pada sekitar tahun 1970-an, Feistel bekerja pada IBM. Disinilah ia memulai riset yang pada akhirnya melahirkan feistel cipher yang ia terapkan pada algoritma Lucifer dan DES. Horst Feistel meraih gelar sarjana

di MIT dan gelar maste di Harvard. Keduanya didapat dalam bidang ilmu fisika. Dalam kehidupan berkeluarga, Feistel menikah pada tahun 1945 dengan Leona, dimana mereka dianugerahi seorang putri yang kemudian diberi nama Peggy.

3. Feistel Cipher

Bagian dari makalah ini akan membahas mengenai sisi teknis dari cipher feistel. Bab ini akan mencakup skema dasar dari pembuatan feistel cipher sederhana dan berbagai prinsip yang dapat kita gunakan dalam rangka mendesain algoritma feistel cipher yang lebih baik, dalam arti lebih sulit untuk dipecahkan oleh para kriptoanalis

3.1 Skema Dasar Feistel Cipher

Saat ini feistel cipher banyak digunakan dalam berbagai skema cipher blok yang umum digunakan, salah satunya Data Encryption Standard(DES). Keuntungan dari feistel cipher adalah proses operasi enkripsi dan dekripsinya yang mirip atau hampir sama. Yang membuat proses enkripsi dan dekripsi berbeda adalah penggunaan kunci yang dibalik.

Berikut adalah langkah-langkah pembentukan skema umum dari feistel cipher atau lebih dikenal dengan sebutan jarigan feistel atau feistel networks:

1. Anggap F adalah fungsi enkripsi yang akan digunakan dan K adalah kunci yang akan digunakan dalam operasi dalam fungsi F.
2.

Anggap F adalah fungsi enkripsi yang akan digunakan dan K adalah kunci yang akan digunakan dalam operasi dalam fungsi F.

Langkah pertama yang diambil adalah membagi dua plainteks yang akan dienkripsi menjadi dua bagian dengan ukuran yang sama. Potongan pertama kita sebut sebagai L0, sedangkan potongan sisanya akan kita sebut sebagai R0. Lalu, untuk setiap ronde i = 0,1,2,...n, kita lakukan perhitungan sebagai berikut :

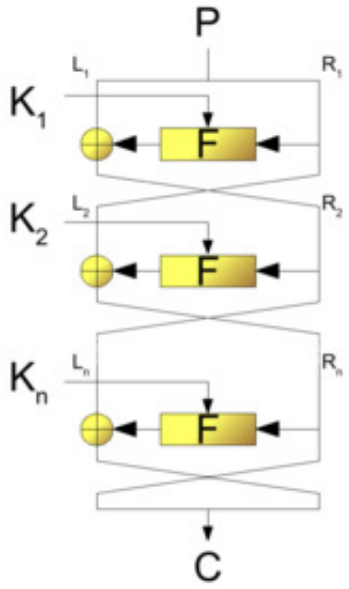
$$L_{i+1} = R_i$$
$$R_{i+1} = L_i \oplus \mathrm{F}(R_i, K_i).$$

Maka hasil dari cipherteksnya adalah (R_{n+1},L_{n+1}). Seperti yang telah diuraikan sebelumnya, proses dekripsi dapat dilakukan dengan langkah yang sama, hanya saja urutan kunci yang digunakan sebelumnya dibalik. Berikut adalah proses dekripsi umum yang digunakan dalam feistel cipher :

Kunci yang akan digunakan untuk setiap ronde adalah i = n,n-1,n-2,....0. Selanjutnya kita akan lakukan langkah berikut ini :

$$R_i = L_{i+1}$$
$$L_i = R_{i+1} \oplus \mathrm{F}(L_{i+1}, K_i).$$

Dari proses tersebut aan didapat bahwa hasil dari penggabungan L0 dan R0 adalah plainteks awal yang diberikan. Gambar di bawah ini menunjukkan bagaimana gambaran proses dari enkripsi berlangsung.



Skema yang diberikan di atas adalah skema jaringan feistel yang paling dasar dari skema yang ada. Untuk mempersulit kriptoanalis dalam memecahkan kode yang dihasilkan, kriptografer dapat menggunakan skema ECB (Electronic Code Book), CBC (Cipher Block Chaining), CFB (Cipher Feedback), dan sebagainya. Skema-skema ini tidak akan dibahas dalam makalah. Selain skema di atas, jaringan feistel juga dapat dilengkapi dengan prinsip diffusion, confusion, dan iterasi sebagai sarana untuk memperkuat keamanan cipherteks yang dihasilkan.

3.2 Prinsip Diffusion, Confusion, dan Iterasi

Prinsip diffusion dan confusion merupakan prinsip yang diterapkan dalam proses pembuatan algoritma enkripsi yang lebih aman. Prinsip ini dikemukakan oleh Claude Shannon pada tahun 1949. Jadi, sebelum feistel cipher ditemukan oleh Horst Feistel, prinsip diffusion dan confusion sendiri telah eksis. Kedua prinsip ini dianjurkan untuk digunakan bila kita ingin mendesain algoritma kriptografi modern yang sulit untuk dipecahkan.

Prinsip dari confusion adalah untuk menyembunyikan keterhubungan antara cipherteks, plainteks, dan kunci.

Prinsip dari diffusion adalah dengan menyebarkan pengaruh dari satu bit plainteks atau cipherteks atau kunci yang digunakan terhadap bit lainnya, sehingga kesalahan dari satu bit akan berpengaruh pada bit lainnya. Prinsip diffusion antara lain diterapkan dalam skema CFB dan CBC.

Prinsip iterasi sendiri telah dijelaskan sebelumnya pada bagian skema umum pembentukan jaringan feistel, dimana pada skema tersebut dilakukan sebanyak i ronde dimana nilai dari i tersebut merupakan jumlah iterasi yang digunakan. Semakin besar jumlah iterasi yang dilakukan, maka cipherteks yang dihasilkan akan semakin aman pula.

4. Perbandingan Feistel Cipher Pada Berbagai Algoritma Kriptografi Modern

Pada sesi ini, kita akan membahas berbagai penggunaan algoritma cipher feistel pada berbagai algoritma kriptografi modern. Yang akan dibahas disini bukanlah algoritma kriptografi-nya itu sendiri, melainkan dibatasi hanya penggunaan jaringan feistelnya saja dengan sedikit penjelasan beberapa algoritma yang menunjang penggunaan cipher feistel pada algoritma tersebut.

4.1 Lucifer Cipher

Lucifer cipher merupakan algoritma blok cipher pertama yang menggunakan feistel cipher. Algoritma ini ditemukan oleh Horst Feistel, dan menjadi cikal bakal algoritma DES. Dalam melakukan pengenkripsian, Lucifer cipher mengenkripsi 128 bits dari plainteks dan menggunakan kunci sebesar 128 bits juga.

Fungsi-f digunakan oleh lucifer cipher di dalam proses enkripsi di dalam jaringan feistel yang digunakan. Langkah-langkah fungsi tersebut:

1. Lakukan proses XOR setengah bagian kanan dari blok dengan delapan bit terakhir dari subkey ronde yang telah ditentukan.
2. Berdasarkan bit dari bit pertama dari subkey ronde tersebut, lakukan swap dalam delapan byte hasil tersebut untuk byte yang berkorespondensi dengan 1 bit.
3. Gunakan S-Box 0 untuk byte paling signifikan dari setiap 8 byte, dan S-Box 1 untuk setiap byte yang paling tidak signifikan.
4. Lakukan proses permutasi dari 64 bit hasil, dengan fungsi permutasi yang telah didefinisikan sebelumnya

Proses ini diulang sebanyak 16 ronde.

Dari data di atas, tampak bahwa algoritma lucifer, yang merupakan algoritma pertama yang memanfaatkan feistel cipher telah menggunakan fungsi permutasi dan iterasi dalam pengimplementasiannya. Fungsi-fungsi tersebut dibuat dengan tujuan untuk menyulitkan kriptoanalis dalam memecahkan cipherteks yang dihasilkan.

Skema dari penggunaan jaringan feistel pada lucifer cipher dapat dilihat pada gambar di bawah ini.

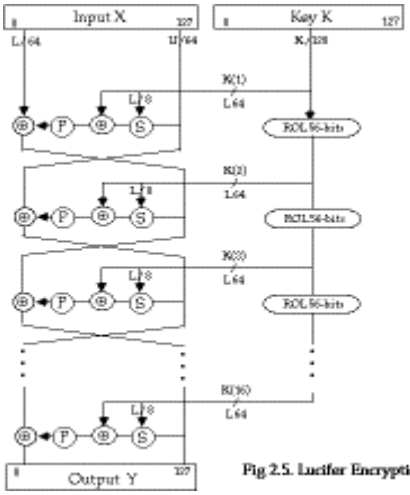


Fig 2.5. Lucifer Encryption Computation

4.2 DES

DES merupakan kependekan dari Data Encryption Standard. Algoritma DES merupakan algoritma kriptografi modern yang paling berjasa dalam mempopulerkan feistel cipher. Algoritma ini digunakan pemerintah Amerika Serikat sebagai standar enkripsi untuk menjamin kerahasiaan pesan pada tahun 1970-an. Algoritma ini sendiri dikembangkan oleh sang bapak feistel, yaitu Horst Feistel.

Algoritma DES menggunakan tiga proses yang dilakukan dalam pengenkripsian pesan, yaitu substitusi, permutasi, dan iterasi. Proses iterasi sendiri di dalam DES dilakukan sebanyak 16 kali, yang juga dikenal dengan sebutan 16 ronde. Gambar di bawah ini menunjukan skema dari algoritma DES.

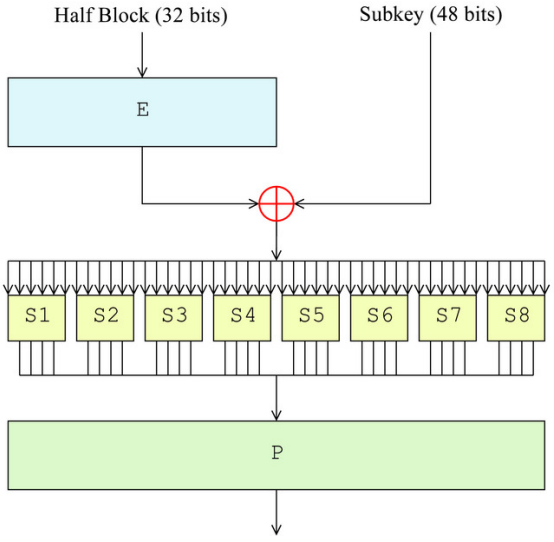


Diagram Feistel Functon oleh DES

Berikut adalah langkah-langkah inti pengenkripsian DES dengan menggunakan jaringan feistel :

1. 64 bit blok dari data input dibagi menjadi masing-masing 32 bit, dengan pembagian 32 bit paling kiri disebut sebagai L, dan pembagian 32 bit paling kanan kita sebut sebagai R. Langkah berikutnya (nomor 2 dan seterusnya) akan diulang sebanyak 16 kali, yang nantinya akan menghasilkan nilai L[0] - L[15] dan R[0] – R[15].
2. Misalkan I adalah bilangan ronde yang dimulai dari angka 1, maka R[I-1] dimasukan ke dalam Bit Selection Table, yang dalam hal ini dapat dibilang bahwa input mngalami **permutasi.** Proses ini akan mengekspand jumlah bit dari 32 bit menjadi 48 bit untuk digunakan pada langkah berikutnya.
3. Bit tersebut akan di XOR-kan dengan kunci dari langkah ke I.
4. Hasil dari langkah tersebut akan dibagi menjadi 8 bagian, dengan rincian dari tiap bagian mengandung 6 bit. 6 bit paling kiri dari hasil tersebut kita masukan sebagai B[1], sedangkan 6 bit paling kanan kita masukkan sebagai B[8]. Blok-blok ini akan dimasukan ke dalam S-Box, dimana di dalam S-Box ini, tiap blok akan mengalami proses **substitusi.** S-Box ini terdiri dari 8 buah array 2 dimensi, dengan jumlah baris sebanyak 4 buah dan jumlah kolom sebanyak 16 buah. Panjang bit dari box selalu sebesar 4 bit, sehingga jangkauan nilai mereka tersebar dari 0-15. Lalu S-Box tersebut diberi nama S[1] – S[8].
5. Dimulai dari B[1], bit pertama dan bit terakhir dari blok 6 bit akan diambil dan digunakan sebagai index untuk baris dari S[1], dengan jangkauan nilai dari 0 sampai 3, dan 4 bit di tengah digunakan sebagai index untuk kolom, dengan jangkauan nilai 0 sampai 15. Setelah itu, bilangan dari posisi ini disimpan. Hal ini terus dilakukan hingga B[8] dengan S[8]. Pada saat ini, kita aka mempunyai 8 buah bilangan 4 bit, yang bila digabungkan akan menghasikan 32 bit. Setelah itu, hasilnya kita masukkan ke dalam tabel permutasi P, untuk dilakukan **permutasi** lagi.

Tabel Permutasi P

Bit	0	1	2	3
1	16	7	20	21
5	29	12	28	17
9	1	15	23	26
13	5	18	31	10
17	2	8	24	14
21	32	27	3	9
25	19	13	30	6
29	22	11	4	25

6. Hasil yang didapat akan di-XOR dengan hasil dari L[I-1], dan dijadikan sebagai R[I]. Sedangkan nilai dari R[I-1] dipindahkan menjadi L[I].
7. Lakukan hal ini sebanyak 16 ronde, hingga kita nantinya mendapatkan nilai L[16] dan R[16]. Proses ini kita sebut sebagai **iterasi**. Langkah berikutnya adalah menukar nilai L dan R yang ada sehingga nilai L menjadi 32 bit paling kanan, dan nilai R menjadi 32 bit paling kiri. Hasilnya setelah digabung menjadi 64 bit dan berfungsi sebagai cipherteks.

Bisa kita lihat dari langkah-langkah di atas, bahwa DES menggunakan permutasi, substitusi, dan iterasi sebagai penunjang algoritma mereka.

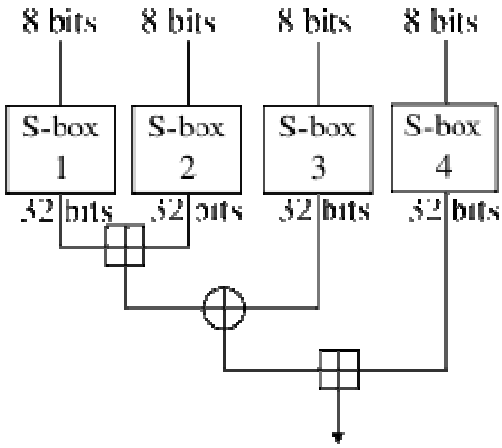
Selain dari tiga proses yang disebutkan diatas, DES juga menggunakan pemetaan nilai pada proses permutasi dengan proses yang dapat dibilang cukup rumit. Selain itu, DES juga menggunakan proses untuk mengurangi dan menambah jumlah bit yang juga tidak kalah rumitnya. Jelaslah terlihat pada langkah-langkah yang telah diuraikan tersebut, bahwa kriptoanalis akan sangat sulit untuk dapat memecahkan algoritma DES. Walaupun begitu, saat ini DES telah berhasil dipecahkan dengan algoritma brute force, walaupun untuk proses ini membutuhkan resource yang sangat besar.

4.3 Blowfish

Algoritma Blowfish diciptakan oleh Bruce Schneier pada tahun 1993. Sekarang ini, algoritma blowfish telah banyak mengalami perubahan atau perbaikan, dan diberi nama TwoFish. Akan tetapi, algoritma TwoFish tidak akan penulis bahas pada makalah ini. Seperti pada DES, blowfish juga menggunakan jaringan feistel sebanyak 16 ronde. Hal yang membedakan blowfish dengan DES adalah penggunaan proses modifikasi pada sisi kiri dan kanan pada setiap rondenya. Pada blowfish, penjadwalan kunci terdiri atas dua komponen, yaitu inisialisasi box-S dan inisialisasi dari array P. Nilai dari kedua komponen tersebut diisi dengan hasil ekspansi bagian dari pi (3.14....) secara hexadesimal. Berikut adalah langkah pertama yang dilakukan oleh algoritma blowfish pada ronde pertama :

- $R1 = L0 \oplus P1;$
- $L1 = R0 \oplus F(L0 \oplus P1)$

Fungsi dari tiap ronde dari blowfish memerlukan empat buah S-Box yang digunakan untuk mengembangkan 4 bit input menjadi 32 bit output. Hasil dari setiap keluaran S-Box didesain demikian rupa sehingga menghasilkan satu buat output dengan ukuran 32 bit juga. Skema dari fungsi tiap ronde algoritma blowfish akan digambarkan sebagai berikut :



Dengan keterangan :

⊕ menandakan fungsi XOR, dan

Segiempat bersilang menandakan proses adisi dengan mod 2^32.

Inti dari algoritma blowfish terletak pada proses pencampuran fungsi XOR dengan adisi seperti yang telah ditunjukan pada gambar diatas. Proses ini akan sangat menyulitkan para kriptoanalis.

Beberapa keuntungan dalam penggunaan blowfish adalah :

1. Kecepatan dalam penggunaan algoritma blowfish membuatnya menjadi pilihan yang baik untuk aplikasi yang membutuhkan proses enkripsi dengan jumalh yang besar dalam waktu yang singkat
2. Belum ada serangan pada blowfish yang benar-benar berhasil bila menggunakan mode 16 ronde secara penuh.

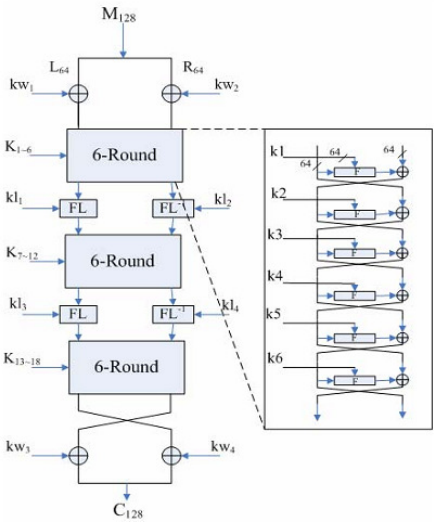
Adapun kekurangan dari algoritma blowfish :

1. Kebutuhan memori yang tinggi dapat menjadi penghambat penggunaan blowfish pada lingkungan yang terbatas
2. Ukuran blok yang hanya 64 bit sangat riskan untuk proses enkripsi dalam jumlah data yang sangat besar namun menggunakan kunci yang sama.

Di samping berbagai kekurangan yang telah disampaikan tersebut, fakta bahwa algoritma blowfish masih belum dapat dipecahkan oleh kriptoanalis menjadi salah satu alasan bahwa blowfish masih digunakan secara luas.

4.4 Camellia

Camellia merupakan salah satu algoritma kriptografi modern yang sangat populer, dan banyak digunakan secara luas. Algoritma ini diciptakan oleh NTT dan Mitsubishi pada tahun 2000, dan telah digunakan oleh proyek NESSIE milik Uni Eropa dan CRYPTREC dari Jepang.



Gambar proses enkripsi dari Camellia

Camellia menggunakan ukuran blok sebesar 128 bit, dan menggunakan ronde jaringan feistel sebanyak 18 atau 24 kali. Jumlah ronde yang digunakan sangat tergantung dari ukuran kunci. Bila ukuran kunci yang digunakan adalah 128 bits, maka kita menggunakan 18 ronde jaringan feistel. Sedangkan bila ukuran kunci 192 bit atau 256 bit, kita menggunakan 24 ronde jaringan feistel.

Proses yang membuat Camellia sulit dipecahkan terletak pada proses transformasi logis, yang dilakukan setiap 6 ronde, yang dikenal dengan fungsi FL dan inversenya.

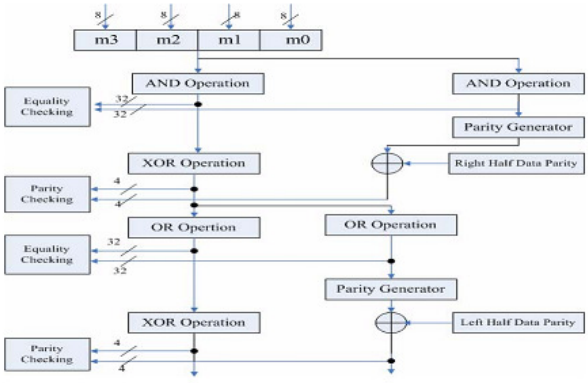


Fig. 5 FL Function with CED

Selain itu, Camellia juga menggunakan empat buah 8x8 bit Box-S dan menggunakan input dan output dari proses key whitening. Kedua hal ini juga menjadi salah satu faktor mengapa Camellia sulit untuk dipecahkan.

4.5 Triple DES

Sesuai dengan namanya, keunikan dari algoritma Triple DES adalah penggunaan algoritma DES sebanyak tiga kali. Hal ini dilakukan untuk

menghindari meet ini the middle attack yang seringkali terjadi pada double DES.

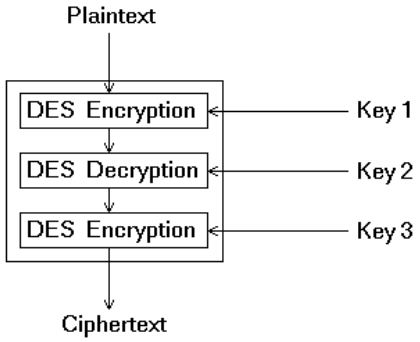
Secara umum proses enkripsi dari triple DES adalah sebagai berikut :

$C = E_{K3}(E_{K2}(E_{K1}(P)))$

Sedangkan proses dekripsinya sebagai berikut :

$P = D_{K1}(D_{K2}(D_{K3}(C)))$

Secara umum, penggunaan jaringan feistel pada triple DES serupa dengan algoritma DES, hanya saja pada triple DES, kita menggunakan dua atau tiga buah kunci sebagai sarana agar cipherteks sulit untuk dipecahkan.



Skema dari Triple DES

5. Serangan terhadap feistel cipher

Hingga saat ini, feistel cipher dapat dikatakan sebagai algoritma kriptografi yang sangat sulit untuk dipecahkan, walaupun ada beberapa serangan yang mengklaim dapat memecahkan feistel cipher. Dua contoh dari algoritma yang menyatakan dapat memecahkan feistel cipher adalah birthday attack dan kriptanalisis bilinear, yang skemanya tidak akan dibahas di makalah ini.

Selain itu untuk beberapa kasus, misalnya pada algoritma DES, serangan dapat berupa brute force attack, dimana proses tersebut memang membutuhkan waktu dan resource yang besar.

Walaupun demikian, secara umum feistel cipher merupakan algoritma kriptografi yang dapat dibilang sangat kuat dan masih dianjurkan untuk digunakan. Tentunya untuk mengurangi tingkat serangan terhadap feistel cipher, kita dapat mendesain feistel cipher yang lebih rumit dari standar bentuk baku yang ada.

6. Kesimpulan

Saat ini telah berkembang sangat banyak algoritma kriptografi modern beserta berbagai variannya. Walaupun begitu, feistel cipher atau jaringan feistel masih banyak digunakan sebagai patokan bagi para kriptografer dalam mendesain algoritmanya. Bahkan, desain jaringan feistel dalam setiap algoritma yang diberikan seolah menjadi seni bagi para kriptografer untuk merancang algoritma kriptografi yang tidak terpecahkan.

Alasan pemilihan digunakannya jaringan feistel :

1. Proses enkripsi dan dekripsi yang kurang lebih sama
2. Tingkat keamanan feistel cipher yang tinggi
3. Dapat dikombinasikan dengan berbagai teknik untuk menyulitkan kriptoanalis, seperti substitusi, transposisi, hingga perubahan ukuran bit

Daftar Pustaka

1. Munir, Rinaldi, Diktat Kuliah IF5054 Kriptografi, Departemen Teknik Informatika Institut Teknologi Bandung, 2006
2. Slide kuliah IF5054 oleh Rinaldi Munir
3. http://en.wikipedia.org/wiki/Feistel_cipher
4. www.math.uic.edu/~leon/mcs425-s08/handouts/feistal-diagram.pdf
5. www.quadibloc.com/crypto/co0401.htm
6. www.faqs.org/rfcs/rfc4312.html
7. http://en.wikipedia.org/wiki/Blowfish_(cipher)