Perbandingan Algoritma LOKI, GOST, dan Blowfish yang Merupakan Kelompok dari Algorima Block Cipher

Yongke Yoswara - 13508034
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
if18034@students.if.itb.ac.id

***Abstrak*—Saat ini begitu banyak algoritma kriptografi modern. Algoritma kriptografi modern beroperasi dalam mode bit (berbeda dengan algoritma kriptografi klasik yang beroperasi pada mode karakter). Ada 2 jenis algoritma yang beroperasi pada mode bit, yaitu *stream cipher* dan *block cipher*. Dalam *block cipher*, ada 5 prinsip perancangan, yaitu : Prinsip Confusion dan Diffusion, Cipher berulang, Jaringan Feistel, Kunci lemah, Kotak-S. Jaringan Feistel banyak dipakai pada algoritma DES, LOKI, GOST, Feal, Lucifer, Blowfish, dan lain-lain karena model ini bersifat *reversible* untuk proses dekripsi dan enkripsi. Sifat reversible ini membuat kita tidak perlu membuat algoritma baru untuk mendekripsi cipherteks menjadi plainteks.**

Kata Kunci—block cipher, jaringan feistel, reversible

I. PENDAHULUAN

Sekarang ini, teknologi berkembang sangat pesat. Hal serupa juga terjadi di dunia informatika. Perkembangan dunia informatika sangat cepat. Kemajuan dunia informatika juga mengakibatkan semakin berkembangnya teknologi informasi. Salah satu hal yang penting dalam teknologi informasi adalah nilai dari informasi itu sendiri. Nilai dari sebuah informasi sangat penting untuk sebagian orang yang tidak ingin isi dari informasi tersebut diketahui oleh orang yang tidak berhak melihatnya. Informasi itu disebarkan melalui media komunikasi yang tidak dapat kita ketahui dengan pasti keamanannya. Banyak media komunikasi yang tidak dapat menjamin keamanan data yang disalurkan melalui media tersebut. Contoh media komunikasi yang berkembang pesat sekarang ini adalah internet.

Oleh karena tidak adanya kepastian tentang keamanan suatu informasi yang disalurkan melalui media komunikasi, maka diperlukan metode yang dapat digunakan untuk menjaga isi dari informasi tersebut. Metode yang dimaksudkan adalah kriptografi. Kriptografi adalah sebuah seni dan bidang keilmuan dalam penyandian informasi atau pesan dengan tujuan menjaga keamanannya. Kriptografi sudah ada sejak jaman dahulu, namun seiring dengan berkembangnya teknologi, maka krptografi ini juga semakin berkembang. Perkembangan kriptografi ini dimakudkan supaya tidak ada yang bisa memecahkannya.

Awal mulanya, kriptografi beroperasi dalam mode karakter. Namun dengan semakin banyaknya penggunaan komputer dijital, maka kriptografi modern beroperasi dalam mode bit (satuan terkecil dalam dunia dijital). Walaupun menggunakan mode bit, algoritma dalam kriptografi modern tetap menggunakan prinsip substitusi dan transposisi yang sebenarnya telah digunakan sejak kriptografi klasik.

Algoritma kriptografi modern dapat dibagi menjadi dua, yaitu algoritma simetri dan kunci public. Algoritma simetri adalah algoritma yang menggunakan kunci yang sama untuk proses enkripsi dan dekripsi. Algoritma simetri yang beroperasi pada mode bit itu sendiri dapat dibagi menjadi 2 jenis, yaitu stream cipher dan block cipher. Stream cipher adalah algoritma kriptografi yang beroperasi dalam bentuk bit tunggal. Sedangkan block cipher beroperasi dalam bentuk blok-blok bit.

Ada banyak jenis algoritma yang menggunakan stream cipher dan block cipher. Untuk stream cipher, beberapa algoritma terkenal di antaranya adalah RC4, Seal, A5, Oryx, dll. . sedangkan dalam block cipher, beberapa algoritma terkenal di antaranya DES, Blowfish, Twofish, LOKI, GOST, dan masih banyak lagi.

GOST merupakan algoritma block cipher yang beroperasi pada mode 64 bit. Panjang kunci dalam algoritma ini adalah 256 bit.

Ada beberapa macam versi untuk algoritma LOKI, di antaranya adalah LOKI89, LOKI91, LOKI97. Sejak pertama kali dibuat, LOKI terus diperbaiki sehingga menghasilkan beberapa versi dari algoritma LOKI. LOKI menggunakan blok data 64 bit dan kunci sepanjang 64 bit pula.

Algoritma Blowfish adalah sebuah algoritma yang ditujukan untuk diimplementasikan pada sebuah mikroprosesor berskala besar. Algoritma blowfish sendiri tidak dipatenkan sehingga dapat digunakan oleh orang banyak. Algoritma ini beroperasi dengan ukuran blok sebesar 64 bit. Kunci yang digunakan dalam algoritma Blowfish sepanjang 32 sampai 388 bit.

II. CIPHER BLOCK

Cipher block adalah algoritma kriptografi modern yang beroperasi dalam brntuk blok bit. Biasanya plainteks akan dibagi ke dalam blok-blok bit yang mempunyai panjang yang sama. Misalnya, kita akan melakukan proses enkripsi pada sebuah plainteks yang panjangnya 64

bit, maka ketika kita menggunakan cipher block, akan dihasilkan block cipherteks sepanjang 64 bit juga. Begitu juga dengan proses dekripsi. Apabila kita memiliki sebuah cipherteks sepanjang 128 bit yang terbagi ke dalam blok-blok kecil, maka akan dihasilkan plainteks dari proses dekripsi dengan panjang 128 bit. Kunci yang digunakan untuk melakukan enkripsi dan dekripsi mempunyai panjang yang samadengan blok bit plainteks.

Proses enkripsi dengan kunci K dinyatakan secara formal dengan persamaan

$$E_K(P) = C$$

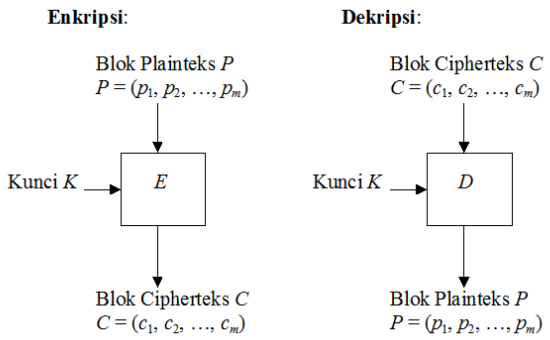
Sedangkan persamaan yang menyatakan proses dekripsi dengan kunci K adalah

$$D_K(C) = P$$

Fungsi E yang digunakan dalam proses enkripsi harus merupakan fungsi yang berkorespondensi satu-satu dengan fungsi D, sehingga

$$E^{-1} = D$$

Skema enkripsi dan dekripsi pada block cipher dapat dilihat pada gambar 1



Gambar 1 Gambar Proses Enkripsi dan Dekripsi pada Block Cipher

Pada block cipher, plainteks dibagi menjadi beberapa blok dengan panjang yang sama, yaitu sama dengan panjang kunci yang akan digunakan. Oleh sebab itu, ada kemungkinan sebuah plainteks tidak habis dibagi dengan panjang kunci / panjang ukuran blok yang ditetapkan. Hal ini mengakibatkan blok terakhir memiliki panjang lebih pendek daripada blok lainnya. kita dapat memberikan solusi, yaitu dengan *padding*. Yang dimaksud dengan *padding* adalah dengan menambahkan blok terakhir dengan pola bit yang teratur hingga panjang blok terakhir itu sama dengan panjang blok lainnya. Pola bit teratur ini misalnyaditambahkan bit 0 semua, atau bit 1 semua, atau bit 0 dan 1 secara bergantian.

Ukuran blok yang digunakan umumnya 64 bit atau 128 bit. Akan tetapi ada beberapa algoritma block cipher yang menggunakan ukuran blok yang berbeda. Ukuran blok 64 bit biasanya digunakan untuk desain yang dirancang sampai pertengahan tahun1990. Desain-desain yang baru umumnya menggunakan 128 bit. Ukuran kunci yang umumnya digunakan adalah 40, 56, 64, 80, 128, 192, dan 256 bit.

Pada algoritma kriptografi yang beroperasi dalam mode bit ini, dikenal empat mode operasi dasar yang sering digunakan untuk proses enkripsi dan dekripsi. Empat mode tersebut adalah :

1. Electronic Code Book (ECB)
2. Cipher Block Chaining (CBC)
3. Cipher Feedback (CFB)
4. Output Feedback (OFB)

Penggunaan mode tersebut tidak akan mengubah fungsi enkripsi dan dekripsi yang telah ada.

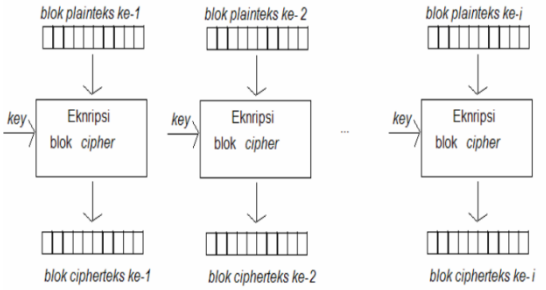
II.1 ELECTRONIC CODE BOOK (ECB)

Mode ECB merupakan mode yang paling sederhana. Pada mode ECB, setiap mode plainteks Pi dienkripsi masing-masing dan terpisah menjadi cipherteks Ci. Secara matematis,proses enkripsi dengan mode ini adalah sebagai berikut :

$$Ci = E_K(Pi)$$

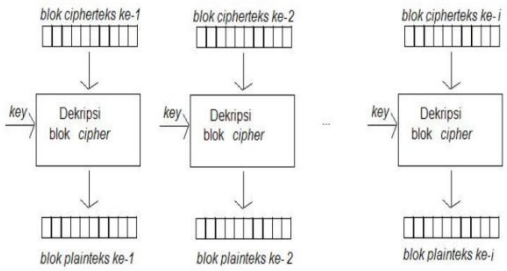
Dan proses dekripsi sebagai berikut ;

$$Pi = D_K(Ci)$$



Gambar 2 Gambar Proses Enkripsi pada Mode ECB

Dalam hal ini, Pi dan Ci merupakan blok plainteks dan cipherteks ke – i.



Gambar 3 Gambar Proses Dekripsi pada Mode ECB

Keuntungan dari mode ini adalah kesalahan satu bit pada satu blok hanya akan mempengaruhi blok cipherteks yang berkoresponden dengan blok plainteks tersebut.

Kelemahan mode ini adalah sebuah plainteks akan dienkripsi menjadi sebuah cipherteks yang identic. Dengan demikian, mode EBC ini tidak dapat menyembunyikan pola dari data. Hal ini mengakibatkan mode EBC tidak mendukung message confidentially. Oleh karena itu mode ini tidak disarankan untuk digunakan sebagai protokol kriptografi.

II.2 CIPHER BLOCK CHAINING (CBC)

Pada mode CBC, ada mekanisme umpan balik pada sebuah blok, yaitu blok plainteks sekarang current) akan

di-XOR-kan dahulu dengan blok cipherteks hasil enkripsi sebelumnya. Selanjutnya, hasil operasi XOR ini dimasukkan ke dalam fungsi enkripsi. Jadi, pada mode CBC, setiap blok cipherteks tidak hanya bergantung pada plainteksnya, tetapi juga bergantung dengan blok plainteks lainnya. Proses dekripsi dilakukan dengan cara memasukkan blok cipherteks current ke dalam fungsi dekripsi, kemudian meng-XOR-kan hasilnya dengan blok cipkerteks sebelumnya.

Untuk blok pertama, digunakan Initial Vector(IV) sebagai pengganti cipherteks sebelumnya. Initial Vector dapat diberikan oleh pengguna atau dibangkitkan secara acak oleh aplikasi. Dengan adanya Intitial Vector ini maka pesan menjadi unik.

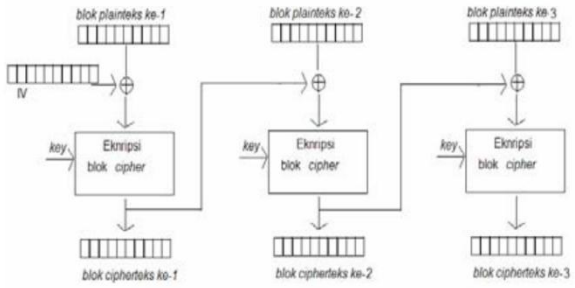
Secara matematis, proses enkripsi pada mode CBC adalah sebagai berikut :

$$Ci=E_K(Pi \oplus Ci-1)$$

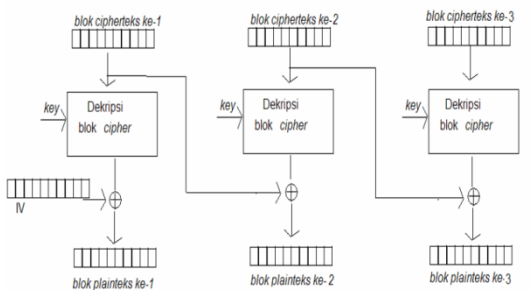
Sedangkan proses dekripsinya dapat dinyatakan dengan

$$Pi=D_K(Ci) \oplus Ci-1$$

Dengan mode CBC, kesalahan pada satu bit plainteks akan mempengaruhi cipherteks berikutnya. Sedangkan kesalahan dalam satu bit cipherteks hanya akan mempengaruhi satu blok plainteks.



Gambar 4 Gambar Proses Enkripsi pada Mode CBC



Gambar 5 Gambar Proses Dekripsi pada Mode CBC

II.3 CIPHER FEEDBACK (CFB)

Mode CBC memiliki kelemahan, yaitu proses enkripsi hanya dapat dilakukan pada ukuran blok yang utuh sehingga mode ini tidak efisien apabila diterakan pada aplikasi komunikasi data. Melihat permasalahan ini, maka mode berikutnya, yaitu CFB dapat mengatasinya. Mode ini melakukan enkripsi data dalam unit yang lebih kecil daripada ukuran blok.

Proses enkripsi pada unit yang lebih kecil daripada ukuran blok ini membuat mode CFB berlaku seperti stream cipher. Karena dapat berlaku dalam stream cipher, mode ini dapat digunakan dalam komunikasi data.

Apabila unit yang dienkripsi berupa satu karakter setiap kalinya, maka mode CFB ini disebut dengan CFB 8-bit. Mode ini membutuhkan sebuah antrian yang berukuran sama dengan ukuran blok masukan. Secara formal, proses enkripsi mode CFB n-bit dapat dinyatakan sebagai berikut.

$$Ci= Pi \oplus MSB_m (E_K(Xi))$$
$$Xi+1= LSB_{m-n}(Xi) \,||\, Ci$$

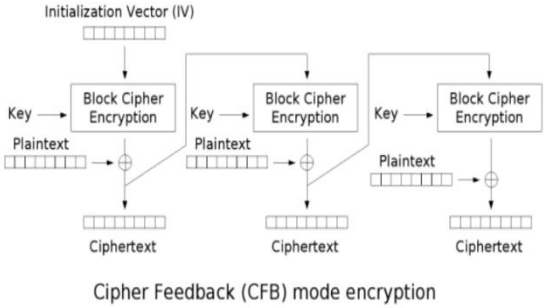
Sedangkan proses dekripsi dapat dinyatakan sebagai berikut :

$$Pi= Ci \oplus MSB_m(D_K(Xi))$$
$$Xi+1= LSB_{m-n}(Xi) \,||\, Ci$$

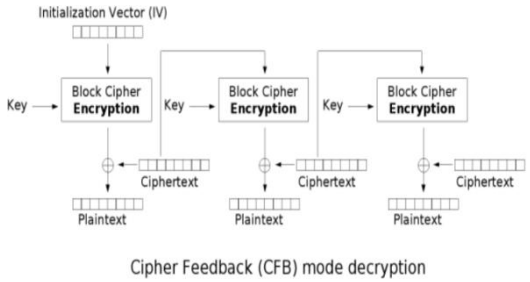
Keterangan rumus tersebut adalah sebagai berikut :

Xi = isi antrian dengan X1 adalah IV
E = fungsi enkripsi
K = kunci
M = panjang blok enkripsi
N = panjang unit enkripsi
|| = operator penyambungan (concatenation)
MSB = Most Significant Byte
LSB = Least Significant Byte

Mode CFB memiliki keunikan sendiri, yaitu untuk proses enkripsi dan dekripsi digunakan fungsi yang sama. Skema enkripsi dan dekripsi dengan mode CFB 8-bit dapat dilihat pada gambar berikut :



Gambar 6 Gambar Proses Enkripsi pada Mode CFB



Gambar 7 Gambar Proses Dekripsi pada Mode CFB

Mode ini juga menggunakan Initial Vector (IV). Kelemahan dari mode ini adalah jika terdapat kesalahan pada satu buah bit yang dienkripsi, maka bit-bit selanjutnya juga akan salah. Hal ini juga berlaku untuk proses dekripsi.

II.4 OUTPUT FEEDBACK (OFB)

Mode operasi ini digunakan apabila kesalahan propagasi sama sekali harus dihindari. Hampir mirip dengan CFB, kecuali n-bit dari hasil fungsi enkripsi terhadap antrian disalin menjadi elemen paling kanan antrian. Secara formal, proses enkripsi mode OFB n-bit dapat dinyatakan sebagai berikut :

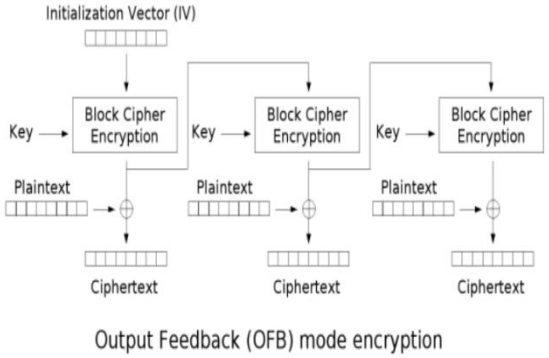
$$Ci = Pi \oplus MSB_m(E_K(Xi))$$
$$Xi+1 = LSB_{m-n}(Xi) \,||\, MS_m(E_K(Xi))$$

Sedangkan proses dekripsi dapat dinyatakan sebagai berikut :

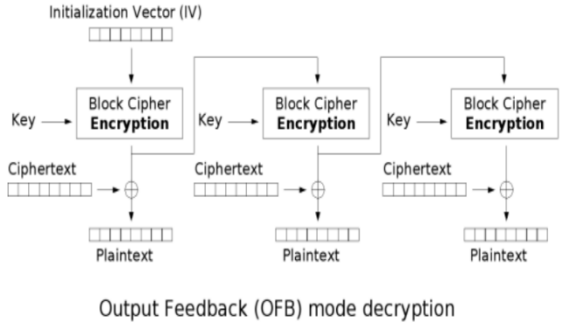
$$Pi = Ci \oplus MSB_m(D_K(Xi))$$
$$Xi+1 = LSB_{m-n}(Xi) \,||\, MSB_m(E_K(Xi))$$

Dengan mode ini, tidak terdapat perambatan kesalahan. Kesalahan satu bit pada plainteks hanya akan mengakibatkan kesalahan satu bit pada cipherteks yang berkoresponden. Sebaliknya kesalahan satu bit cipherteks hanya mengakibatkan kesalahan satu bit yang berkoresponden pada plainteks.

Perbedaan mode ini dengan mode CFB adalah jika pada CFB yang dimasukkan ke dalam antrian adalah hasil enkripsi, maka pada OFB yang dimasukkan ke dalam antrian adalah elemen antrian terdepan yang telah dikenakan fungsi enkripsi.



Gambar 8 Gambar Proses Enkripsi pada Mode OFB



Gambar 9 Gambar Proses Dekripsi pada Mode OFB

III. ALGORITMA LOKI

Algoritma LOKI memiliki 3 jenis, yaitu LOKI89, LOKI91, dan LOKI97. Setiap LOKI umumnya terdiri dari permutasi, ekspansi, pembangkitan kunci yang dilakukan sebanyak 16 putaran, pembagian blok kunci dan blok masukan menjadi dua buah blok yang sama, serta penggunaan S-Box untuk pembangkitan kunci.

Pada dasarnya keseleruhan proses dalam algoritma LOKI adalah sama, khusunya LOKI89 dan LOKI91. Perbedaan antara kedua LOKI tersebut adalah pada pendeklarasian fungsi f.

Dalam makalah ini, akan dibahas lebih lanjut mengenai LOKI97. LOKI97 merupakan LOKI yang lebih baru dibandingkan dengan LOKI89 dan LOKI91.

III.1 LOKI97

Algoritma ini merupakan algoritma blok cipher yang menggunakan kunci privat yang mengoperasikan data sebesar 128 bit dengan kunci 256 bit. LOKI97 merupakan pengembangan dari LOKI89 dan LOKI91. Algoritma ini melakukan evolusi dengan memperkuat penjadwalan kunci dan memperbesar ukuran kunci yang digunakan. LOKI97 diajukan sebagai kandidat untuk menggantikan DES (Data Encryption Standard)

Struktur dan desain dari LOKI97 dibuat oleh Dr. Lawrie Browns. Sedangkan Prof. Pieprzyk mendesain S-Box yang digunakan pada LOKI97.

III.1.1 SPESIFIKASI LOKI97

LOKI97 melakukan proses enkripi dan dekripsi data 128 bit. Kunci yang digunakan bisa sepanjang 128, 192, atau 256 bit.

LOKI97 merupakan algoritma blok cipher, sehingga data keluaran akan menghasilkan panjang yang sama dengan data masukan, yaitu 128 bit juga.

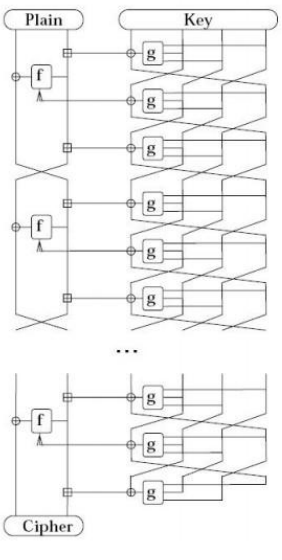
Proses pertama yang dilakukan dalam melakukan enkripsi adalah dengan membagi blok masukan menjadi dua buah blok yang sama panjang (blok L dan blok R).

Setelah data masukan tersebut dibagi dua bagian yang sama besar, dilakuka proses putaran sebanyak 16 kali dengan menggunakan Jaringan Feistel. Proses yang dilakukan dalam setiap kali putaran adalah sebagai berikut :

$R_i = L_{i-1}$ XOR $f(R_{i-1} + SK_{3i-2}, SK_{3i-1})$

$L_i = R_{i-1} + SK_{3i-2} + SK_{3i}$

Setiap putaran menggunakan operasi XOR dan +. Operasi + adalah operasi modulo 2^{64} terhadap 64 bit data masukan bersamaan dengan fungsi f yang merupakan fungsi nonlinier yang cukup rumit. Hal ini digunakan untuk memaksimalkan avalanche antara bit-bt masukan. Berikut ini adalah gambar proses enkripsi pada LOKI97



Gambar 10 Gambar Proses Enkripsi pada LOKI97

Untuk proses dekripsi, proses putaran hanya dibalik. Dengan demikian, perhitungan untuk proses dekripsi adalah sebagai berikut :

$L_{i-1} = R_i$ XOR $f(L_i - SK_{3i}, SK_{3i-1})$

$R_{i-1} = L_i - SK_{3i} - SK_{3i-2}$

III.1.2 PEMBANGKITAN SUBKUNCI LOKI97

Algoritma LOKI97 menggunakan jadwal pembangkitan subkunci berdasarkan jaringan Feistel yang tidak seimbang yang mengoperasikan 4 buah blok sebesar 64 bit. Dikarenakan ukuran kunci yang digunakan bisa memiliki 3 buah ukuran yang berbeda, maka inisialisasi kunci untuk masing-masing ukuran pun berbeda. Misalkan untuk kunci sepanjang 256 bit, inisialisasinya adalah

$[K4_0|K3_0|K2_0|K1_0] = [Ka|Kb|Kc|Kd]$.

Untuk kunci dengan panjang 192 bit, maka inisialisasinya adalah dengan cara :

$[K4_0|K3_0|K2_0|K1_0] = [Ka|Kb|Kc|f(Ka,Kb)]$

Sedangkan untuk kunci sepanjang 128 bit, inisialisasi pembangkit subkuncinya adalah sebagai berikut :

$[K4_0|K3_0|K2_0|K1_0] = [Ka|Kb|f(Kb,Ka)|f(Ka,Kb)]$

Berikut ini adalah proses untuk mendapatkan subkunci Ski dengan melakukan putaran sebanyak 48 kali.

$SKi = K1_i = K4_{i-1}$ XOR $g_i(K1_{i-1}, K3_{i-1}, K2_{i-1})$

$K4_i = K3_{i-1}$

$K3_i = K2_{i-1}$

$K2_i = K1_{i-1}$

dengan I mulai dari 1 sampai dengan 48 dan :

$gi(K1, K3, K2i) = f(K1+K3+ (Delta * i), K2)$

$Delta = \lfloor (sqrt(5)-1) * 2^{63} \rfloor$

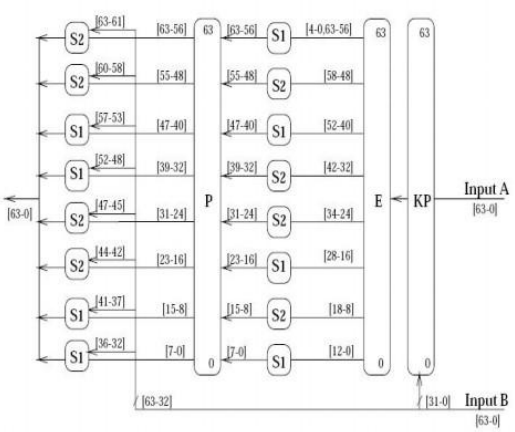
III.1.2 FUNGSI F LOKI97

Fungsi f pada LOKI97 menerima dua buah masukan (L dan R) sebesar 64 bit dan menghasilkan blok yang masing-masing berukuran 64 bit juga. Cara pemrosesannya dengan menggunakan S-Box dua lapis dengan nilai nonlinier maksimal yang memungkinkan untuk menghasilkan sebuah blok berukuran 64 bit.

Selain menggunakan S-Box, digunakan juga proses permutasi sebanyak dua kali untuk memaksimalkan nilai avalanche pada keseluruhan bit. Fungsi f ini didefinisikan sebagai berikut ;

$F(A,B) = Sb(P(Sa(E(KP(A,B)))),B)$

Gambar fungsi f pada LOKI97 adalah sebagai berikut :



Gambar 11 Gambar Fungsi f pada LOKI97

Pada deskripsi fungsi f di atas, terdapat beberapa fungsi antara, yaitu :

- KP(A<B)

Fungsi ini merupakan fungsi permutasi biasa yang memecah A menjadi dua bagian dengan masing-masing bagian memiliki panjang 32 bit dan menggunakan bit paling kanan dari B untuk menentukan apakah menukar bit pada posisi tersebut.

- E()

E() merupakan fungsi perluasan seperti yang terdapat dalam LOKI91 namun pergeseran yang ada lebih disederhanakan.

- Sa(),Sb()

Sa() dan Sb() merupakan dua buah kolom dari S-Box yang dibentuk dengan cara menggabungkan S1 dan S2 dimana Sa() = [S1,S2,S1,S2,S2,S1,S2,S1] dan Sb() = [S2,S2,S1,S1,S2,S2,S1,S1].

- P()

Fungsi antara ini merupakan sebuah fungsi permutasi untuk menyebarkan hasil keluaran dari S-Box untuk dibentuk menjadi blok berukuran 64 bit.

III.1.3 S-BOX PADA LOKI97

LOKI97 menggunakan S-Box berbentuk kotak memakai nilai Galois Field yang ganjil $GF(2^n)$. supaya masukan yang didapat berjumlah ganjil maka S1 menggunakan masukan sebanyak 13 bit, sedangkan S2 menggunakan masukan sebanyak 11 bit. Nilai masukan yang dimasukan dibalik terlebih dahulu agar nilai masukan 1 tidak akan menghasilkan 1, begitu pula nilai masukan 0 tidak akan menghasilkan 0.

$S1[x] = ((x \text{ XOR } 1FFF^3) \bmod 2911) \,\&\, FF$,dalam $GF(2^{13})$.

$S2[x] = ((x \text{ XOR } 7FF^3) \bmod AA7) \,\&\, FF$,dalam $GF(2^{11})$.

IV ALGORITMA GOST

GOST merupakan blok cipher dari bekas Uni Sovyet, yang merupakan singkatan dari "Gosudarstvennyi Standard" atau Standar Pemerintah, standar ini bernomor 28147-89 oleh sebab itu metoda ini sering disebut sebagai

GOST 28147-89. GOST secara struktural mirip dengan DES.

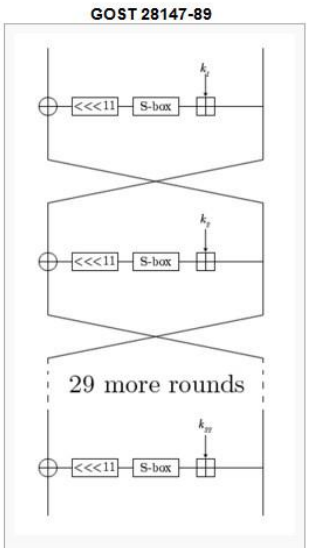
IV.1 SPESIFIKASI ALGORITMA GOST

GOST merupakan blok cipher 64 bit dengan panjang kunci 256 bit. Algoritma ini mengiterasi algoritma enkripsi sederhana sebanyak 32 putaran (round). Untuk mengenkripsi pertama-tama plainteks 64 bit dipecah menjadi 32 bit bagian kiri, L dan 32 bit bagian kanan, R. Subkunci (subkey) untuk putaran i adalah Ki. Pada satu putaran ke-i operasinya adalah sebagai berikut :

$L_i = R_{i-1}$

$R_i = L_{i-1}$ xor $f(R_{i-1},K_i)$

Berikut ini adalah diagram algoritma GOST



Gambar 12 Gambar Algoritma GOST

IV.1.1 PEMBANGKITAN SUBKUNCI GOST

Subkunci dihasilkan secara sederhana yaitu dari 256 bit kunci yang dibagi menjadi delapan 32 bit blok : k1, k2, …, k8. Setiap putaran menggunakan subkunci yang berbeda. Dekripsi sama dengan enkripsi dengan urutan ki dibalik

IV.1.2 FUNGSI F PADA GOST

Pada fungsi f mula-mula bagian kanan data ditambah dengan subkunci ke-i modulus 232. Hasilnya dipecah menjadi delapan bagian 4 bit dan setiap bagian menjadi input s-box yang berbeda. Di dalam GOST terdapat 8 buah s-box, 4 bit pertama menjadi s-box pertama, 4 bit kedua menjadi s-box kedua, dan seterusnya. Output dari 8 s-box kemudian dikombinasikan menjadi bilangan 32 bit kemudian bilangan ini dirotasi 11 bit kekiri. Akhirnya hasil operasi ini di-xor dengan data bagian kiri yang kemudian menjadi bagian kanan dan bagian kanan menjadi bagian kiri (swap). Pada implementasinya nanti rotasi pada fungsi f dilakukan pada awal saat inisialisasi sekaligus membentuk s-box 32 bit dan dilakukan satu kali saja sehingga lebih menghemat operasi dan dengan demikian mempercepat proses enkripsi/dekripsi.

IV.1.3 S-BOX PADA GOST

Standar GOST tidak menentukan bagaimana menghasilkan s-box sehingga ada spekulasi bahwa sebagian organisasi di (eks) Sovyet mempunyai s-box yang baik dan sebagian diberi s-box yang buruk sehingga mudah diawasi/dimata-matai.

Berikut ini adalah gambar S-Box yang digunakan oleh Federasi Bank Sentral Rusia (Central Bank of Russian Federation) :

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

Gambar 13 Gambar salah satu contoh S-Box untuk Algoritma GOST

V ALGORITMA BLOWFISH

Blowfish merupakan algoritma kriptografi yang beroperasi pada mode blok. Algoritma ini ditujukan untuk diimplementasikan pada sebuah mikroprosesor berskala besar. Algoritma Blowfish sendiri tidak dipatenkan sehingga dapat digunakan oleh banyak orang. Blowfish dirancang untuk memenuhi kriteria sebagai berikut :

- Cepat. Blowfish dirancang agar dapat mengenkripsi data pada mikroprosesor 32 bit dengan kecepatan 26 clock cycles per byte.
- Kompak. Blowfish dirancang agar dapat berjalan dengan penggunaan memori kurang dari 5kB.
- Sederhana. Blowfish dirancang hanya menggunakan operasi-operasi sederhana. Operasi yang digunakan dalam blowfish adalah penambahan, XOR, dan table lookup dalam operand 32 bit. Rancangan yang sederhana ini mempermudah proses analisa yang membuat B;lowfish terhidar dair kesalahan implementasi.
- Keamanan yang beragam. Panjang kunci yang digunakan dalam algoritma blowfish bervariasi dengan panjang kunci maksimal adalah 448 bit.

V.1 SPESIFIKASI ALGORITMA BLOWFISH

Blowfish merupakan blok cipher yang beroperasi pada blok berukuran 64 bit dengan panjang kunci yang variatif. Algoritma blowfish terdiri dari dua bagian, yaitu ekspansi kunci dan enkripsi data. Ekspansi kunci mengubah sebuah kunci dengan panjang maksimal 448 bit kepada beberapa array upa-kunci dengan ukuran total 4168 byte.

Enkripsi data terdiri dari sebuah fungsi sederhana yang mengalami putaran atau iterasi sebanyak 16 kali. Setiap putaran terdiri dari sebuah permutasi yang bergantung

pada kunci dan substitusi yang bergantung pada kunci-dan-data. Seluruh operasi berupa penambahan dan XOR dengan kata sepanjang 32 bit. Operasi tambahan yang digunakan hanya berupa data lookup terhadap array dengan empat indkes yang dilakukan setiap putaran.

Blowfish adalah algoritma yang menerapkan jaringan Feistel (Feistel Network) yang terdiri dari 16 putaran. Input adalah elemen 64-bit,X.Untuk alur algoritma enkripsi dengan metoda Blowfish dijelaskan sebagai berikut :

a) Bentuk inisial P-array sebanyak 18 buah (P1,P2,.....P18) masing-msing bernilai 32-bit. Array P terdiri dari delapan belas kunci 32-bit subkunci :
    a. P1,P2,.....,P18

b) Bentuk S-box sebanyak 4 buah masing-masing bernilai 32-bit yang memiliki masukan 256.
    a. Empat 32-bit S-box masing-masing mempunyai 256 entri :
    b. S1,0,S1,1,.....,S1,255
    c. S2,0,S2,1,.....,S2,255
    d. S3,0,S3,1,.....,S3,255
    e. S4,0,S4,1,.....,S4,255

c) Plaintext yang akan dienkripsi diasumsikan sebagai masukan, Plaintext tersebut diambil sebanyak 64-bit, dan apabila kurang dari 64-bit maka kita tambahkan bitnya, supaya dalam operasi nanti sesuai dengan datanya.

d) Hasil pengambilan tadi dibagi 2, 32-bit pertama disebut XL, 32-bit yang kedua disebut XR.

e) Selanjutnya lakukan operasi XL = XL xor Pi dan XR = F(XL) xor XR

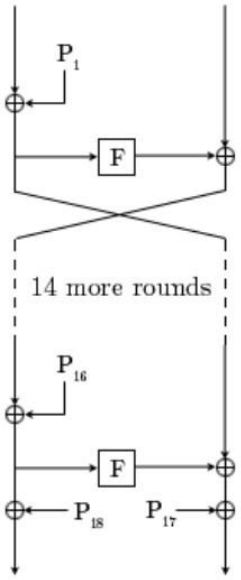
f) Hasil dari operrasi diatas ditukar XL menjadi XR dan XR menjadi XL.

g) Lakukan sebanyak 16 kali, perulangan yang ke-16 lakukan lagi proses penukaran XL dan XR.

h) Pada proses ke-17 lakukan operasi untuk XR = XR xor P17 dan XL = XL xor P18.

i) Proses terakhir satukan kembali XL dan XR sehingga menjadi 64-bit kembali.

Untuk lebih jelas lagi dapat dilihat digambar dibawah ini :



Gambar 14 Gambar Algoritma Blowfish

V.1.1 PEMBANGKITAN SUBKUNCI BLOWFISH

Subkunci dihitung menggunakan algoritma Blowfish, caranya adalah sebagai berikut:

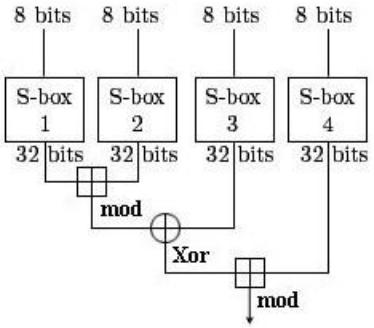
a) Pertama-tama inisialisasi P-array dan kemudian empat S-box secara berurutan dengan string yang tetap. String ini terdiri digit hexadesimal dari p1.
b) Langkah ke dua XOR P1 dengan 32 bit kunci pertama, XOR P2 dengan 32 bit kedua dari kunci dan seterusnya untuk setiap bit dari kunci (sampai P18). Ulangi terhadap bit kunci sampai seluruh P-array di XOR dengan bit kunci.flowchart
c) Enkrip semua string nol dengan algoritma Blowfish dengan menggunakan subkunci seperti dijelaskan pada langkah (1) dan (2).
d) Ganti P1 dan P2 dengan keluaran dari langkah (3)
e) Enkrip keluaran dari langkah (3) dengan algoritma Blowfish dengan subkunci yang sudah dimodifikasi.
f) Ganti P3 dan P4 dengan keluaran dari langkah (5).
g) Lanjutkan proses tersebut, ganti seluruh elemen dari P-array, dan kemudian seluruh keempat S-box berurutan, dengan keluaran yang berubah secara kontinyu dari algoritma Blowfish.

V.1.2 FUNGSI F PADA BLOWFISH

Fungsi f pada algoritma blowfishadalah sebagai berikut :

Bagi XL, menjadi empat bagian 8-bit : a,b,c dan d.

F(XL)=((S1,a + S2,b mod 232)xor S3,c) + S4,c mod 232



Gambar 15 Gambar fungsi f pada algoritma Blowfish

V.1.3 S-BOX PADA BLOWFISH

Cara pembangkitan S-box dilakukan dengan operasi penambahan dan XOR. S-box bergantung pada kunci.

VI KESIMPULAN

Salah satu jenis algoritma kriptografi modern yang sering digunakan adalah block cipher. Algoritma kriptografi ini beroperasi pada plainteks/ cipherteks dalam bentuk blok bit, yaitu enkripsi/dekripsi dilakukan terhadap satu blok (terdiri dari beberapa bit) dalam satu waktu. Misalnya panjang blok adalah 64 bit, maka itu berarti algoritma enkripsi memperlakukan 64 bit / 8 karakter setiap kali enkripsi / dekripsi (1 karakter = 8 bit dalam pengkodean ASCII).

Pada algoritma kriptografi dengan block cipher, ada 4 mode yang dapat digunakan. Keempat mode itu adalah Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), dan Output Feedback (OFB). Setiap mode memiliki beberapa kelemahan dan kekurangan masing-masing. Sebagai contoh, ECB cocok digunakan untuk basis data. Sedangkan mode OFB cocok digunakan untuk video atau suara, karena kesalahan satu bit pada dekripsi tidak akan terlalu mempengaruhi.

Ada banyak contoh algoritma yang menggunakan block cipher. Contohnya : Twofish, Blowfish, DES, Gost, Idea, RC5, Safer, Square, RC6, Loki97, dll. Makalah ini akan membahas lebih lanjut mengenai algoritma Blowfish, GOST, dan LOKI.

Algoritma LOKI merupakan algoritma blok cipher yang menggunakan kunci privat yang mengoperasikan data sebesar 128 bit dengan kunci 256 bit. LOKI97 merupakan pengembangan dari LOKI89 dan LOKI91. Algoritma ini melakukan evolusi dengan memperkuat penjadwalan kunci dan memperbesar ukuran kunci yang digunakan.

GOST merupakan blok cipher dari bekas Uni Sovyet, yang merupakan singkatan dari "Gosudarstvennyi Standard" atau Standar Pemerintah, standar ini bernomor 28147-89 oleh sebab itu metoda ini sering disebut sebagai GOST 28147-89. GOST secara struktural mirip dengan DES.

GOST merupakan blok cipher 64 bit dengan panjang kunci 256 bit. Algoritma ini mengiterasi algoritma enkripsi sederhana sebanyak 32 putaran (round).

Sedangkan Algoritma Blowfish merupakan blok cipher yang beroperasi pada blok berukuran 64 bit dengan panjang kunci yang variatif. Algoritma blowfish terdiri dari dua bagian, yaitu ekspansi kunci dan enkripsi data. Ekspansi kunci mengubah sebuah kunci dengan panjang maksimal 448 bit kepada beberapa array upa-kunci dengan ukuran total 4168 byte.Enkripsi data terdiri dari sebuah fungsi sederhana yang mengalami putaran atau iterasi sebanyak 16 kali. Setiap putaran terdiri dari sebuah permutasi yang bergantung pada kunci dan substitusi yang bergantung pada kunci-dan-data.

Berikut ini adalah perbedaan ketiga algoritma dilihat dari pembangkitan subkunci, fungsi f, dan S-Box yang digunakan oleh masing-masing algoritma.

Tabel 1. Tabel Perbandingan Algoritma LOKI, GOST, dan Blowfish

Perbedaan	LOKI97	GOST	Blowfish
Pembangkitan subkunci	Algoritma LOKI97 menggunakan jadwal pembangkitan subkunci berdasarkan jaringan Feistel yang tidak seimbang yang mengoperasikan 4 buah blok sebesar 64 bit.	Subkunci dihasilkan secara sederhana yaitu dari 256 bit kunci yang dibagi menjadi delapan 32 bit blok : k1, k2, …, k8. Setiap putaran menggunakan subkunci yang berbeda. Dekripsi sama dengan enkripsi dengan urutan ki dibalik	Pembangkitan subkunci pada blowfish memiliki 8 langkah.
Fungsi f	Fungsi f pada LOKI97 menerima dua buah masukan (L dan R) sebesar 64 bit dan menghasilkan blok yang masing-masing berukuran 64 bit juga. Cara pemrosesann	Pada fungsi f mula-mula bagian kanan data ditambah dengan subkunci ke-i modulus 232. Hasilnya dipecah menjadi delapan	Fungsi f pada algoritma blowfishadalah sebagai berikut : Bagi XL, menjadi empat bagian 8-bit : a,b,c dan d. F(XL)=((S1,a + S2,b mod 232)xor

	ya dengan menggunakan S-Box dua lapis dengan nilai nonlinier maksimal yang memungkinkan untuk menghasilkan sebuah blok berukuran 64 bit.	bagian 4 bit dan setiap bagian menjadi input s-box yang berbeda. Di dalam GOST terdapat 8 buah s-box, 4 bit pertama menjadi s-box pertama, 4 bit kedua menjadi s-box kedua, dan seterusnya	S3,c) + S4,c mod 232
S-Box	LOKI97 menggunakan S-Box berbentuk kotak memakai nilai Galois Field yang ganjil GF(2^n). supaya masukan yang didapat berjumlah ganjil maka S1 menggunakan masukan sebanyak 13 bit, sedangkan S2 menggunakan masukan sebanyak 11 bit. Nilai masukan yang dimasukan dibalik terlebih dahulu agar nilai masukan 1 tidak akan menghasilkan 1, begitu pula nilai masukan 0 tidak akan menghasilkan 0.	Standar GOST tidak menentukan bagaimana menghasilkan s-box.	Cara pembangkitan S-box dilakukan dengan operasi penambahan dan XOR. S-box bergantung pada kunci

VII. UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Tuhan Yang Maha Esa karena atas rahmat-Nya, penulis dapat menyelesaikan makalah ini. Terima kasih juga penulis ucapkan kepada Bapak Rinaldi Munir, selaku dosen Kriptografi semester genap tahun ajaran 2010/2011 atas bimbingan yang diberikan selama ini. Makalah ini juga dapat terselesaikan berkat kuliah yang disampaikan oleh Bapak Rinaldi Munir.

REFERENCES

[1] Munir, Rinaldi. Algoritma Kriptografi modern
[2] http://en.wikipedia.org/wiki/LOKI97
Tanggal Akses: 10 Maret 2011, 18:00 WIB
[3] http://www.unsw.adfa.edu.au/~lpb/research/loki97/
Tanggal Akses: 15 Maret 2011, 18:30 WIB.
[4] http://en.wikipedia.org/wiki/GOST_(block_cipher)
Tanggal Akses: 12 Maret 2011, 17:00 WIB
[5] http://id.wikipedia.org/wiki/GOST
Tanggal Akses: 17 Maret 2011, 18:00 WIB
[6] http://www.klik-kanan.com/forums/index.php?topic=1117.0
Tanggal Akses: 15 Maret 2011, 19:30 WIB.
[7] http://purbasari.wordpress.com/2007/12/28/algoritma-blowfish/
Tanggal Akses: 19 Maret 2011, 17:00 WIB
[8] http://www.bimacipta.com/blowfish.htm
Tanggal Akses: 17 Maret 2011, 21:00 WIB

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 21 Maret 2011

Yongke Yoswara - 13508034