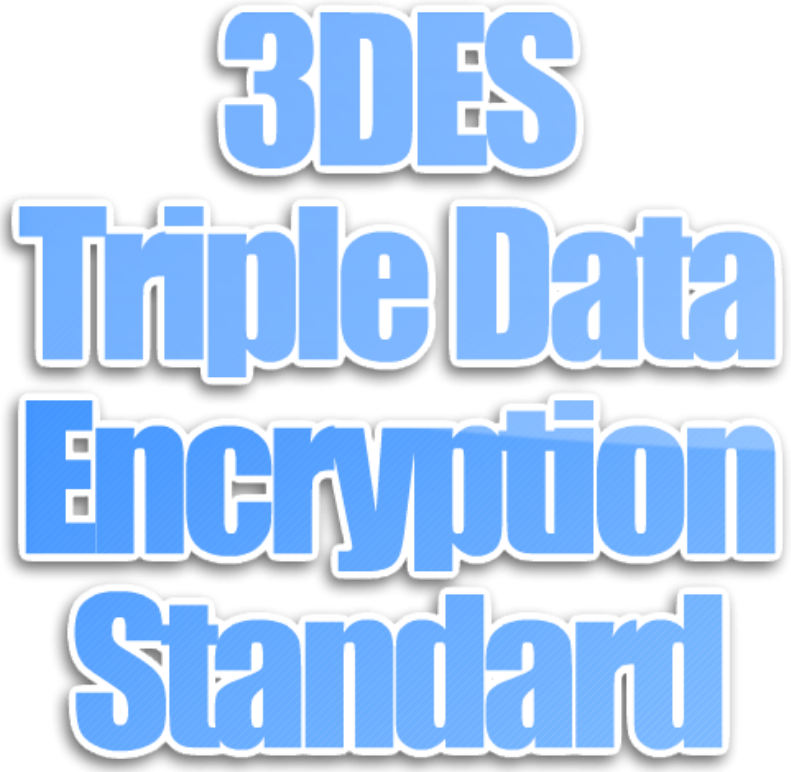
Bahan kuliah II4031 Kriptografi dan Koding

Beberapa Block Cipher Lainnya

Oleh: Rinaldi Munir

Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2024



Triple DES (3DES)

DES Berganda

- Karena DES mempunyai potensi kelemahan pada *brute force atack*, maka dibuat varian dari DES.

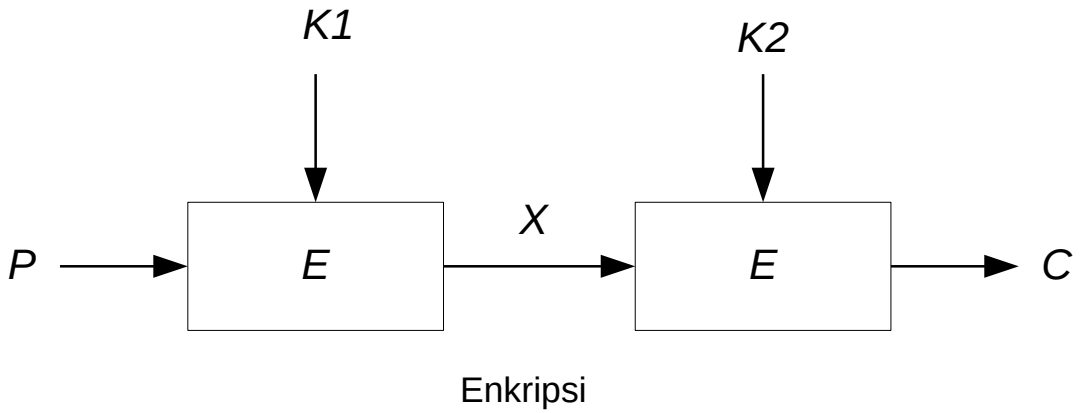
- Varian DES yang paling luas digunakan adalah DES berganda (*multiple DES*).

- DES berganda adalah enkripsi berkali-kali dengan DES dan menggunakan kunci ganda.

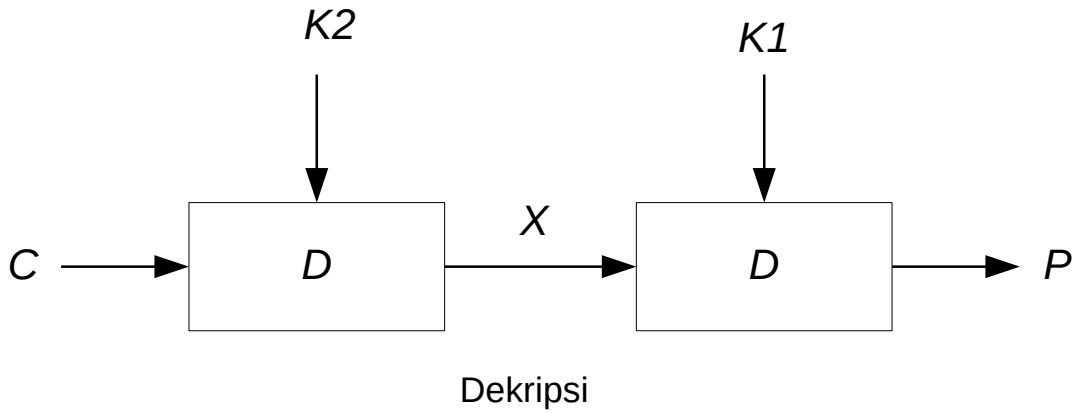
- Tinjau dua macam DES berganda:
  1. *Double DES*
  2. *Triple DES* (3DES)

Double DES

- Menggunakan 2 buah kunci eksternal, K_1 dan K_2.
- Enkripsi: $C = E_{K2}(E_{K1}(P))$
- Dekripsi: $P = D_{K1}(D_{K2}(C))$



Enkripsi



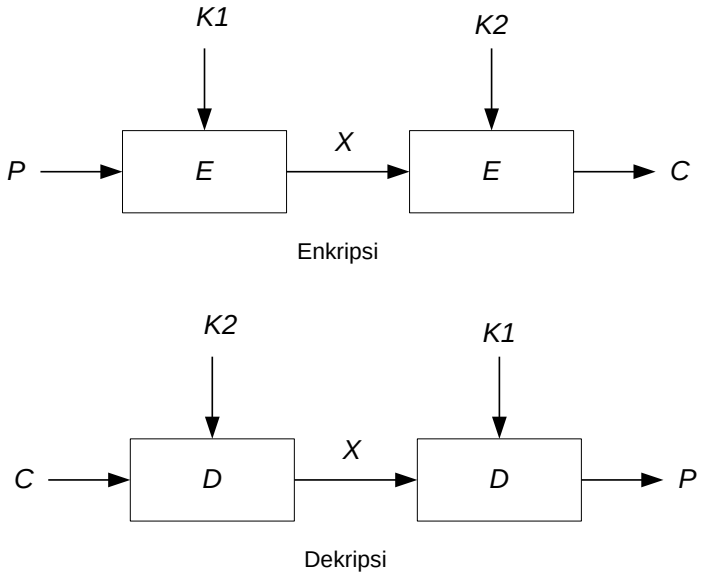
Dekripsi

- Kelemahan *Double DES*: serangan *meet-in-the-middle attack*:
- Dari pengamatan,

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

maka

$$X = E_{K1}(P) = D_{K2}(C)$$



- Misalkan kriptanalis memiliki potongan *C* dan *P* yang berkorepsonden (*known-plaintext attack*).

- Enkripsi *P* untuk semua kemungkinan nilai *K*1 (yaitu sebanyak 2^{56} kemungkinan kunci). Hasilnya adalah semua nilai *X*

- Simpan semua nilai *X* ini di dalam tabel

- Berikutnya, dekripsi *C* dengan semua semua kemungkinan nilai *K*2 (yaitu sebanyak 2^{56} kemungkinan kunci).

- Bandingkan semua hasil dekripsi ini dengan elemen di dalam tabel tadi. Jika ada yang sama, maka dua buah kunci, *K*1 dan *K*2, telah ditemukan.

- Tes kedua kunci ini dengan pasangan plainteks-cipherteks lain yang diketahui. Jika kedua kunci tersebut menghasilkan cipherteks atau plainteks yang benar, maka *K*1 dan *K*2 tersebut merupakan kunci yang benar

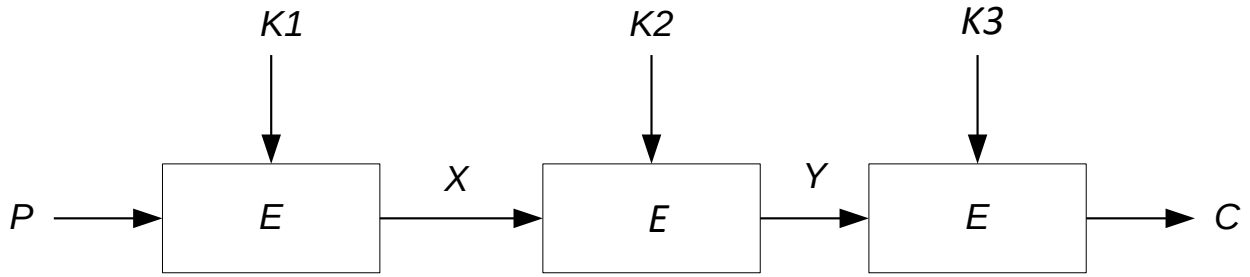
$X = E_{K1}(P)$		$X = D_{K2}(C)$	
K1	X	K2	X
000…00	1010..11	000…00	0110..10
000…01	1101..00	000…01	1000..10
…		…	
…		…	
…		…	
101..10	**010..01**	101..10	010..01
…		…	
…		…	
…		**110..10…**	**010..01**
…		…	
…		…	
111..11	011.. 10	111..11	011.. 10

Triple DES (TDES)

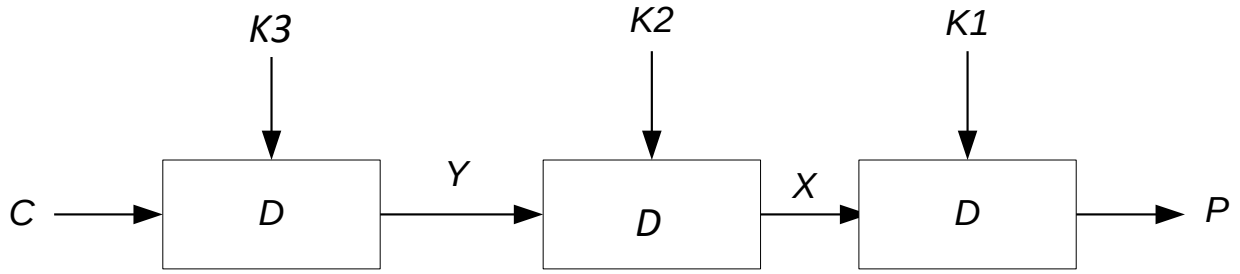
- Menggunakan DES tiga kali
- Bertujuan untuk mencegah *meet-in-the-middle attack*.
- Bentuk umum Triple DES (mode EEE):

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

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



Enkripsi



Dekripsi

- Untuk menyederhanakan TDES, maka langkah di tengah diganti dengan D (mode EDE).

- Ada dua versi TDES dengan mode EDE:

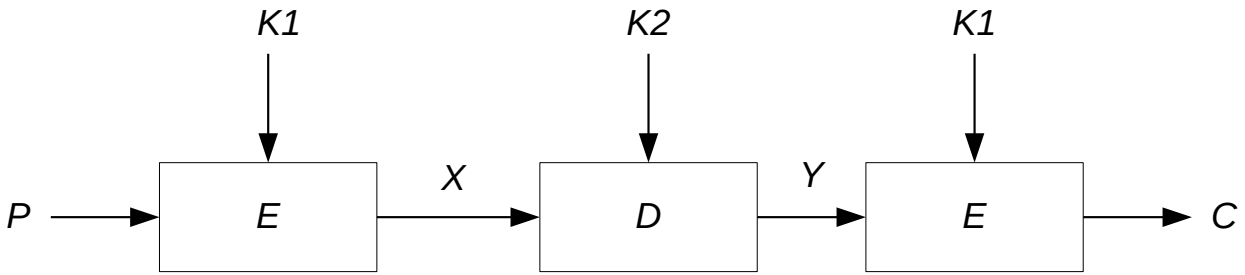
  - Menggunakan 2 kunci

  - Menggunaakn 3 kunci

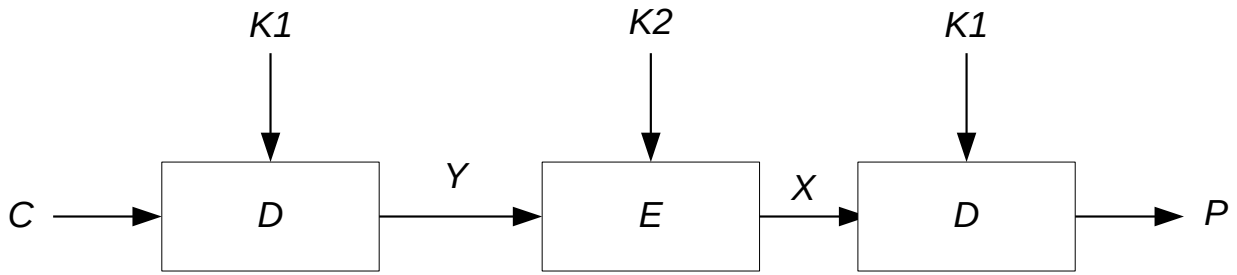
- Kenapa EDE? Supaya kompatibel dengan single DES:

Triple DES

Triple **DES dengan 2 kunci (56 + 56 = 112 bit)**

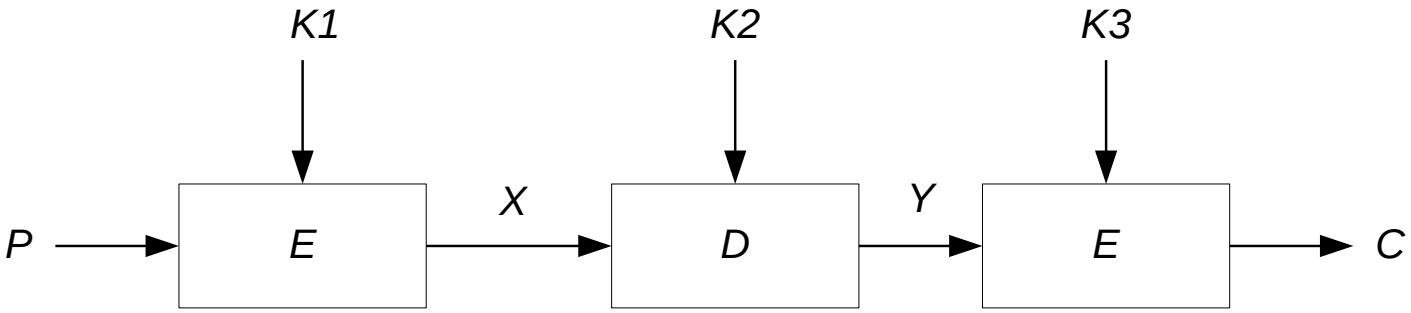


Enkripsi

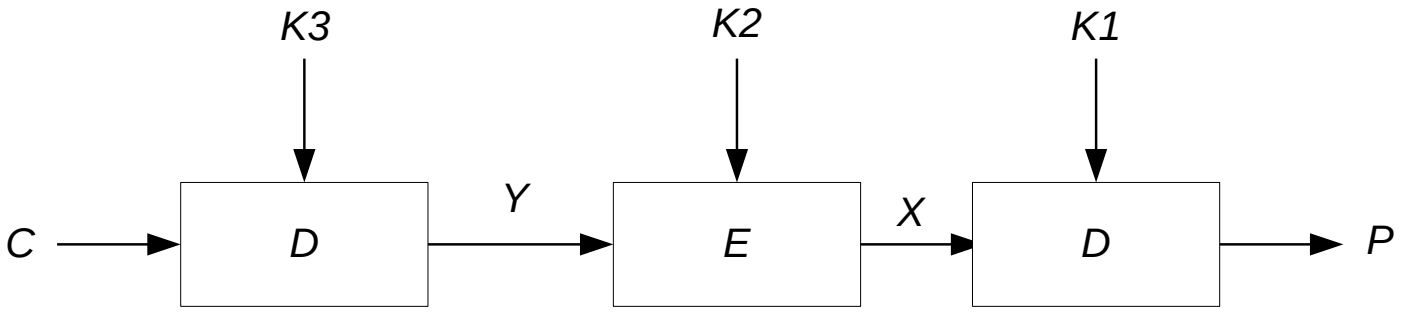


Dekripsi

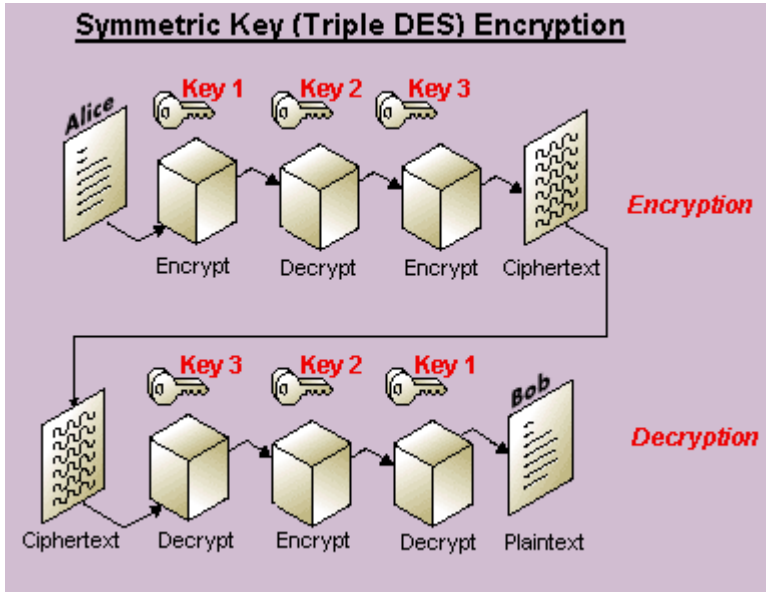
Triple DES dengan 3 kunci (56 + 56 + 56 = 168)



Enkripsi



Dekripsi



GOST

Tinjauan Umum GOST (Magma)

- *GOST* = *Gosudarstvenny Standard*, artinya standard pemerintah, disebutkan di dalam standard GOST 28147-89 (RFC 5830).
- Merupakan algoritma enkripsi *cipher* blok standard dari negara Uni Soviet dulu (USSR).
- Pada mulanya *cipher* blok ini tidak mempunyai nama, tetapi di dalam standard yang baru, GOST R 34.12-2015 (RFC 7801, RFC 8891), *cipher* blok ini dinamakan *Magma*.
- GOST dikembangkan pada tahun 1970 namun baru dirilis kepada publik pada tahun 1990 setelah Uni Soviet bubar.
- Dibuat oleh Soviet sebagai alternatif terhadap algoritma enkripsi standard Amerika Serikat, *DES*.
- *GOST* secara struktural mirip dengan *DES* (menggunakan jaringan Feistel)

- Ukuran blok pesan = 64 bit
- Panjang kunci = 256 bit
- Jumlah putaran = 32 putaran
- Setiap putaran menggunakan kunci putaran.
- Kunci putaran sebenarnya hanya ada 8 buah, K_1 sampai K_8, namun karena ada 32 putaran, maka 8 buah kunci putaran ini dijadwalkan penggunaannya.

Tabel 6.2 Penjadwalan kunci internal GOST

Putaran	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Kunci internal	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8

Putaran	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
Kunci internal	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8	K_8	K_7	K_6	K_5	K_4	K_3	K_2	K_1

- Standard GOST yang baru menspesifikasikan *cipher* blok 128-bit yang diberi nama *Kuznyechik*.

- Pembangkitan kunci putaran sangat sederhana.

- Kunci eksternal yang panjangnya 256 bit dibagi ke dalam delapan bagian yang masing-masing panjangnya 32 bit.

- Delapan bagian ini yang dinamakan K_1, K_2, …, K_8.

- Contoh: K = ‘abcdefghijklmnopqrstuvwxyz123456’ (32 x 8 bit = 256 bit)

maka	K1 = ‘abcd’	K5 = ‘qrst’
	K2 = ‘efgh’	K6 = ‘uvwx’
	K3 = ‘ijkl’	K7 = ‘yz12’
	K4 = ‘mnop’	K8 = ‘3456’

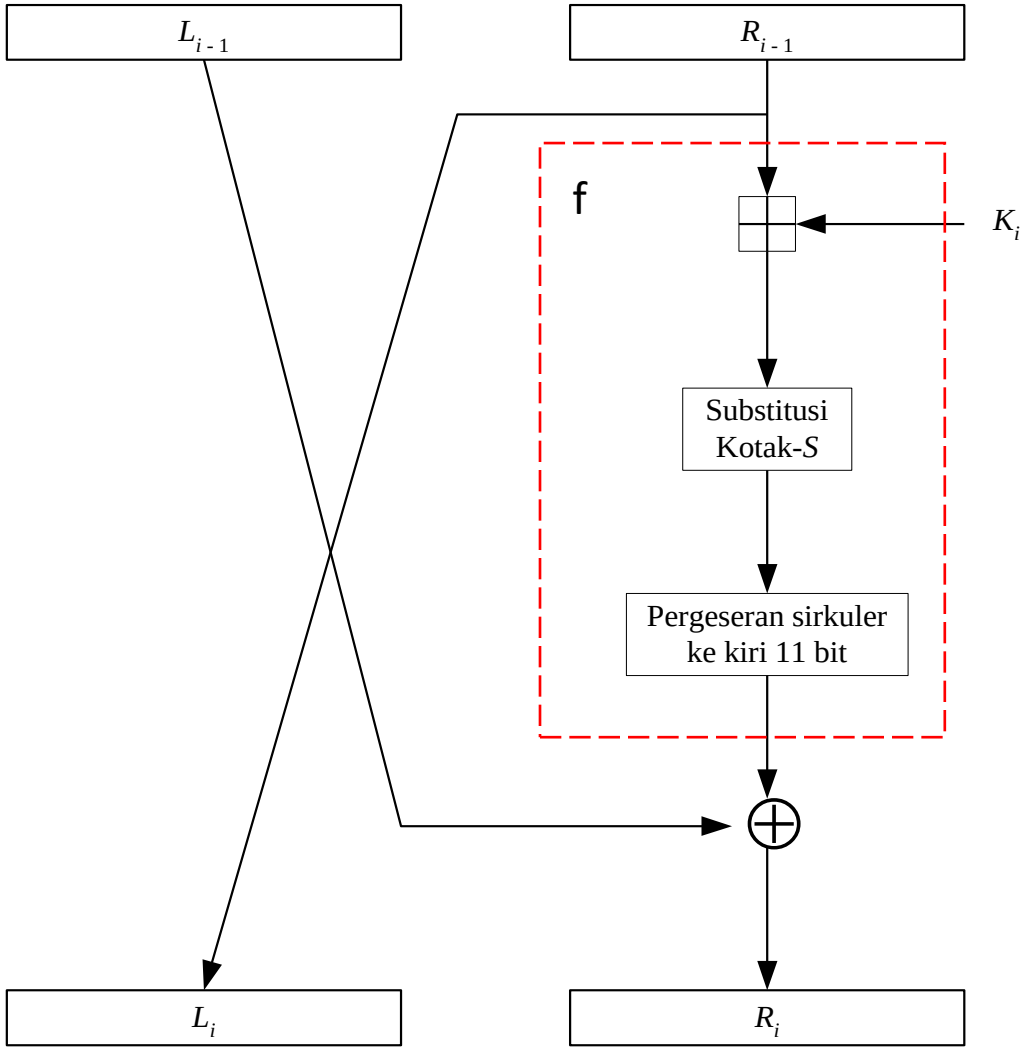
- *GOST* menggunakan Jaringan *Feistel*
- Satu putaran *GOST:*

$L_i = R_{i-1}$

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

- Fungsi *f* terdiri dari

  - penjumlahan dalam modulus 2^{32}
  - substitusi
  - pergeseran

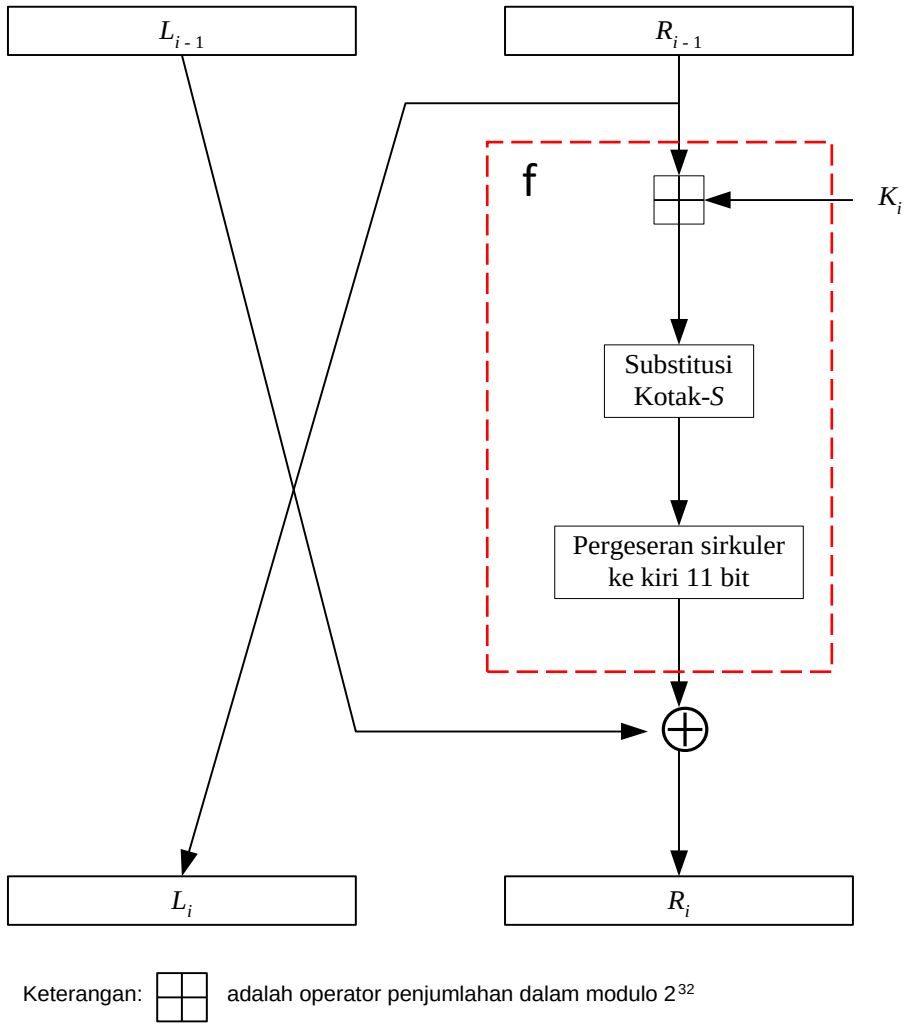


- Hasil penjumlahan R_{i-1} dengan K_i menghasilkan luaran yang panjangnya 32 bit.

- Luaran ini dibagi mejadi 8 bagian yang masing-masing panjangnya 4 bit.

- Setiap 4 bit masuk ke dalam kotak *S* untuk proses substitusi. Empat bit pertama masuk ke dalam kotak *S* pertama, 4 bit kedua masuk ke dalam kotak S kedua, demikian seterusnya.

- Hasil substitusi setiap kotak *S* adalah 4 bit. *GOST* memiliki 8 buah kotak *S*, setiap kotak berisi 16 buah elemen nilai. Setiap kotak berisi permutasi angka 0 sampai 15.



Kotak-*S* 1:															
4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3

Kotak-*S* 2:															
14	11	4	12	6	13	15	10	2	3	8	1	0	7	5	9

Kotak-*S* 3:															
5	8	1	13	10	3	4	2	14	15	12	7	6	0	9	11

Kotak-*S* 4:															
7	13	10	1	0	8	9	15	14	4	6	12	11	2	5	3

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

Kotak-*S* 6:															
4	11	10	0	7	2	1	13	3	6	8	5	9	12	14	14

Kotak-*S* 7:															
13	11	4	1	3	15	5	9	0	10	14	7	6	8	2	12

Kotak-*S* 8:															
1	15	13	0	5	7	10	4	9	2	3	14	6	11	8	12

- Misalnya pada kotak *S* pertama, masukannya:

  `0010` (nilai desimal 2)

  maka luarannya: nilai di dalam elemen ke-2:

  9 atau `1001`

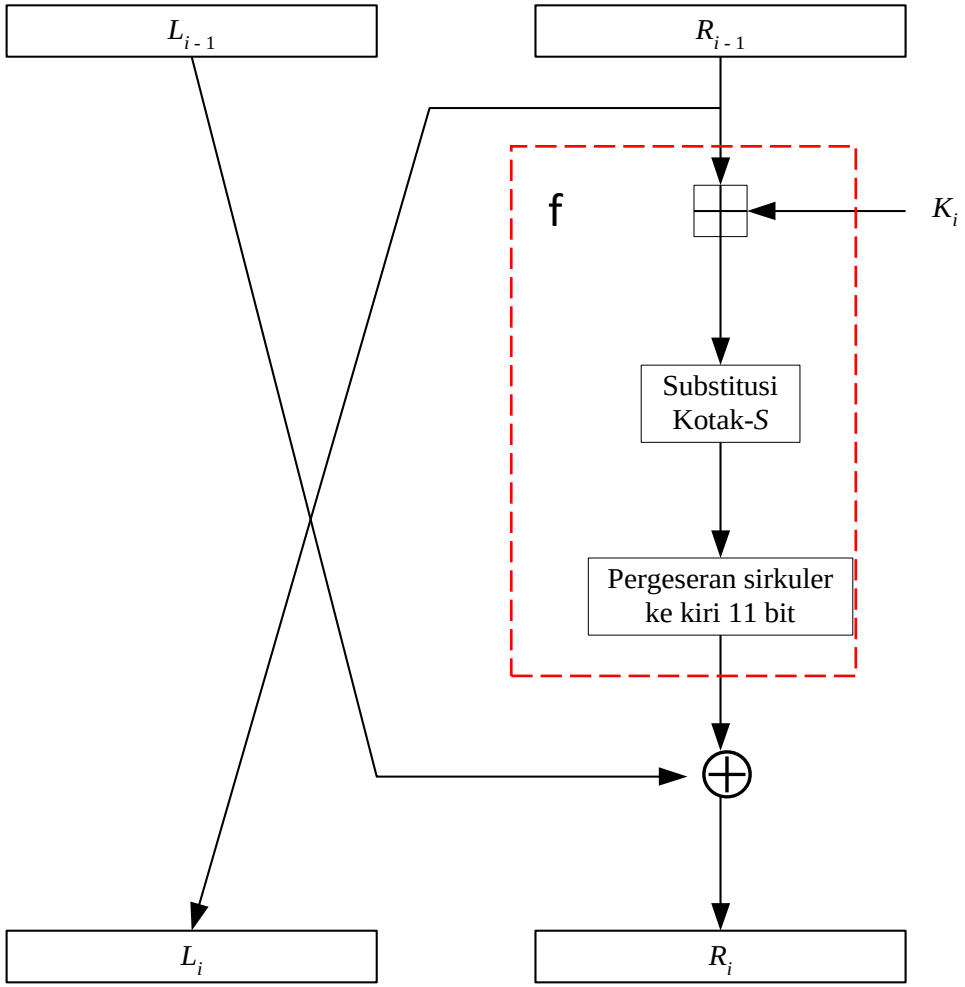
Kotak-*S* 1:															
4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3

- Hasil substitusi dari semua kotak *S* ini digabung menjadi pesan 32-bit.

- Kemudian pesan 32-bit ini digeser ke kiri sejauh 11 bit secara sirkuler.

- Hasilnya kemudian di-*XOR*-kan dengan L_{i-1} untuk kemudian memberikan bagian cipherteks kanan yang baru, R_i. Proses ini diulang sebanyak 32 kali.

- Dekripsi merupakan kebalikan dari enkripsi. Dari "bawah" ke "atas" mengikuti jaringan Feistel.



Keterangan: ⊞ adalah operator penjumlahan dalam modulo 2^{32}

Perbedaan GOST dengan DES:

1. Kunci *DES* 56 bit, sedangkan kunci GOST lebih panjang yaitu 256 bit. Ini menyebabkan *exhaustive key search* terhadap *GOST* lebih sukar dibandingkan dengan *DES*.

2. Jumlah putaran DES 16 kali, sedangkan GOST lebih banyak yaitu 32 kali sehingga membuat kriptanalisis menjadi sangat sulit

3. Kotak *S* di dalam *DES* menerima masukan 6 bit dan luaran 4 bit (berukuran 6×4), sedangkan kotak *S* di dalam GOST menerima masukan 4 bit dan luaran 4 bit (berukuran 4×4)

4. Pembangkitan kunci internal *DES* rumit, sedangkan di dalam *GOST* pembangkitan kunci internalnya sederhana

5. DES mempunyai permutasi yang tidak teratur, sedangkan GOST hanya menggunakan pergeseran 11-bit secara sirkuler

Cryptanalysis of GOST [edit]

The latest cryptanalysis of GOST shows that it is secure in a theoretical sense. In practice, the data and memory complexity of the best published attacks has reached the level of practical, while the time complexity of even the best attack is still 2^{192} when 2^{64} data is available.

Since 2007, several attacks have been developed against reduced-round GOST implementations and/or weak keys.[6][7]

In 2011 several authors discovered more significant flaws in GOST, being able to attack the full 32-round GOST with arbitrary keys for the first time. It has even been called "a deeply flawed cipher" by Nicolas Courtois.[8] Initial attacks were able to reduce time complexity from 2^{256} to 2^{228} at the cost of huge memory requirements,[9] and soon they were improved up to 2^{178} time complexity (at the cost of 2^{70} memory and 2^{64} data).[10][11]

In December 2012, Courtois, Gawinecki, and Song improved attacks on GOST by computing only 2^{101} GOST rounds.[12] Isobe had already published a single key attack on the full GOST cipher,[13] which Dinur, Dunkelman, and Shamir improved upon, reaching 2^{224} time complexity for 2^{32} data and 2^{36} memory, and 2^{192} time complexity for 2^{64} data.[14]

Since the attacks reduce the expected strength from 2^{256} (key length) to around 2^{178}, the cipher can be considered broken. However, for any block cipher with block size of n bits, the maximum amount of plaintext that can be encrypted before rekeying must take place is $2^{n/2}$ blocks, due to the birthday paradox,[15] and none of the aforementioned attacks require less than 2^{32} data.

Sumber: Wikipedia

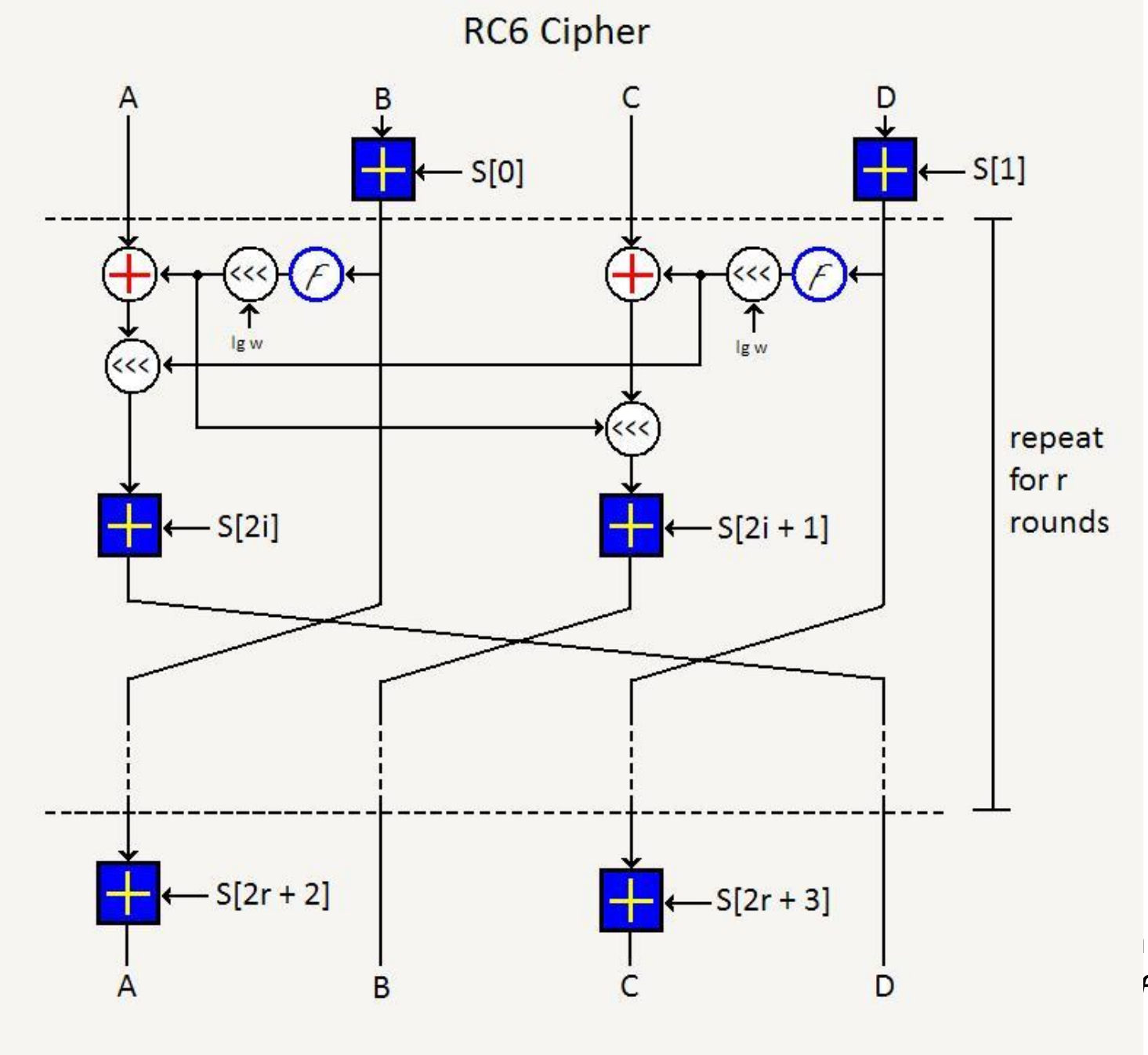
RC6

RC6

- RC6 juga dibuat oleh Ronald Rivest, Matt Robshaw, Ray Sidney, dan Yiqun Lisa Yin dari Laboratorium RSA (Rivest, 1998).

- RC6 merupakan pengembangan RC5. RC6 adalahs alah satu finalis *Advanvced Encryption Standard* atau AES.

- RC6 melakukan enkripsi terhadap blok data berukuran 128 bit, sedangkan ukuran kuncinya bervariasi 128, 192, 256 sampai 2040 bit. Cara pembangkitan kunci internalnya identik dengan RC5.



bar Jaringan Feistel di dalam RC6.
f adalah $f(x) = x(2x + 1)$ (Rivest, 1998)

{ Enkripsi RC6-w/r/b

Masukan: Plainteks disimpan di dalam empat register berukuran

w bit: A, B, C dan D

r adalah jumlah putaran

S[0, … , 2r + 3] adalah kunci internal, panjangnya w bit

Luaran: Cipherteks disimpan di dalam A, B, C, D

}

Algoritma:

$B \leftarrow B + S[0]$

$D \leftarrow D + S[1]$

for $i \leftarrow 1$ *to* r *do*

$t \leftarrow (B*(2B + 1)) <<< \lg w$

$u \leftarrow (D*(2D + 1)) <<< \lg w$

$A \leftarrow ((A \oplus t) <<< u) + S[2i]$

$C \leftarrow ((C \oplus u) <<< t) + S[2i + 1]$

$(A, B, C, D) = (B, C, D, A)$

end

$A \leftarrow A + S[2r + 2]$

$C \leftarrow C + S[2r + 3]$

{ Dekripsi RC6-w/r/b

Masukan: Cipherteks disimpan di dalam empat register berukuran

w bit: A, B, C dan D r adalah jumlah putaran

S[0, … , 2r + 3] adalah kunci internal, panjangnya w bit

Luaran: Plainteks disimpan di dalam A, B, C, D

}

Algoritma:

$C \leftarrow C - S[2r + 3]$

$A \leftarrow A - S[2r + 2]$

for $i \leftarrow r$ *downto 1 do*

$(A, B, C, D) \leftarrow (D, A, B, C)$

$u \leftarrow (D*(2D + 1)) <<< lg\ w$

$t \leftarrow (B*(2B + 1)) <<< lg\ w$

$C \leftarrow ((C - S[2i + 1]) >>> t) \oplus u$

$A \leftarrow ((A - S[2i]) >>> u) \oplus t$

end

$D \leftarrow D - S[1]$

$B \leftarrow B - S[0]$

Keterangan:

$a + b$ adalah penjumlahan dalam modulo 2^w

$a - b$ adalah pengurangan dalam modulo 2^w

$a \oplus b$ adalah operasi *bitwise* XOR dari word yang panjangnya w bit

$a \times b$ adalah perkalian dalam modulo 2^w

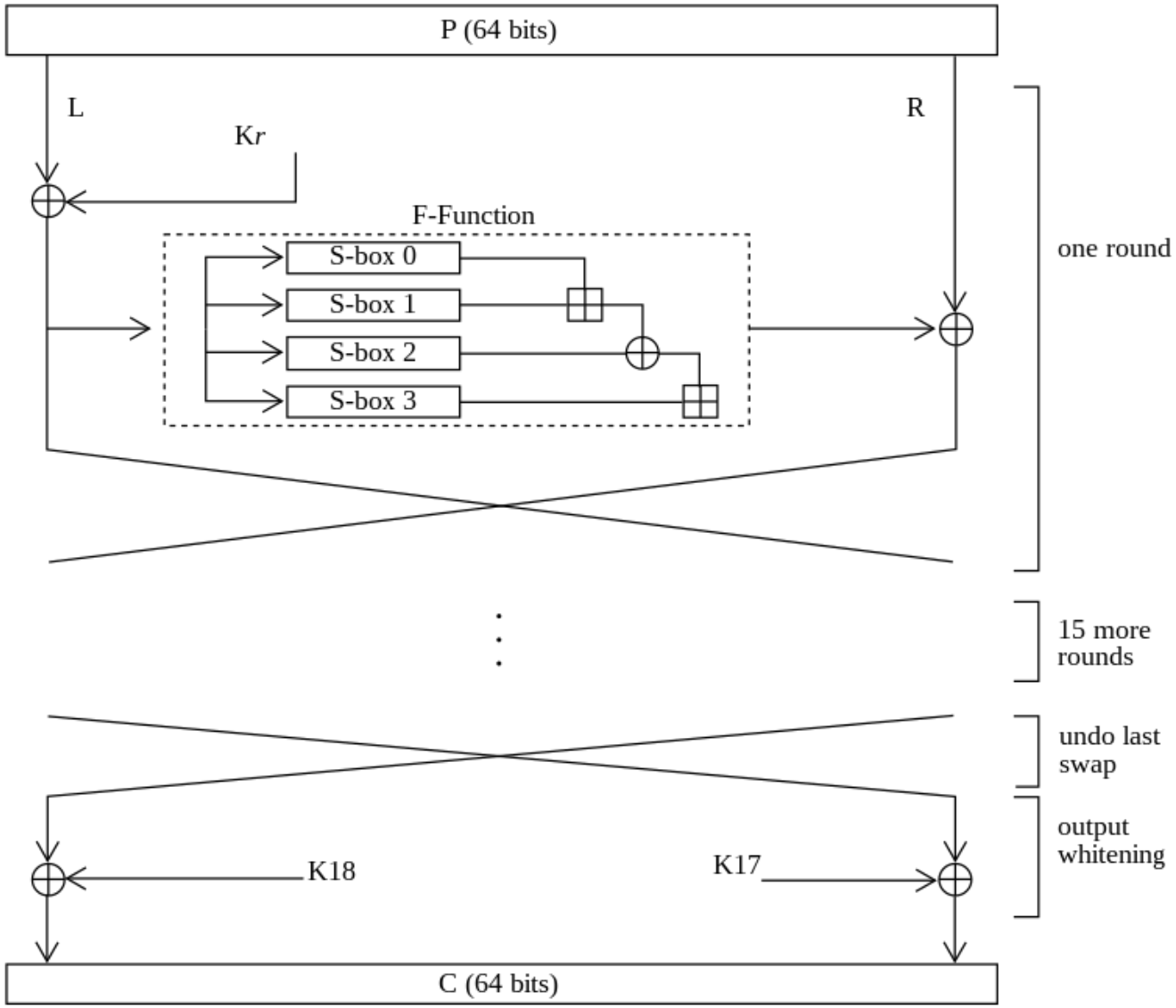
$a <<< b$ adalah operasi pergeseran a (yang panjangnya w bit) ke kiri secara sirkular sejauh log w *least significant* bit dari b

$a >>>$ b adalah operasi pergeseran a (yang panjangnya w bit) ke kanan secara sirkular sejauh log w *least significant* bit dari b

Blowfish

Blowfish

- Dirancang oleh Bruce Schneier tahun 1993.
- Ukuran blok plainteks dan cipherteks di dalam Blowfish adalah 64 bit
- Panjang kunci 32 sampai 448 bit
- Jumlah putaran 16 kali, setiap putaran melakukan operasi substitusi-permutasi
- Jumlah kotak S-box adalah 4 buah



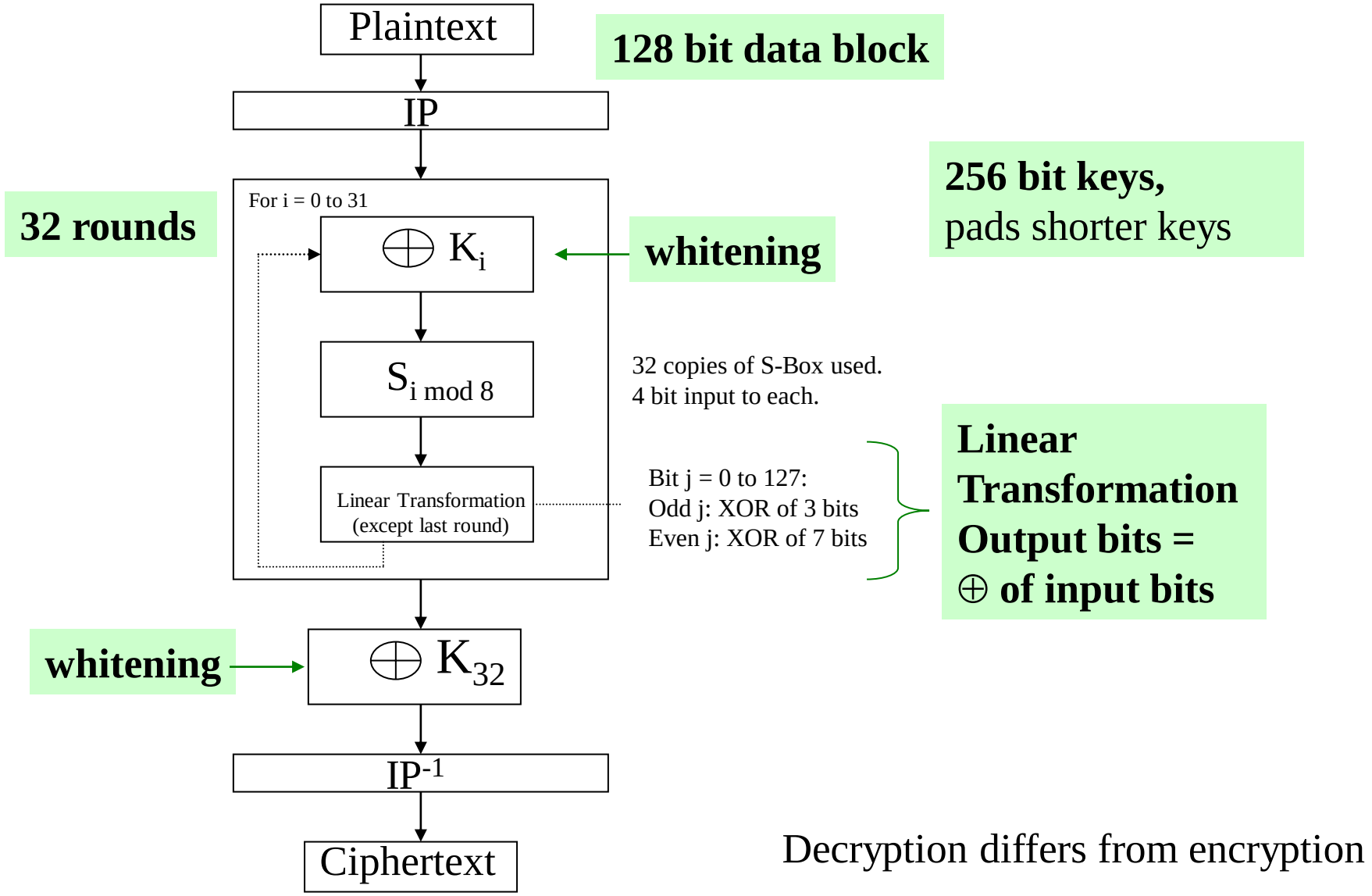
P=Plaintext; C=Ciphertext; K*x* = P-array-entry *x*

⊕ = xor ⊞ = addition mod 2^32

Serpent

Serpent

- Salah satu finalis AES (Advanced Encryption Standard)
- Dirancang oleh Ross Anderson, Eli Biham, dan Lars Knudsen.
- Ukuran blok plainteks dan cipherteks di dalam Serpent adalah 128 bit
- Panjang kunci 128, 192, dan 256 bit
- Jumlah putaran 32 kali, setiap putaran melakukan operasi substitusi-permutasi
- Jumlah kotak S-box adalah 8 buah



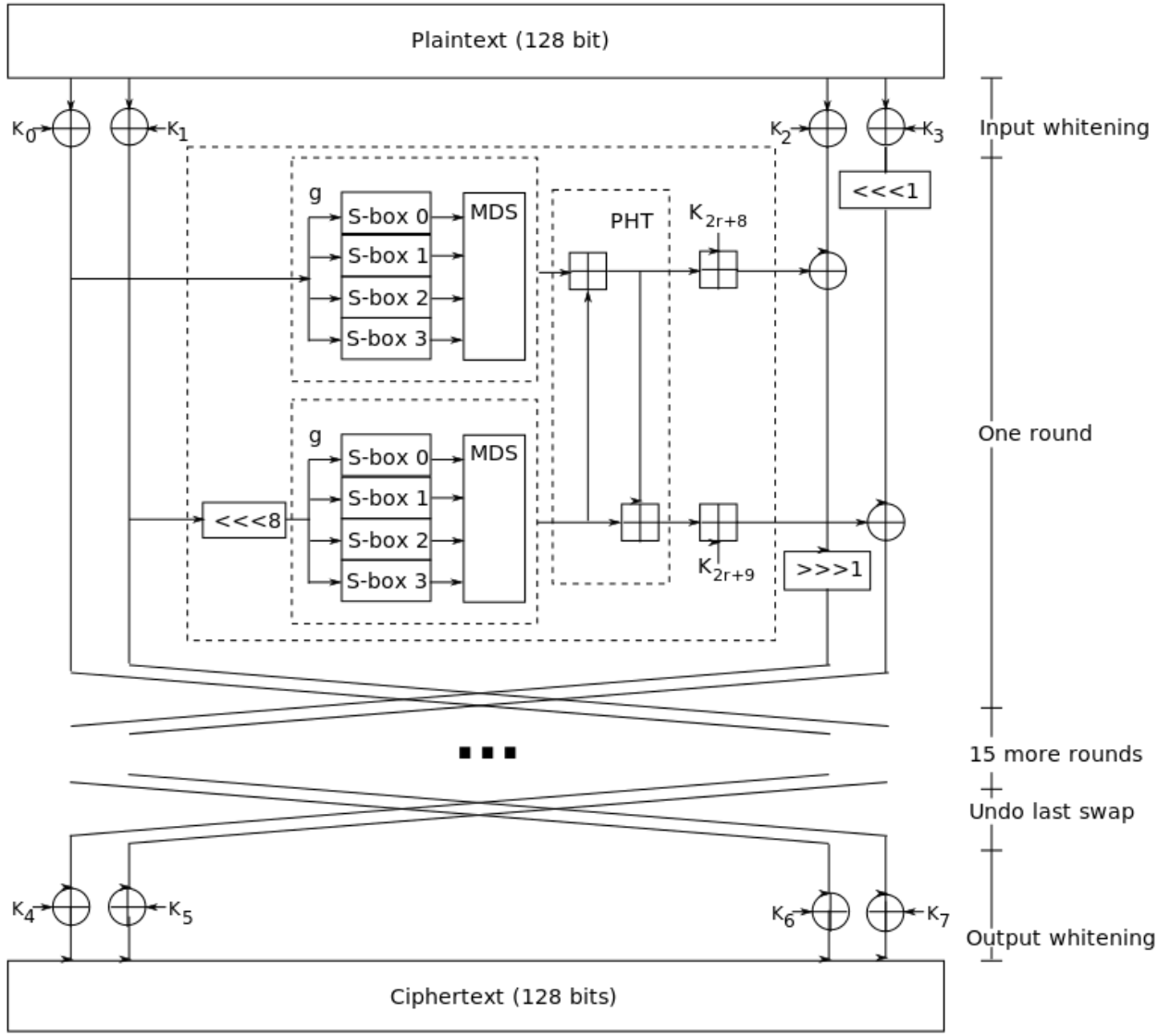
Decryption differs from encryption

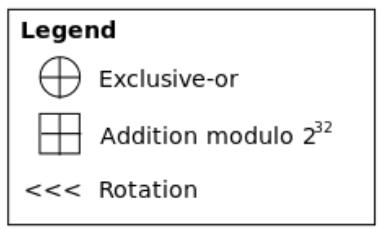
Sumber: Introduction to Practical Cryptography, Lecture 3 Block Ciphers

Twofish

Twofish

- Salah satu finalis AES (*Advanced Encryption Standard*)
- Dirancang oleh Bruce Schneier.
- Twofish dapat dianggap sebagai kelanjutan dari Blowfish
- Ukuran blok plainteks dan cipherteks di dalam Twofish adalah 128 bit
- Panjang kunci sampai 256 bit
- Jumlah putaran 16 kali, setiap putaran melakukan operasi substitusi-permutasi
- Jumlah kotak S-box adalah 5 buah


Plaintext (128 bit)
K0
K1
K2
K3
Input whitening
<<<1
g
S-box 0
S-box 1
S-box 2
S-box 3
MDS
PHT
K2r+8
One round
<<<8
K2r+9
>>>1
15 more rounds
Undo last swap
K4
K5
K6
K7
Output whitening
Ciphertext (128 bits)


Legend
Exclusive-or
Addition modulo 2^32
<<< Rotation

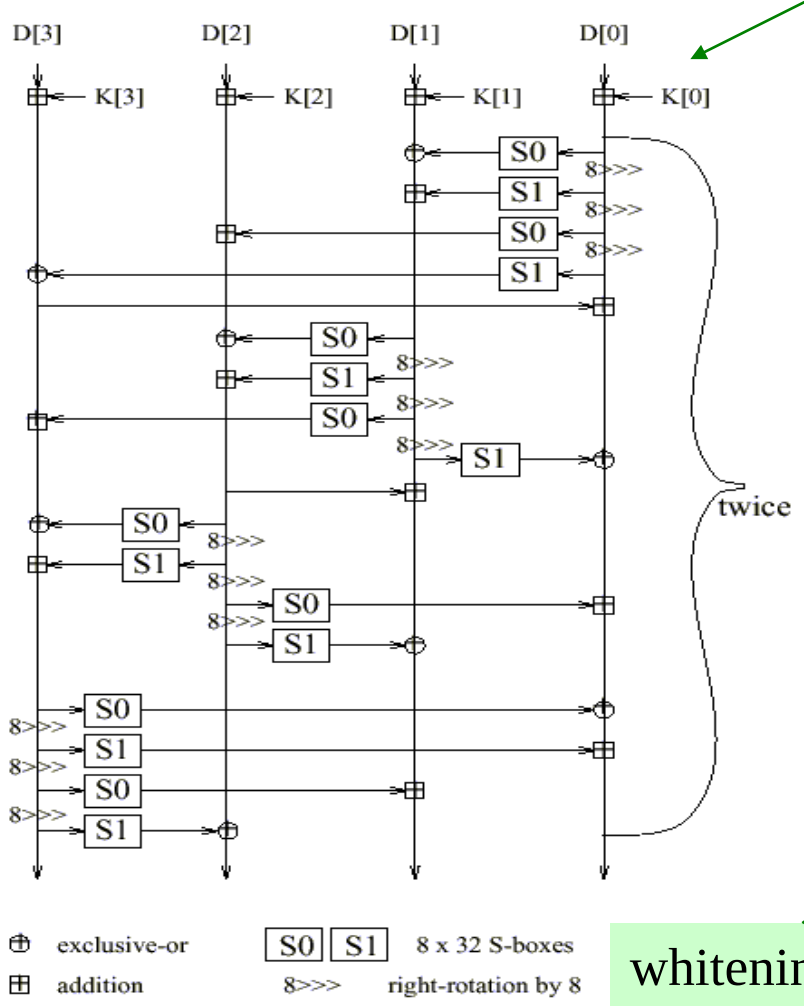
Mars

MARS

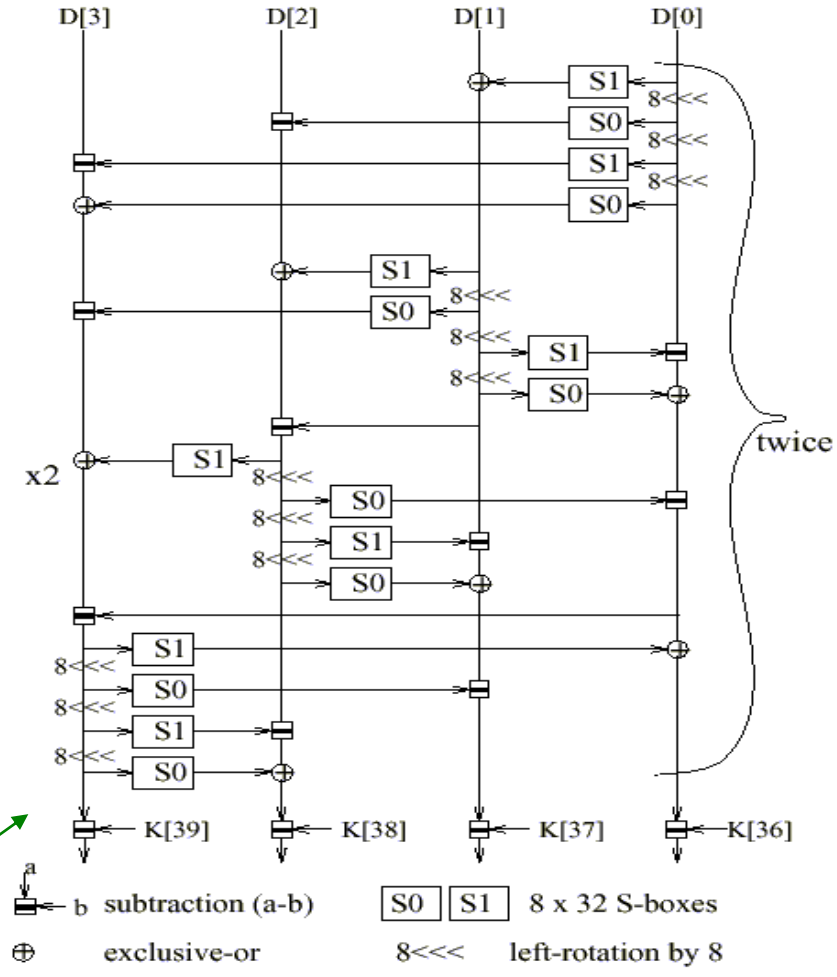
- Salah satu finalis AES (*Advanced Encryption Standard*)
- Dirancang oleh IBM.
- Ukuran blok plainteks dan cipherteks di dalam MARS adalah 128 bit
- Panjang kunci dari 128 sampai 448 bit (pertambahan 32 bit)
- Jumlah putaran 32 kali, setiap putaran melakukan operasi substitusi-permutasi

Forward Mixing

whitening



Backward Mixing



whitening