Algoritma Block Cipher HIFAT

Hilmi Naufal Yafie[1], **M Algah Fattah Illahi**[2].

[1,2] Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika (STEI), Institut Teknologi Bandung (ITB), Jalan Ganesha 10, Bandung 40132
E-mail: 13517035@std.stei.itb.ac.id, 13517122@std.stei.itb.ac.id

Abstract. Di dalam makalah ini, kami mengajukan sebuah block cipher baru HIFAT dengan ukuran blok 64-bit dan ukuran kunci 64-bit. HIFAT merupakan sebuah block cipher yang terdiri atas operasi sederhana yang membuatnya cocok dijalankan pada perangkat dengan daya komputasi rendah. **Keywords**: *block cipher, ciphertext, plaintext,* enkripsi, dekripsi.

1. Pendahuluan

Penerapan kriptografi menyediakan beragam layanan keamanan yang mencakup confidentiality, integrity, dan sebagainya, yang berperan penting dalam pengembangan sistem informasi digital. Data berupa pesan dikirimkan lewat berbagai jaringan komunikasi tiap detiknya, dimana pesan-pesan tersebut dapat berupa hal sepele seperti gambar kucing lucu yang ingin dikirimkan kepada teman atau rahasia penting seperti kode peluncuran nuklir. Kriptografi memainkan peran penting dalam menjaga kerahasiaan dan keaslian dari pesan-pesan tadi. Salah satu dari banyaknya tipe kriptografi adalah *block cipher*, yang merupakan algoritma kriptografi simetrik yang mengenkripsi pesan dengan membaginya ke dalam blok-blok dengan ukuran tertentu, yang dimanfaatkan dalam algoritmanya untuk menyulitkan analisis oleh kriptanalis.

Beberapa contoh algoritma block cipher yang terkenal adalah DES, GOST, dan AES. Ketiga algoritma tersebut memiliki prinsip yang mirip, karena AES dan GOST merupakan bentuk pengembangan dari DES yang masih memiliki beberapa kelemahan.

HIFAT merupakan algoritma block cipher yang dibangun dengan menggunakan prinsip *confusion, diffusion*, s-box dan jaringan feistel dengan mengambil inspirasi dari beberapa algoritma yang sudah ada, serta menerapkan imajinasi dari perancang.

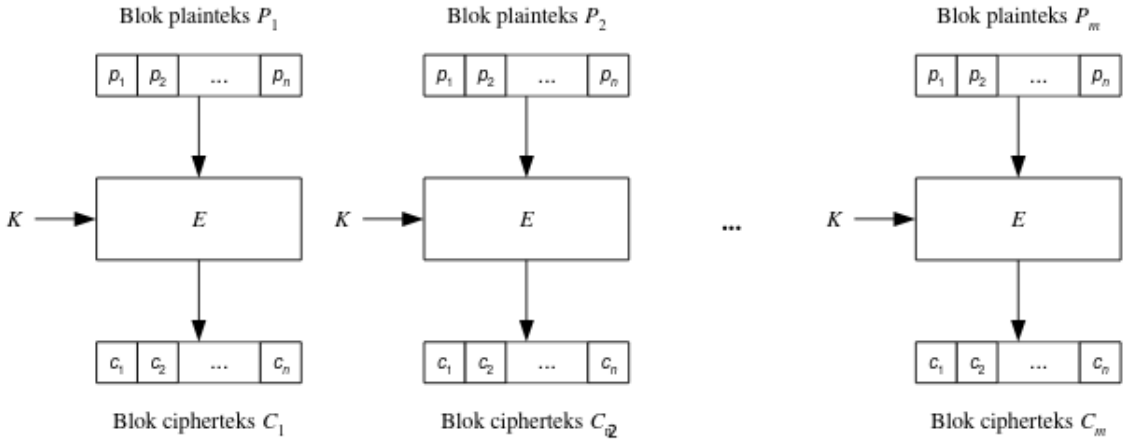
2. Studi Pustaka

A. Cipher Blok

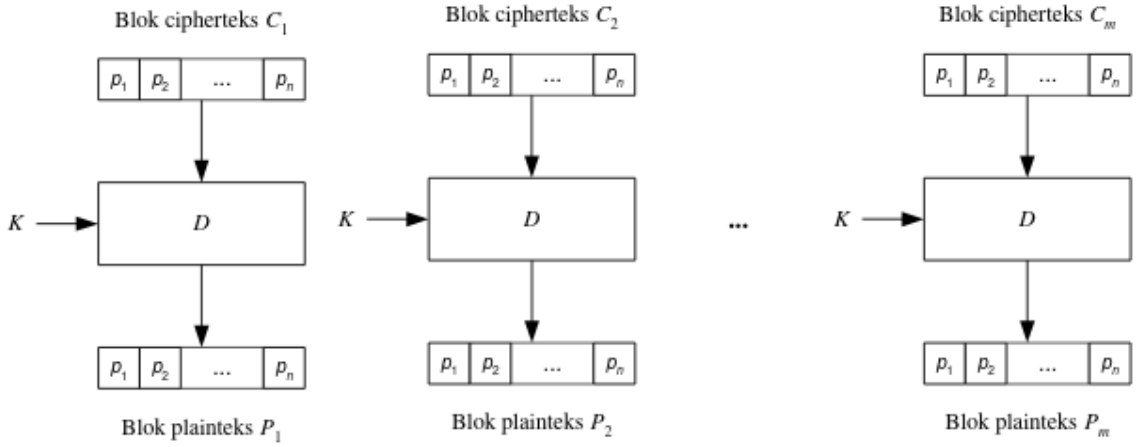
Cipher blok merupakan salah satu bentuk dari kriptografi simetri, dimana rangkaian bit dari plainteks dibagi menjadi blok bit dengan panjang yang sama sebelum dilakukan enkripsi. Hasil dari algoritma enkripsi pada cipher blok akan memiliki panjang bit yang sama dengan bit plainteks. Namun, ketika suatu blok plainteks yang dienkripsi menggunakan algoritma enkripsi yang sama, maka hasil enkripsi pada kedua blok tersebut juga akan sama. Oleh karena itu, terdapat lima mode operasi pada cipher blok yang dapat digunakan untuk membantu meningkatkan kompleksitas dari pengenkripsian blok plainteks.

1. Electronic Code Book (ECB)

Electronic Code Book (ECB) merupakan mode block cipher dimana setiap blok pesan dienkripsi secara individual dan terpisah, yang berarti enkripsi dan dekripsi suatu blok tidak bergantung pada blok-blok pesan lainnya.



Gambar II.1 Skema enkripsi dengan mode ECB



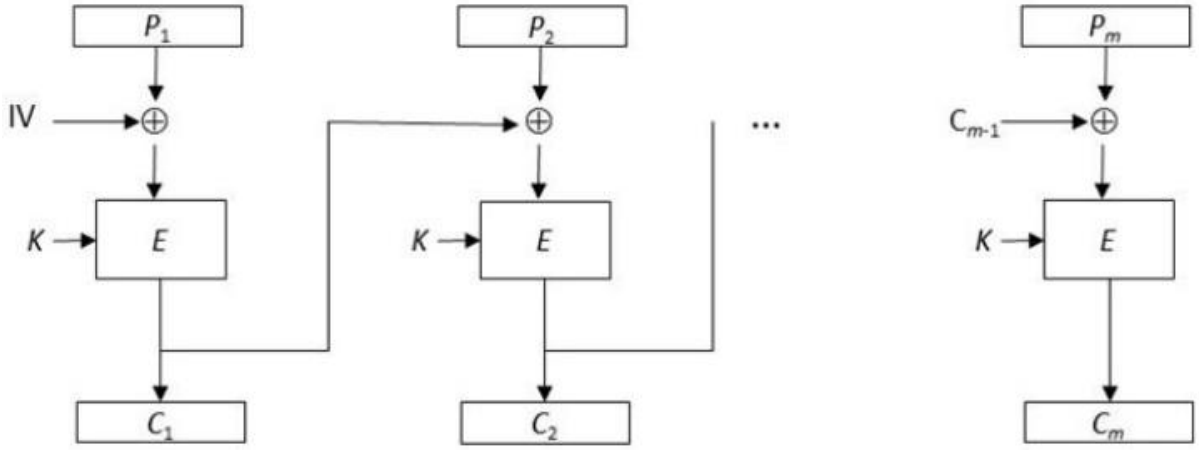
Gambar II.2 Skema dekripsi dengan mode ECB

Pada mode ECB, karena tiap blok dienkripsi secara independen, maka enkripsi tidak perlu dilakukan secara sekuensial, yang membuat mode ECB cocok digunakan untuk enkripsi arsip yang diakses secara acak. Selain itu, karena kesalahan pada satu atau lebih bit pada suatu blok ciphertext tidak memengaruhi blok ciphertext yang lain dalam proses dekripsi.

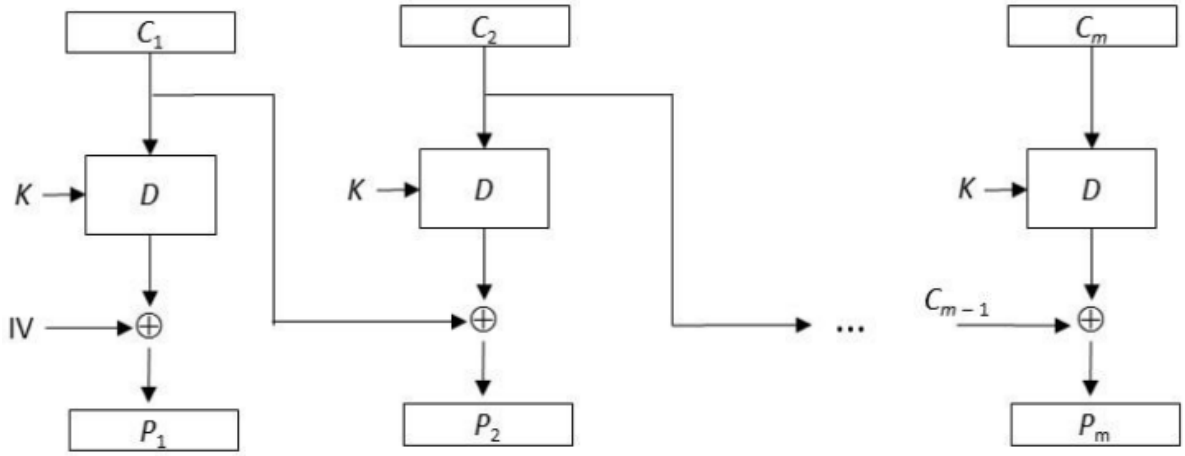
Enkripsi pada mode ECB hanya dilakukan dengan mengenkripsi blok pesan dengan kunci enkripsi yang menyebabkan terjadinya pengulangan, dimana blok pesan yang sama akan menghasilkan blok ciphertext yang sama. Pengulangan blok ciphertext ini membuat cipher blok mode ECB mudah diserang dengan menggunakan analisis frekuensi.

2. Cipher Block Chaining (CBC)

Cipher Block Chaining (CBC) merupakan mode block cipher dimana nilai dari blok ciphertext tidak hanya bergantung pada nilai dari blok pesan yang berkoresponden, tetapi juga bergantung pada seluruh blok pesan sebelumnya. Nilai dari blok ciphertext akan digunakan untuk proses enkripsi pada blok pesan selanjutnya.



Gambar II.3 Skema enkripsi dengan mode CBC



Gambar II.4 Skema enkripsi dengan mode CBC

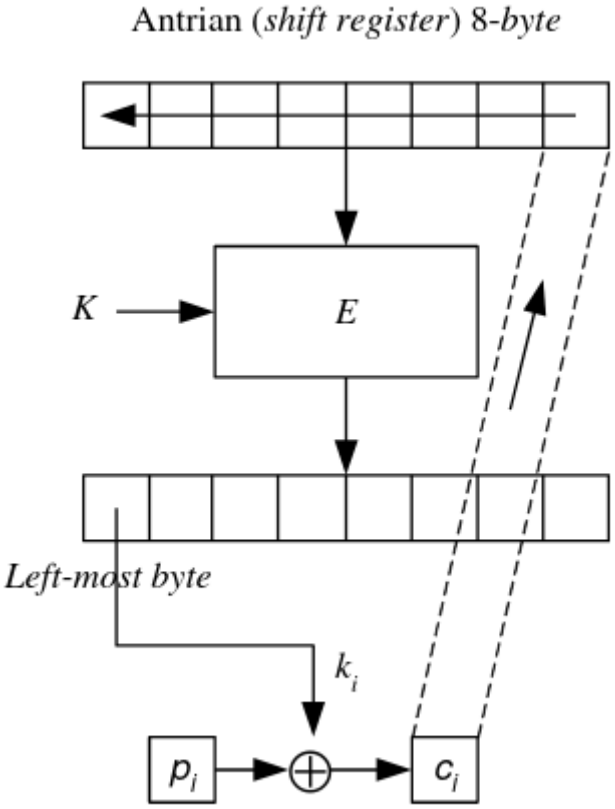
Enkripsi pada blok pesan pertama membutuhkan blok semu yang disebut *IV* (*initialization vector*), dimana nilai IV dapat diberikan oleh pengguna atau dibangkitkan secara acak oleh pengguna. Dimana blok pesan pertama akan di-xor dengan IV sebelum dienkripsi dengan menggunakan kunci enkripsi. Hasil enkripsi yang merupakan blok ciphertext pertama, digunakan sebagai xor key untuk blok pesan kedua, sebelum dienkripsi begitu seterusnya.

Blok-blok ciphertext yang didapat dengan menggunakan mode CBC tidak selalu sama untuk blok pesan yang sama, yang membuat CBC tidak rentan terhadap serangan analisis frekuensi. Kesalahan satu atau lebih bit pada blok pesan akan menghasilkan kesalahan pada blok ciphertext yang berkoresponden, yang kemudian akan merambat ke semua blok ciphertext berikutnya.

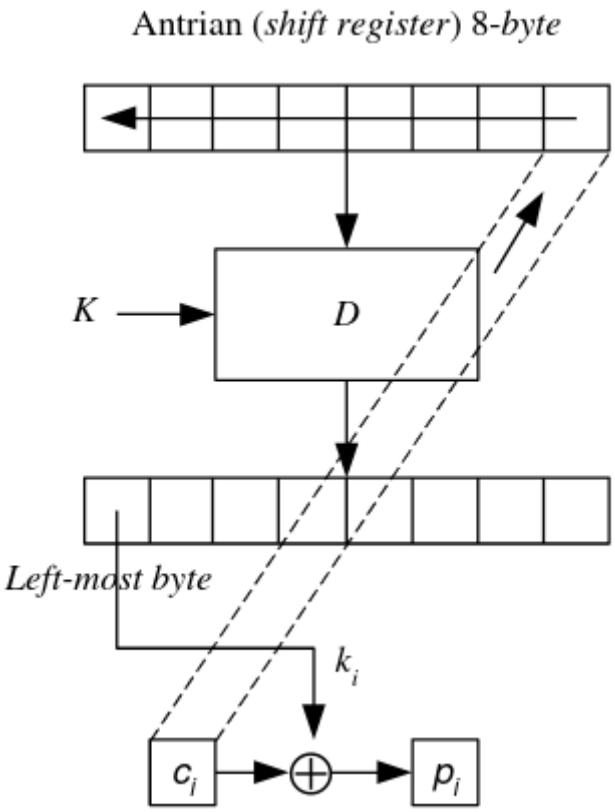
3. Cipher-Feedback (CFB)

Cipher-Feedback (CFB) merupakan mode block cipher yang mengatasi kekurangan pada mode CBC apabila diterapkan pada pengiriman data yang belum mencapai ukuran satu blok, dimana data dienkripsi dalam unit yang lebih kecil daripada ukuran blok. CFB n-bit mengenkripsi pesan sebanyak n bit setiap kalinya, dengan n ≤ m (m = ukuran blok). Dengan kata lain, CFB

n−bit memperlakukan block cipher seperti stream cipher. Mode CFB membutuhkan sebuah antrian yang berukuran sama dengan ukuran blok.



Gambar II.5 Skema enkripsi block cipher dengan mode CFB 8-bit



Gambar II.6 Skema dekripsi block cipher dengan mode CFB 8-bit

Secara formal, mode CFB n-bit dapat dinyatakan sebagai berikut

Proses enkripsi: $C_i = P_i \oplus MSB_m(E_K(X_i))$
$X_{i+1} = LSB_{m-n}(X_i) \parallel C_i$

Proses dekripsi: $P_i = C_i \oplus MSB_m(D_K(X_i))$
$X_{i+1} = LSB_{m-n}(X_i) \parallel C$

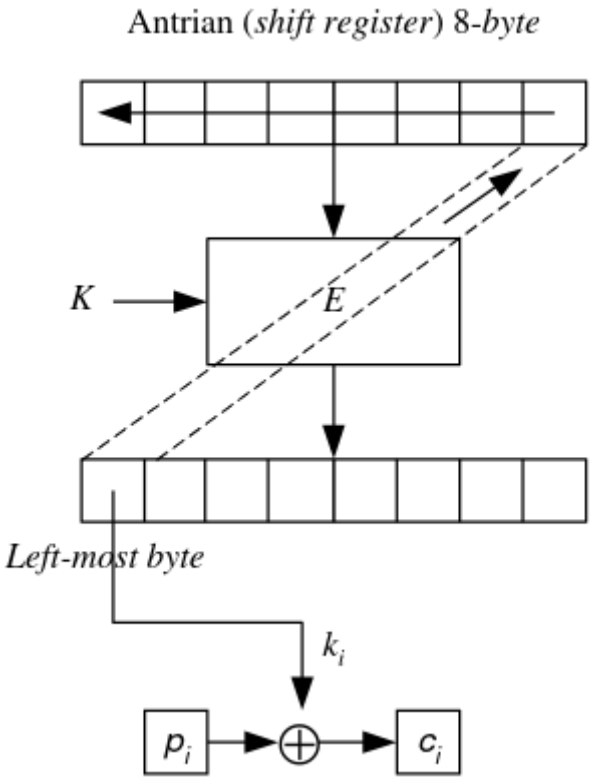
dengan

X_i = 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

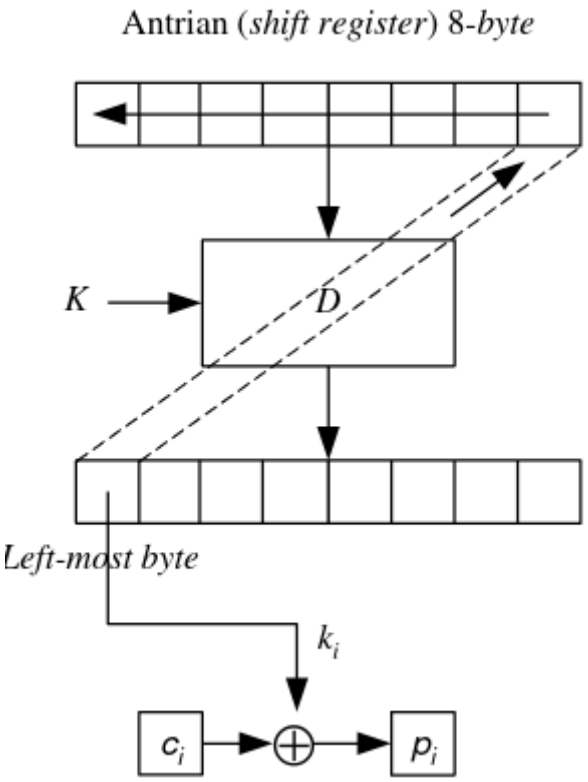
Kesalahan pada satu atau lebih bit pada blok pesan akan menghasilkan kesalahan pada blok ciphertext yang berkoresponden dan merambat pada blok-blok ciphertext berikutnya.

4. Output-Feedback (OFB)

Output-Feedback (OFB) merupakan sebuah mode yang mirip dengan mode CFB, kecuali n-bit dari hasil enkripsi antrian disalin menjadi elemen posisi paling akhir pada antrian.

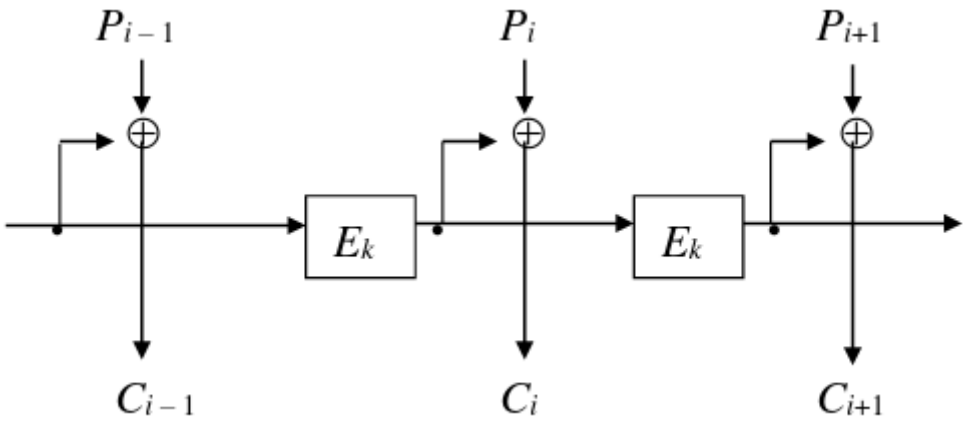


Gambar II.7 Skema enkripsi block cipher dengan mode OFB



Gambar II.8 Skema dekripsi block cipher dengan mode OFB

Jika m = n, maka mode OFB n-bit adalah seperti berikut

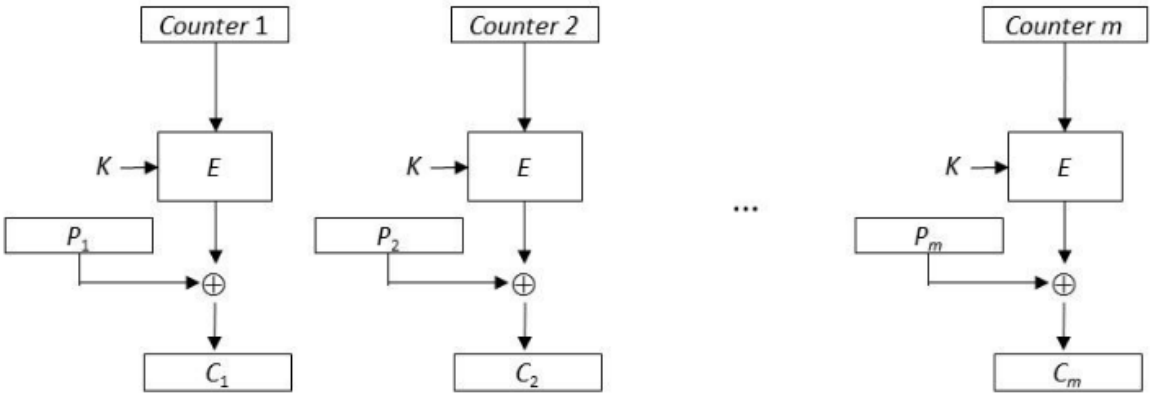


Gambar II.9 Skema enkripsi mode OFB n-bit untuk blok n-bit

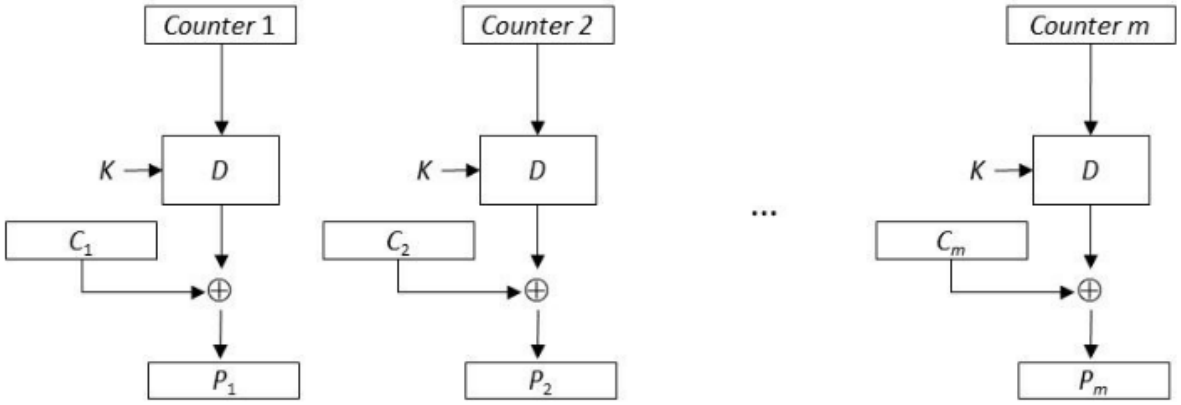
Kesalahan satu atau lebih bit pada suatu blok pesan hanya akan mempengaruhi blok ciphertext yang berkoresponden saja, begitu pula pada dekripsi. Karakterisitik yang dimiliki oleh mode OFB ini membuatnya cocok digunakan dalam transmisi analog yang digitisasi, seperti suara atau video, yang dalam hal ini kesalahan 1-bit dapat ditolerir, tetapi penjalaran tidak diperbolehkan.

5. Counter Mode

Mode counter tidak melakukan chaining seperti pada CBC. Counter merupakan sebuah nilai berupa blok bit yang ukurannya sama dengan ukuran blok pesan. Nilai counter harus berbeda dari setiap blok yang dienkripsi. Pada mulanya, untuk enkripsi blok pertama, counter diinisialisasi dengan sebuah nilai. Selanjutnya, untuk enkripsi blok-blok berikutnya counter dinaikkan nilainya satu.



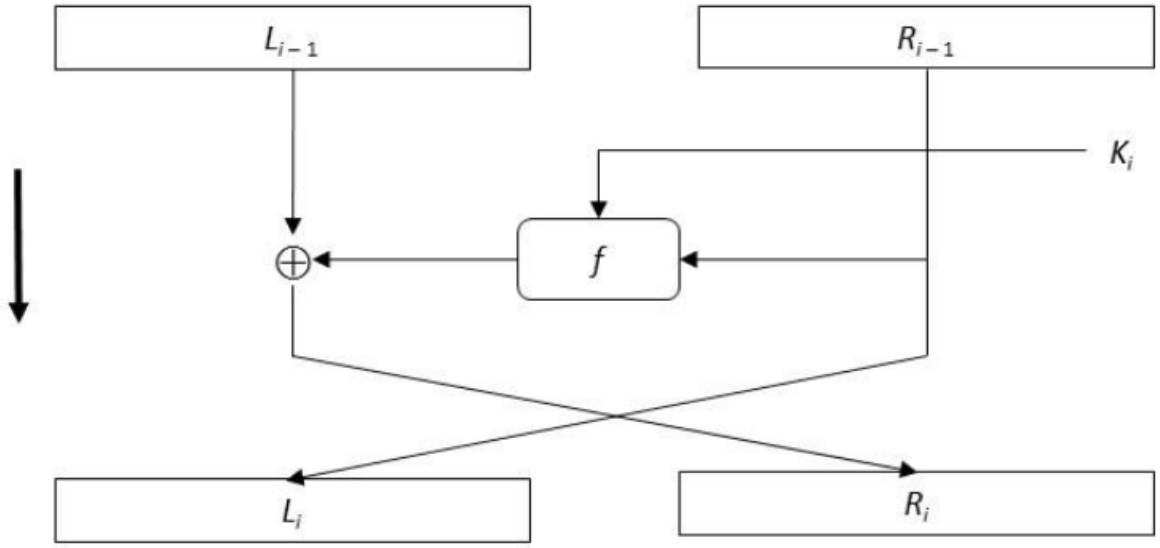
Gambar II.10 Skema enkripsi mode counter



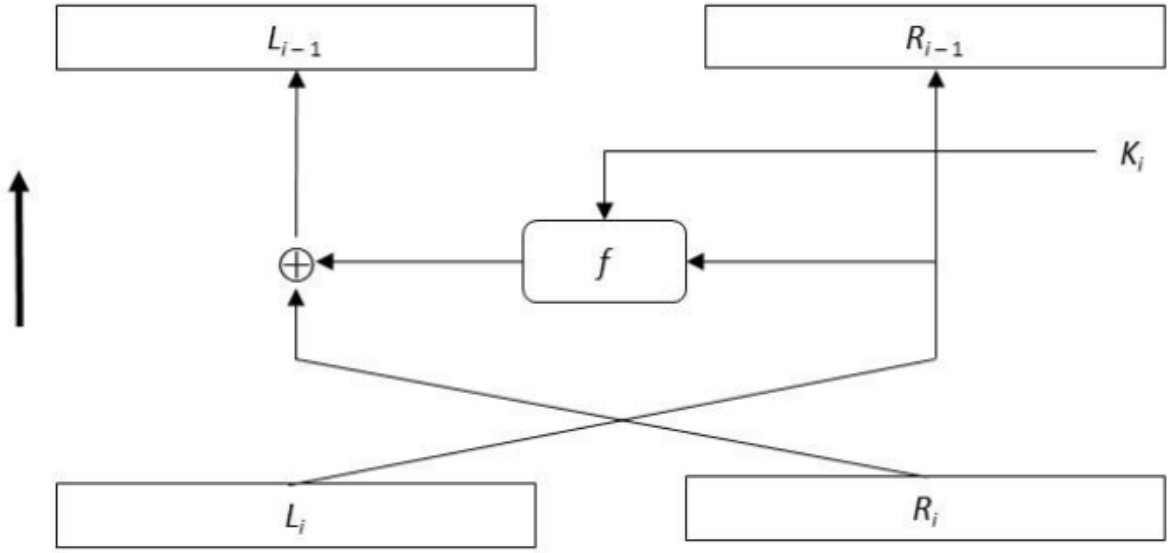
Gambar II.11 Skema dekripsi mode counter

B. Struktur Feistel

Jaringan Feistel banyak digunakan pada algoritma kriptografi DES, LOKI, GOST, FEAL, Blowfish, dan lain-lain karena prinsip ini bersifat reversible untuk proses enkripsi dan dekripsi. Sifat reversible ini membuat algoritma enkripsi dapat digunakan ulang sebagai algoritma dekripsi.



Gambar II.12 Skema enkripsi jaringan Feistel pada putaran ke-i



Gambar II.13 Skema dekripsi jaringan Feistel pada putaran ke-i

C. Prinsip Diffusion dan Confusion Shannon

Diffusion dan Confusion merupakan prinsip dalam kriptografi untuk membuat sebuah cipherteks menjadi lebih aman yang dikembangkan oleh Claude Shannon. Pada penerapannya, prinsip diffusion dan confusion digunakan untuk mempersulit terbongkarnya teknik enkripsi yang digunakan melalui deduksi atau potongan plainteks dari cipherteks.

1. Diffusion

Prinsip ini menyebarkan pengaruh dari satu bit plaintext atau kunci ke sebanyak mungkin ciphertext, sehingga sedikit perubahan yang terjadi pada plaintext akan menyebabkan perubahan pada ciphertext yang tidak dapat diprediksi. Prinsip diffusion hanya bisa diterapkan pada block cipher. Block cipher yang menerapkan prinsip ini adalah block cipher dengan mode CBC dan CFB. Diffusion dapat direalisasikan melalui algoritma transposisi.

2. Confusion

Prinsip confusion adalah prinsip untuk menyembunyikan hubungan apapun yang ada antara plaintext, ciphertext, dan kunci. Prinsip ini membuat algoritma kriptografi yang menerapkannya menjadi lebih tahan terhadap serangan analisis statistik. Prinsip confusion dapat diterapkan baik pada stream cipher maupun block cipher. Salah satu contoh algoritma yang menerapkan prinsip ini adalah *One-Time Pad*. Confusion dapat direalisasikan dengan menggunakan algoritma substitusi yang kompleks.

D. Cipher Berulang (Iterated Cipher)

Prinsip dari cipher berulang adalah cipher transformasi sederhana yang diulang beberapa kali, dimana pada setiap iterasi, digunakan subkey yang berbeda.

Cipher berulang dapat dinyatakan sebagai

$$C_i = f(C_{i-1}, K_i)$$

dengan

i = 1, 2, …, r (r merupakan jumlah putaran)
K_i = subkey putaran ke-i

f = fungsi transformasi (dapat berupa operasi substitusi, transposisi, ekspansi, dan/atau kompresi)

E. Kotak-S

Kotak-S merupakan sebuah matriks yang berisi substitusi sederhana yang memetakan satu atau lebih bit dengan satu atau lebih bit yang lain. Pada kebanyakan algoritma block cipher, kotak-S memetakan m bit masukan menjadi n bit keluaran, sehingga kotak-kotak tersebut dinamakan kotak m x n S-box.

Pada kotak-S, operasi dilakukan dengan melakukan look up table, dimana bit yang menjadi input diasumsikan sebagai indeks yang terdapat pada kotak-S dan output yang dihasilkan adalah nilai yang terdapat pada indeks yang ditunjuk tersebut. Salah satu dari kotak-S yang cukup terkenal adalah kotak-S Rijndael yang digunakan sebagai dasar dari algoritma kriptografi Advanced Encryption Standard (AES), seperti yang ditunjukkan pada gambar II.14.

hex	y															
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x 0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Gambar II.14 Kotak-S yang digunakan oleh algoritma AES

3. Proposed Block Cipher

Block cipher yang diajukan merupakan block cipher dengan ukuran block sebesar 64 bit. Jika pesan yang diinput memiliki panjang lebih dari 64 bit, maka pesan tersebut akan dibagi menjadi block yang berisi masing-masing 64 bit. Jika terdapat block yang memiliki ukuran kurang dari 64 bit, maka block tersebut akan ditambahkan padding 0 pada akhir block hingga mencapai ukuran 64 bit. Block cipher menerapkan skema jaringan feistel, dimana pada setiap ronde dilakukan pembangkitan kunci putaran untuk digunakan pada fungsi jaringan feistel. Pada inisialisasi jaringan feistel, block 64 bit dibagi menjadi 32 bit block kiri dan 32 bit block kanan seperti yang telah dijelaskan pada gambar II.12. Sementara itu, untuk fungsi yang digunakan akan dijelaskan lebih lanjut.

A. Pembangkitan Kunci Putaran

Kunci putaran dibangkitkan pada setiap ronde yang dijalani pada jaringan feistel. Untuk membangkitkan kunci putaran, dibutuhkan sebuah kunci dengan ukuran 64 bit yang diinput oleh pengguna. Kunci putaran digunakan untuk pada fungsi feistel, dimana digunakan juga bagian dari bit kanan block yang berukuran 32 bit. Dengan begitu, hasil dari pembangkitan kunci putaran akan sebesar 32 bit. Fungsi pembangkitan yang digunakan untuk kunci putaran adalah sebagai berikut:

1. Melakukan rotasi pada bit kunci input menggunakan circular left shift, dengan rumus sebagai berikut:

Ki = CircularLeftShift(K, ceil(i/2))

dengan

Ki = Kunci baru yang didapat setelah dilakukan circular left shift
K = Kunci input pengguna
i = ronde ke-i pada jaringan feistel

2. Membagi Ki menjadi 32 bit Ki sebelah kiri (Kil) dan 32 bit Ki sebelah kanan (Kir), dimana pembagian kunci mirip dengan pembagian block pada saat melakukan inisialisasi jaringan feistel
3. Untuk i bilangan ganjil, maka kunci yang diambil adalah 32 bit Ki sebelah kiri
4. Untuk i bilangan genap, maka kunci yang diambil adalah 32 bit Ki sebelah kanan

Berdasarkan pengamatan, maksimal akan didapat 64 kunci berbeda sebelum kembali pada kunci yang pertama kali didapatkan (dengan asumsi pada setiap pembangkitan kunci selalu didapatkan kunci yang unik)

B. Fungsi pada Jaringan Feistel

Pada block cipher yang dibuat, seluruh alur enkripsi yang digunakan dimasukkan ke dalam fungsi pada jaringan feistel. Sehingga, nantinya ketika melakukan proses enkripsi maupun dekripsi cukup menggunakan satu algoritma yang sama yang disimpan di dalam fungsi pada jaringan feistel. Berdasarkan gambar II.12, fungsi feistel memiliki dua parameter input, yang pertama adalah block sebelah kanan (ukuran block 32 bit) dan subkey ke i (ukuran subkey 32 bit). Fungsi yang digunakan pada jaringan feistel secara berurutan adalah:

1. Melakukan iterasi pada block kanan dengan mengambil 4 bit pada setiap iterasinya secara berurutan (ukuran block cipher yang digunakan menjadi 32 bit setelah block dibagi menjadi block kiri dan kanan), dimana untuk setiap iterasi ke-n digunakan aturan sebagai berikut terhadap 4 bit yang terpilih:
   a. Untuk n bilangan ganjil, maka dilakukan operasi pertukaran bit pertama dengan bit keempat, sebagai contoh jika terdapat bit 0 1 2 3 maka dilakukan operasi pertukaran menjadi bit 3 1 2 0,
   b. Untuk n bilangan genap, maka dilakukan operasi pertukaran bit kedua dengan bit ketiga, sebagai contoh jika terdapat bit 0 1 2 3 maka dilakukan operasi pertukaran menjadi bit 0 2 1 3.
2. Melakukan XOR dengan subkey yang didapat melalui pembangkitan kunci putaran, untuk ronde ke-i, jika i merupakan bilangan ganjil maka block kiri pada subkey akan digunakan sebagai pasangan XOR, sementara jika i merupakan bilangan genap maka block kanan pada subkey akan digunakan sebagai pasangan XOR dengan block.
3. Melakukan substitusi menggunakan kotak-S Rinjdael berukuran 16 x 16. Dalam implementasinya, diperlukan block berukuran 64 bit untuk melakukan substitusi, sementara block yang tersedia hanya berukuran 32 bit. Sehingga block ditambahkan 32 bit yang berasal dari pasangan subkey yang tidak digunakan pada ronde ini, agar block yang disubstitusi berukuran 64 bit.
4. Block yang telah disubstitusi diambil kembali sebesar 32 bit. Kemudian, block diubah menjadi dalam bentuk byte (4 byte) lalu dilakukan transposisi berdasarkan tabel yang telah didefinisikan sebagai berikut: [2 1 0 3].

Hasil dari fungsi feistel yang didapat nantinya akan di XOR dengan block sebelah kiri (ukuran block 32 bit) lalu disimpan sebagai block sebelah kanan pada ronde berikutnya, dan block sebelah kanan dijadikan block sebelah kiri pada ronde berikutnya. Jumlah ronde yang digunakan bergantung pada pengaturan yang diinginkan pengguna algoritma, namun disarankan tidak lebih dari 64 kali agar kunci pembangkitan yang didapat besar kemungkinan seluruhnya unik.

4. Eksperimen dan Analisis Hasil

Berdasarkan hasil eksperimen menggunakan pesan berikut:

0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x73 0x73 0x73 0x64 0x61 0x73 0x77 0x76 0x64 0x73 0x34 0x67 0x66 0x72 0x63 0x76

dengan kunci sebagai berikut:

0x71 0x77 0x65 0x72 0x74 0x79 0x75 0x69

di dapatkan hasil cipher untuk feistel round sebanyak 4 ronde dengan hasil sebagai berikut:

0x6e 0x66 0x6c 0x61 0xf 0x4 0xf 0xc 0x7e 0x7a 0x68 0x76 0x1b 0x9 0xd 0x12 0x6a 0x72 0x66 0x76 0x32 0x8 0x7 0x18

ketika salah satu byte pada pesan diubah menjadi seperti berikut:

0x7a 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x73 0x73 0x73 0x64 0x61 0x73 0x77 0x76 0x64 0x73 0x34 0x67 0x66 0x72 0x63 0x76

dengan menggunakan kunci yang sama, didapatkan hasil sebagai berikut:

0x63 0x61 0x74 0x7a 0x6a 0x73 0x7e 0x7e 0x73 0x7d 0x70 0x6d 0x7e 0x7e 0x7c 0x60 0x67 0x75 0x7e 0x6d 0x57 0x7f 0x76 0x6a

5. Kesimpulan dan Saran

HIFAT merupakan sebuah algoritma block cipher “baru” yang menerapkan operasi pengacakan sederhana sebagai implementasi dari prinsip confusion dan diffusion.

Sebagai saran pengembangan HIFAT, algoritmanya dapat dibuat sedikit lebih kompleks untuk mencegah serangan dari kriptanalis.

6. Daftar Referensi

[1] Munir, Rinaldi. 2020. Slide Kuliah IF4020 Kriptografi: Kriptografi Modern Bagian 3: Block Cipher.

[2] Munir, Rinaldi. 2020. Slide Kuliah IF4020 Kriptografi: Kriptografi Modern Bagian 4: Prinsip Perancangan Block Cipher.