Algoritma *Block Cipher*: Go-Block *Cipher*

Hansel Valentino Tanoto - 13520046 [1]
Ghebyon Tohada Nainggolan - 13520079 [2]
Afrizal Sebastian - 13520120 [3]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: [1] 13520046@std.stei.itb.ac.id, [2] 13520079@std.stei.itb.ac.id, [3] 13520120@std.stei.itb.ac.id

Abstract*—Block cipher adalah salah satu jenis algoritma kriptografi yang melakukan enkripsi dan dekripsi dengan terlebih dahulu mempartisi plaintext/ciphertext menjadi ukuran bit tertentu yang disebut blok. Go-Block Cipher merupakan sebuah algoritma block cipher yang beroperasi pada ukuran blok 128-bit dan menggunakan kunci eksternal yang rahasia sepanjang 128-bit juga. Algoritma ini menggunakan berbagai macam kombinasi operasi permutasi, substitusi, dan juga jaringan Feistel untuk memastikan adanya penerapan prinsip diffusion dan confusion pada ciphertext.

Kata Kunci*—block cipher; feistel; permutasi; substitusi; pembangkit kunci internal.

I. PENDAHULUAN

Dalam dunia kriptografi, *Block Cipher* merupakan salah satu teknik enkripsi yang sangat populer digunakan. Teknik ini memproses pesan dalam bentuk blok dengan ukuran tertentu dan mengubahnya menjadi pesan terenkripsi dengan menggunakan kunci enkripsi yang hanya diketahui oleh pihak tertentu. Penggunaan *Block Cipher* memungkinkan untuk mengamankan data dalam jumlah yang besar dan efisien.

Seiring dengan perkembangan teknologi, diperlukan adanya inovasi dan pengembangan baru dalam bidang *Block Cipher* untuk meningkatkan keamanan data. Makalah ini bertujuan untuk memperkenalkan sebuah *Block Cipher* baru yang dirancang dengan menggunakan pendekatan jaringan Feistel yang telah terbukti keamanannya. *Block Cipher* yang diusulkan memiliki ukuran *block* 128-bit dan *key* yang berukuran 128-bit atau lebih. Selain itu, teknik permutasi dan substitusi yang digunakan pada ronde enkripsi dan dekripsi dilakukan dengan men-*generate S-Box* dan *P-Box* dengan bantuan *Secure Hash Algorithm* (SHA) menggunakan *key* yang diberikan sebelumnya. Hal ini dilakukan untuk meningkatkan keamanan dan efisiensi pada pesan. *Block Cipher* ini diberi nama Go-Block *Cipher*.

Diharapkan dengan adanya *Block Cipher* baru ini, dapat memberikan alternatif yang lebih aman dan efisien dalam mengamankan dapa pada berbagai aplikasi digital dan pengaplikasian lainnya.

II. DASAR TEORI

A. *Block Cipher*

Block Cipher merupakan suatu bentuk kriptografi dengan menggunakan kunci simetrik. *Plaintext* yang ingin dienkripsi akan dibagi menjadi blok-blok bit dengan ukuran tiap blok yang sama panjang. Pada umumnya, ukuran blok yang digunakan adalah 64-bit, 128-bit, 256-bit. Enkripsi nantinya akan dilakukan pada setiap blok *plaintext* dengan menggunakan kunci. Kunci yang di-*input* oleh pengguna tidak harus memiliki panjang yang sama dengan panjang blok *plaintext*. Nantinya, kunci eksternal (*input* pengguna) akan digunakan untuk membangkitkan kunci internal.

Pada *block cipher* terdapat 5 mode operasi yang digunakan, yaitu *Electronic Code Book* (ECB), *Cipher Block Chaining* (CBC), *Cipher Feedback* (CFB), *Output Feedback* (OFB), dan *Counter Mode. Electronic Code Book* melakukan enkripsi secara individual dan independen pada setiap blok. *Cipher Block Chaining* melakukan enkripsi dengan setiap blok yang dihasilkan tidak hanya bergantung pada blok plainteks, tetapi juga pada seluruh blok plainteks sebelumnya. *Cipher Feedback* melakukan enkripsi dengan ukuran yang lebih kecil daripada ukuran blok dan memperlakukan *block cipher* sama seperti *stream cipher*. *Cipher Feedback* juga membutuhkan antrian yang berukuran sama dengan ukuran blok masukan. *Output Feedback* mirip dengan CFP, kecuali *r*-bit dari hasil enkripsi antrian disalin menjadi elemen posisi paling kanan di antrian. *Counter Mode* melakukan enkripsi dengan "*counter". Counter* adalah sebuah nilai berupa blok bit yang ukurannya sama dengan blok plainteks. Nilai *counter* harus berbeda dari setiap blok yang dienkripsi.

Terdapat beberapa *Block Cipher* yang sering digunakan pada saat ini, yaitu AES, Blowfish, DES/Triple DES, Serpent, Twofish, dll.

B. *Jaringan Feistel*

Jaringan Feistel adalah sebuah struktur kriptografi yang digunakan untuk mengenkripsi dan mendekripsi data. Struktur ini terdiri dari blok data uang dipecah menjadi dua bagian dengan ukuran yang sama yaitu bagian kiri dan bagian kanan. Lalu bagian kanan diubah dengan menggunakan suatu fungsi non-linear yang menghaislkan *output* yang berbeda-beda setiap

kali dijalankan dengan *input* yang berbeda. Kemudian *output* tersebut digabungkan dengan bagian kiri dan dilakukan pertukaran posisi antara bagian kiri dan kanan. Proses ini diulangi sebanyak ronde enkripsi yang ditentukan, dimana setiap ronde menggunakan kunci yang berbeda.

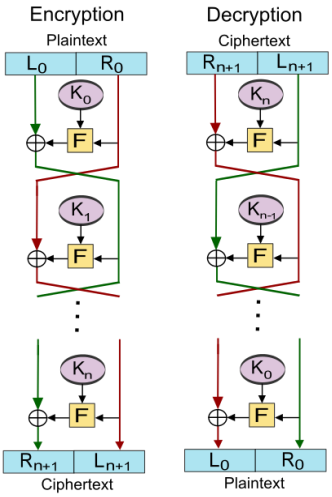

Fig. 1. Struktur jaringan Feistel

C. *Prinsip Diffusion dan Confusion Shannon*

Prinsip *diffusion* dan *confusion* merupakan 2 buah prinsip kunci dalam mendesain sebuah algoritma kriptografi yang diperkenalkan oleh Claude Shannon pada tahun 1949 dalam makalah klasiknya yang berjudul *Communication Theory of Secrecy System*.

Prinsip *diffusion* bertujuan untuk memastikan bahwa setiap perubahan 1-bit saja pada *plaintext* atau kunci memiliki pengaruh yang besar pada bit-bit *ciphertext*. Biasanya prinsip ini diterapkan dengan menggunakan operasi permutasi atau transposisi secara berulang sehingga bisa mengacak susunan bit dalam *plaintext*. Sedangkan, prinsip *confusion* bertujuan untuk membuat hubungan statistik (*pattern*) antara *plaintext*, *ciphertext*, dan kunci menjadi serumit mungkin sehingga menyulitkan proses kriptanalisis. Umumnya, prinsip ini direalisasikan dengan melakukan operasi substitusi non-linear dengan *lookup table*, yang biasa disebut *S-Box*.

Dengan kombinasi kedua prinsip ini, diharapkan bisa diperoleh tingkat keamanan yang tinggi terhadap percobaan kriptanalisis seperti teknik *bruteforce* dan analisis frekuensi.

III. RANCANGAN BLOCK CIPHER

A. *Algoritma Enkripsi dan Dekripsi*

Secara umum, algoritma yang dibentuk menggunakan tiga tahapan. Di awal digunakan blok pesan sebesar 128 bit. Tahapan pertama yang dilakukan adalah melakukan *Initial Permutation*. Tahapan *Initial Permutation* menggunakan *P-Box* yang di-*generate* dengan memanfaatkan perhitungan *Secure Hash Algorithm*. Pada awalnya, *P-Box* diisi dengan nilai indeks yang berurutan dari 0 hingga *size*-1. Selanjutnya, dilakukan iterasi sebanyak *size* kali, dan pada setiap iterasi, dilakukan penghitungan *hash* SHA-256 dari *key* yang diberikan ditambah dengan bilangan bulat i, dan hasilnya diubah menjadi bilangan bulat dalam basis 16. Nilai ini kemudian dimodulo dengan *size*-i untuk menghasilkan nilai j, yang merupakan indeks acak dalam *P-Box*. Dengan nilai j yang dihasilkan, dilakukan pertukaran nilai antara elemen pada indeks i dan indeks i+j dalam *P-Box*. Hal ini dilakukan untuk menciptakan permutasi susunan blok data *input* agar lebih random dan sulit ditebak. Tahapan kedua yang dilakukan adalah melakukan *enciphering* berulang sebanyak 16 kali dengan menggunakan Jaringan Feistel. Setelah melakukan *enciphering* dilakukan tahap ketiga yaitu *Final Substitution* menggunakan *S-Box* yang juga di-*generate* dengan memanfaatkan perhitungan *Secure Hash Algorithm*. *S-Box* di-*generate* dengan melakukan perhitungan hash SHA-256 dari *key* yang diberikan ditambah dengan bilangan bulat-i. Kemudian diambil 2 nilai terdepat dari hasil *hash* yang kemudian dimasukkan ke dalam *S-Box* apabila nilai tersebut belum ada di dalam *S-Box*. *S-Box* harus diisi dengan 256 nilai yang berbeda, oleh karena itu untuk melengkapi sisanya, dilakukan iterasi dari nilai 0 hingga 255, apabila nilai tersebut belum muncul di dalam *S-Box* maka nilai tersebut di-*append* ke dalam *S-Box*.

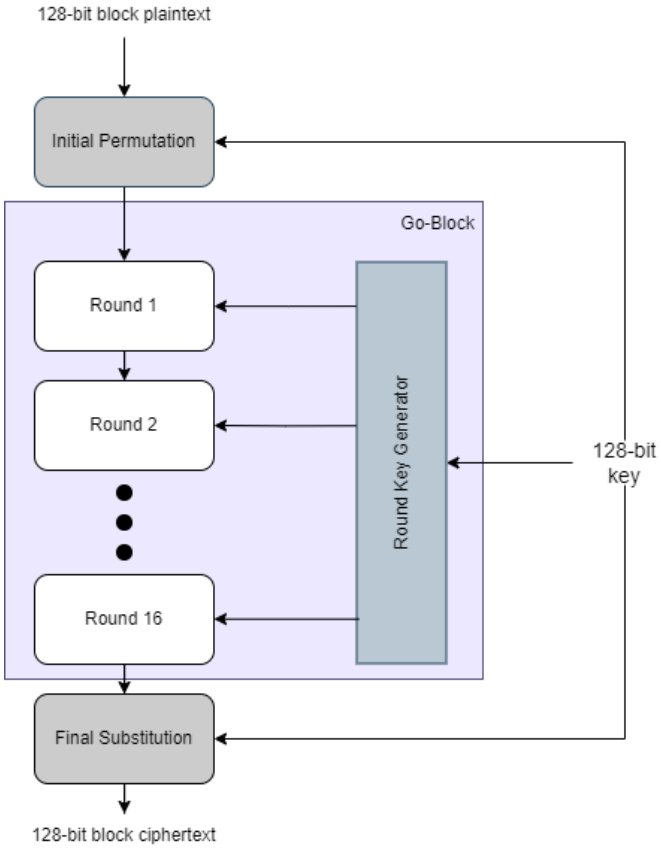

Fig. 2. Struktur algoritma Go-Block *Cipher*

B. *Pembangkit Kunci Internal*

Kunci internal dibangkitkan dengan menggunakan kunci eksternal yang diberikan oleh pengguna. Pada rancangan kali ini, kunci yang dibutuhkan untuk membangkitkan kunci internal adalah 128-bit atau 16 karakter. Namun, pengguna dapat memberikan kunci ekternal lebih dari 16 karakter. Kunci ekternal yang telah diberikan oleh pengguna akan menjadi *seed* untuk membuat *S-Box* yang nantinya akan digunakan untuk subtitusi.

Kunci ekternal yang diberikan akan diambil 128-bit pertama untuk diolah menjadi kunci internal. Kunci tersebut akan dibagi menjadi matriks berukuran 4x4 yang setiap selnya memiliki ukuran 8-bit dalam bentuk heksadesimal. Pada matriks akan dilakukan *shift* yang sirkluler.

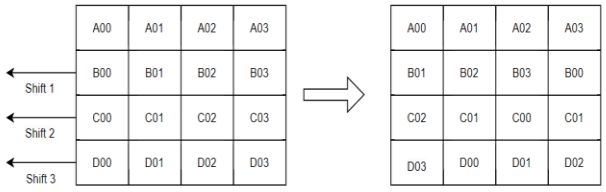

Fig. 3. *Shift* pada *Current Key*

Setelah melakukan *shift*, masing-masing sel akan disubtitusi dengan nilai yang terdapat pada *S-Box*. Untuk mempermudah penggunaan, *S-Box* dibentuk dalam array 1 dimensi dan dibentuk menggunakan fungsi *hash SHA-256*. Niali pada *S-Box* akan terus berganti sesuai dengan kunci eksternal yang diberikan pengguna sebagai *seed* untuk *S-Box*. Sehingga nilai pada sel akan menjadi indeks pada *S-Box* lalu nilai pada sel akan diganti dengan nilai yang terdapat pada *S-Box*.

Setelah melakukan substitusi, akan dilakukan operasi XOR pada baris pertama dan ketiga untuk membentuk baris pertama pada kunci internal yang baru. Lalu hasil XOR baris kedua dan baris pertama kunci internal yang baru akan membentuk baris kedua pada kunci internal yang baru.

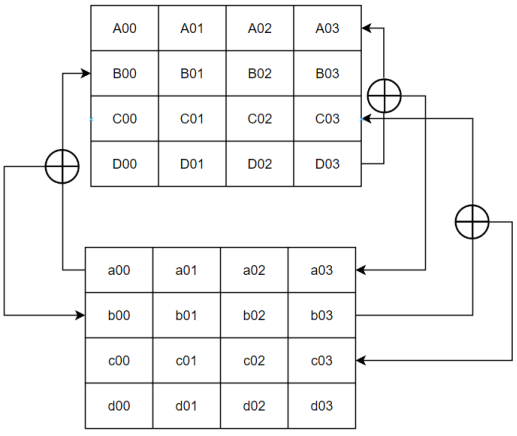

Fig. 4. Melakukan XOR pada baris-baris yang ada untuk membentuk baris yang baru

Setelah melakukan operasi XOR, nilai-nilai yang ada pada sel akan di-*concat* untuk membentuk kunci internal yang memiliki panjang 128-bit. Hal tersebut dilakukan berulang kali hingga mendapatkan kunci internal sebanyak putaran yang dilakukan.

C. *Jaringan Feistel*

Seperti yang sudah dijelaskan sebelumnya, setiap blok *plaintext* nantinya akan dibagi menjadi 2 segmen, yaitu bagian kiri L_i dan bagian kanan R_i dengan i merupakan indeks iterasi saat itu). Kemudian, keduanya akan diproses melalui jaringan Feistel. Jaringan Feistel yang digunakan untuk proses enkripsi pada algoritma Go-Block *Cipher* ditunjukkan oleh gambar berikut ini.

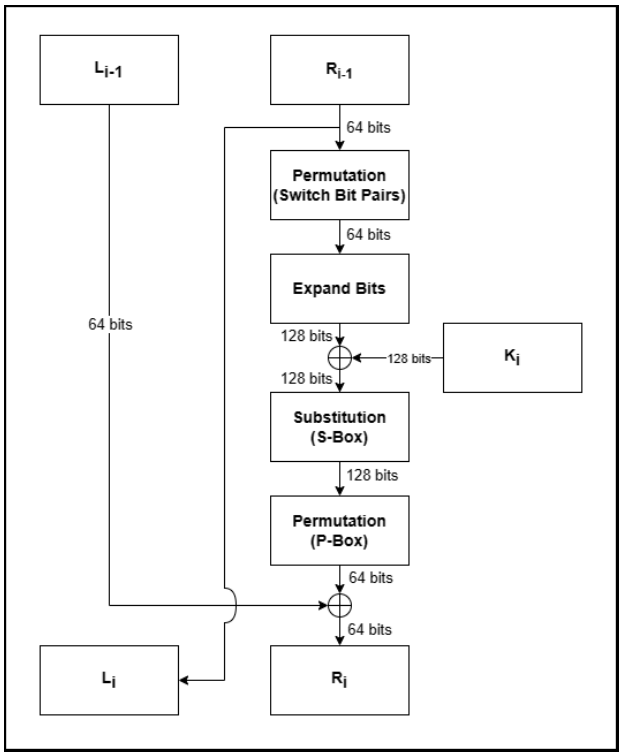

Fig. 5. Jaringan Feistel enkripsi algoritma Go-Block *Cipher*

Berdasarkan skema jaringan Feistel di atas, segmen L_i dapat langsung diperoleh dari R_{i-1}, yaitu segmen bagian kanan pada iterasi sebelumnya. Sementara itu, R_i akan diperoleh setelah melalui 6 langkah enkripsi, yaitu permutasi (*switch bit pairs*), ekspansi bit, operasi XOR dengan kunci internal ke-I, substitusi dengan *S-Box*, permutasi dengan *P-Box*, dan terakhir operasi XOR dengan L_{i-1}. Berikut adalah penjelasan lengkap dari setiap langkah.

1. Permutasi (*Switch Bit Pairs*)

   Pada tahap ini segmen R_{i-1} akan dibagi-bagi menjadi pasangan bit ganjil dan pasangan bit genap. Pasangan bit ganjil adalah sepasang bit (2-bit) dengan indeks 0-1, 4-5, 8-9, 12-13, dan seterusnya. Sedangkan pasangan bit genap adalah sepasang bit (2-bit) dengan indeks 2-3, 5-6, 10-11, 14-15, dan seterusnya. Kemudian akan dilakukan pertukaran posisi (*switch bit pairs*) antara

pasangan bit genap dengan pasangan bit ganjil sehingga diperoleh hasil dengan urutan indeks bit 2-3-0-1-5-6-4-5-10-11-8-9-14-15-12-13-…. Sebagai contoh, rangkaian bit 01110010 akan berubah menjadi 11011000. Dalam implementasinya, operasi ini dapat diimplementasikan dengan *bitwise operation* (& dan |) dan *bitwise shifting* (<< dan >>).

2. Ekspansi Bit

   Pada tahap ini, hasil operasi pada tahap sebelumnya yang masih berukuran 64-bit akan diekspansi menjadi 128-bit sebagai persiapan untuk melakukan operasi XOR dengan kunci K_i yang juga berukuran 128 bit. Ekspansi ini dilakukan dengan cara menambahkan hasil operasi XOR antara sepasang heksadesimal berurutan. Misalnya, terdapat rangkaian bit dalam format heksadesimal $H = H_1H_2H_3H_4$, maka akan dilakukan operasi XOR antara 2 heksadesimal berurutan secara siklik sehingga diperoleh hasil $X = X_1X_2X_3X_4$ dengan ketentuan:

   - X_1 adalah hasil XOR antara H_1 dengan H_2,
   - X_2 adalah hasil XOR antara H_2 dengan H_3,
   - X_3 adalah hasil XOR antara H_3 dengan H_4, dan
   - X_4 adalah hasil XOR antara H_4 dengan H_1.

   Selanjutnya akan dilakukan *merging* antara H dengan X sehingga diperoleh hasil $Y = H_1X_1H_2X_2H_3X_3H_4X_4$. Sebagai contoh, rangkaian heksadesimal 0x6865 akan menjadi 0x6E8E6353.

3. XOR (*Exclusive Or*) dengan K_i

   Hasil dari tahap sebelumnya akan di-XOR-kan secara *bitwise* dengan kunci internal yang dibangkitkan untuk iterasi Feistel ke-i, yaitu kunci K_i.

4. Substitusi dengan *S-Box*

   Teks hasil operasi tahap sebelumnya kemudian akan disubstitusi menggunakan *S-Box* berikut.

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

Fig. 6. *S-Box* untuk substitusi pada jaringan Feistel

   Untuk melakukan substitusi, teks akan dibagi-bagi terlebih dahulu menjadi ukuran 8-bit (1 *byte*) dalam representasi heksadesimal. Menggunakan representasi tersebut, setiap *byte* akan disubstitusi dengan nilai pada *S-Box* berdasarkan ketentuan:

   - Heksadesimal pertama merepresentasikan baris pada *S-Box* dan
   - Heksadesimal kedua merepresentasikan kolom pada *S-Box*.

   Misalnya, *byte* 0x46 akan disubstitusi dengan *byte* 0xd9 seperti ditunjukkan gambar di atas.

5. Permutasi dengan *P-Box*

   Selanjutnya, teks hasil substitusi akan dipermutasi dengan bantuan *P-Box* berikut ini.

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

Fig. 7. *P-Box* untuk permutasi pada jaringan Feistel

   P-Box tersebut berisi indeks urutan penempatan (pengaturan ulang posisi) setiap heksadesimal dari total 128-bit (32 bilangan heksadesimal) yang ada. Cara membacanya adalah mulai dari sel kiri atas ke kanan lalu dilanjutkan ke baris berikutnya. Setelah disusun ulang dengan konfigurasi tersebut, akan dilakukan proses reduksi menjadi 64-bit agar dapat dilakukan operasi XOR dengan L_{i-1} pada tahap selanjutnya. Proses reduksi tersebut dilakukan dengan melakukan operasi XOR terhadap bilangan heksadesimal bagian kiri dengan bilangan heksadesimal bagian kanan sedemikian sehingga jika terdapat suatu teks dalam format heksadesimal $H = H_1H_2H_3H_4H_5H_6$, maka akan diperoleh teks baru $Z = Z_1Z_2Z_3$ dengan ketentuan:

   - Z_1 adalah hasil XOR antara H_1 dengan H_6,
   - Z_2 adalah hasil XOR antara H_2 dengan H_5, dan
   - Z_3 adalah hasil XOR antara H_3 dengan H_4.

6. XOR (*Exclusive Or*) dengan L_{i-1}

   Hasil dari permutasi pada tahap sebelumnya akan di-XOR-kan secara *bitwise* dengan segmen L_{i-1} sehingga akhirnya diperoleh nilai R_i.

Jaringan Feistel ini akan diulang sebanyak 16 kali untuk setiap blok *plainteks* dengan kunci berbeda untuk setiap iterasi, sesuai hasil pembangkitan kunci. Karena menggunakan jaringan Feistel, salah satu kelebihannya adalah algoritma dekripsi akan sama saja seperti enkripsi, tetapi menggunakan urutan kunci yang terbalik. Berikut adalah skema jaringan Feistel untuk proses dekripsi.

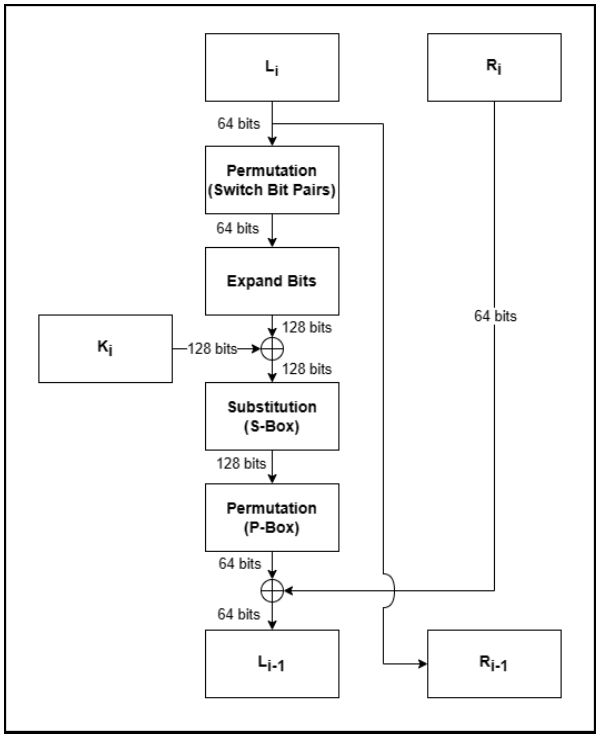

Fig. 8. Jaringan Feistel dekripsi algoritma Go-Block *Cipher*

IV. EKSPERIMEN DAN PEMBAHASAN

A. Waktu Enkripsi dan Dekripsi

Pesan dengan ukuran *small* (173 karakter):

> Institut Teknologi Bandung (ITB) adalah salah satu perguruan tinggi terkemuka di Indonesia yang berfokus pada pendidikan dan penelitian di bidang sains, teknologi, dan seni.

Fig. 9. Sampel uji ukuran *small*

Pesan dengan ukuran *medium* (407 karakter):

> Institut Teknologi Bandung (ITB) adalah salah satu perguruan tinggi terkemuka di Indonesia yang berfokus pada pendidikan dan penelitian di bidang sains, teknologi, dan seni. Didirikan pada tahun 1920, ITB memiliki reputasi yang kuat dalam menghasilkan lulusan berkualitas yang mampu bersaing di tingkat global. Kampus ITB terletak di Kota Bandung, Jawa Barat, dan mencakup area seluas lebih dari 770 hektar.

Fig. 10. Sampel uji ukuran *medium*

Pesan dengan ukuran *large* (1214 karakter):

> Institut Teknologi Bandung (ITB) adalah salah satu perguruan tinggi terkemuka di Indonesia yang berfokus pada pendidikan dan penelitian di bidang sains, teknologi, dan seni. Didirikan pada tahun 1920, ITB memiliki reputasi yang kuat dalam menghasilkan lulusan berkualitas yang mampu bersaing di tingkat global. Kampus ITB terletak di Kota Bandung, Jawa Barat, dan mencakup area seluas lebih dari 770 hektar. Institut Teknologi Bandung (ITB) adalah salah satu perguruan tinggi terkemuka di Indonesia yang berfokus pada pendidikan dan penelitian di bidang sains, teknologi, dan seni. Didirikan pada tahun 1920, ITB memiliki reputasi yang kuat dalam menghasilkan lulusan berkualitas yang mampu bersaing di tingkat global. Kampus ITB terletak di Kota Bandung, Jawa Barat, dan mencakup area seluas lebih dari 770 hektar. ITB memiliki berbagai fakultas yang menawarkan program studi di berbagai disiplin ilmu, seperti teknik, sains, seni dan desain, manajemen, dan humaniora. Program studi yang tersedia di ITB mencakup berbagai tingkat pendidikan, mulai dari sarjana, magister, hingga doktoral. Selain itu, ITB juga memiliki program studi internasional yang menarik mahasiswa dari seluruh dunia untuk belajar di kampus.

Fig. 11. Sampel uji ukuran *large*

Pesan dengan ukuran *very large* (3077 karakter):

> Institut Teknologi Bandung (ITB) adalah salah satu perguruan tinggi terkemuka di Indonesia yang berfokus pada pendidikan dan penelitian di bidang sains, teknologi, dan seni. Didirikan pada tahun 1920, ITB memiliki reputasi yang kuat dalam menghasilkan lulusan berkualitas yang mampu bersaing di tingkat global. Kampus ITB terletak di Kota Bandung, Jawa Barat, dan mencakup area seluas lebih dari 770 hektar. ITB memiliki berbagai fakultas yang menawarkan program studi di berbagai disiplin ilmu, seperti teknik, sains, seni dan desain, manajemen, dan humaniora. Program studi yang tersedia di ITB mencakup berbagai tingkat pendidikan, mulai dari sarjana, magister, hingga doktoral. Selain itu, ITB juga memiliki program studi internasional yang menarik mahasiswa dari seluruh dunia untuk belajar di kampus. ITB juga memiliki berbagai pusat penelitian yang terkenal, seperti Pusat Studi Energi, Pusat Studi Kebencanaan, dan Pusat Studi Transportasi. Pusat penelitian ini bertujuan untuk menghasilkan penelitian yang inovatif dan bermanfaat bagi masyarakat, serta memberikan kesempatan bagi mahasiswa untuk terlibat langsung dalam penelitian yang sedang dilakukan. Sebagai institusi yang memiliki peran penting dalam menghasilkan sumber daya manusia dan penelitian yang berkualitas, ITB juga terus berupaya untuk meningkatkan kualitas pendidikan dan penelitian yang dilakukan. ITB bekerja sama dengan berbagai institusi nasional dan internasional untuk memperluas jaringan, serta terus mengembangkan program dan infrastruktur yang ada untuk memenuhi tuntutan zaman dan memberikan manfaat bagi masyarakat. ITB memiliki banyak kegiatan ekstrakurikuler yang menarik dan beragam, salah satunya adalah kegiatan musik. Terdapat banyak grup musik yang terdiri dari mahasiswa ITB, mulai dari grup band rock hingga paduan suara. Kegiatan musik ini seringkali menjadi tempat mahasiswa untuk menyalurkan bakat dan hobi mereka, selain itu juga dapat menjadi wadah untuk menambah teman dan relasi. Selain kegiatan musik, ITB juga memiliki kegiatan olahraga yang cukup populer, seperti bulu tangkis, sepak bola, dan futsal. Mahasiswa ITB juga sering mengikuti kompetisi olahraga antar kampus di Indonesia, bahkan di tingkat internasional. Olahraga juga dijadikan sebagai media untuk membangun kerja sama dan persahabatan di antara mahasiswa ITB. Di ITB, juga terdapat banyak organisasi yang dapat diikuti oleh mahasiswa. Organisasi-organisasi ini dapat membantu mahasiswa untuk mengembangkan diri, baik dari segi keterampilan maupun kepribadian. Mahasiswa ITB dapat memilih organisasi yang sesuai dengan minat dan bakat mereka, seperti organisasi rohis, pecinta alam, hingga komunitas seni dan budaya. Dari kegiatan-kegiatan yang ada di ITB, terlihat bahwa ITB merupakan tempat yang cocok untuk para mahasiswa yang ingin mengembangkan diri dalam berbagai aspek. Di ITB, mahasiswa dapat mengeksplorasi berbagai bidang, baik itu dalam bidang akademis, seni, olahraga, maupun organisasi. Selain itu, di ITB juga terdapat atmosfer yang mendukung untuk pembelajaran dan pengembangan diri mahasiswa.

Fig. 12. Sampel uji ukuran *very large*

Hasil uji:

1. Pesan ukuran *small*
   Enkripsi : 0.014002323150634766 seconds
   Dekripsi : 0.018860340118408203 seconds
2. Pesan ukuran *medium*
   Enkripsi : 0.059897422790527344 seconds
   Dekripsi : 0.05061221122741699 seconds
3. Pesan ukuran *large*
   Enkripsi : 0.07773494720458984 seconds
   Dekripsi : 0.07306361198425293 seconds
4. Pesan ukuran *very large*
   Enkripsi : 0.17602753639221191 seconds
   Dekripsi : 0.21286749839782715 seconds

Dari haisl uji waktu tersebut, data dilihat bahwa Go-Block *Cipher* memiliki waktu eksekusi yang relatif cepat untuk ukuran pesan yang kecil hingga besar. Namun untuk pesan dengan ukuran *very large* dibutuhkan waktu yang cukup lama. Hal ini masih dalam batas wajar melihat kenaikan waktu eksekusi masih dalam pola linear.

B. Analisis Efek Longsoran (Avalanche Effect)

Efek longsoran adalah property kunci yang dinginkan semua algoritma enkripsi karena efek ini menunjukkan penerapan prinsip *diffusion* dan *confusion* yang sangat baik. Artinya, perubahan pada 1 bit di *plaintext* atau kunci bisa menghasilkan perubahan yang besar pada *ciphertext* sehingga menyulitkan proses penebakan kunci.

Algoritma Go-Block *Cipher* ini menunjukkan efek longsoran yang cukup baik yang bisa dilihat pada contoh berikut ini. Misalkan terdapat *external key* berupa “Ayo buat cipher block”, *plaintext* 1 berupa “block cipher DES”, dan *plaintext* 2 berupa “block cipher AES”.

Fig. 13. Eksekusi enkripsi untuk analisis efek longsoran (1)

Fig. 14. Eksekusi enkripsi untuk analisis efek longsoran (2)

Dapat dilihat pada contoh tersebut, perubahan 1 huruf saja pada *plaintext* yaitu antara huruf ‘D’ dan ‘A’ berakibat pada hasil *ciphertext* yang berubah secara drastis. Tabel di bawah ini menunjukkan hanya terdapat 1 heksadesimal dari *ciphertext* yang sama meskipun hanya terjadi 1 perubahan huruf pada *plaintext*.

Cipher 1	0xe0f8f2ab456cb77154e1fef208a5a115
Cipher 2	0x375f1e95df607b3037651b8ea3b9b760

Fig. 15. Perbandingan *ciphertext* akibat perubahan 1 huruf *plaintext*

Hal yang sama juga berlaku untuk perubahan pada *external key* seperti ditunjukkan pada hasil eksekusi berikut.

Fig. 16. Eksekusi enkripsi untuk analisis efek longsoran (3)

Bisa dilihat, perubahan 1 huruf saja pada kunci yaitu dari “Ayo” menjadi “Aye” menyebabkan perubahan *ciphertext* yang drastis. Tabel di bawah ini menunjukkan hanya terdapat 2 heksadesimal dari *ciphertext* yang sama meskipun hanya terjadi 1 perubah huruf pada *external key*.

Cipher 1	0xe0f8f2ab456cb77154e1fef208a5a115
Cipher 3	0x6949d5cebb4ef27d7b8ea9a21f96137f

Fig. 17. Perbandingan *ciphertext* akibat perubahan 1 huruf *external key*

Jadi, kesimpulannya, algoritma Go-Block *Cipher* memiliki efek longsoran yang baik sehingga lebih sulit untuk menemukan *pattern* untuk menebak kunci.

C. Analisis Ruang Kunci (Key Space)

Ruang kunci merupakan banyaknya kombinasi kunci internal yang dihasilkan oleh kunci eksternal yang diberikan oleh pengguna. Pada Algoritma Go-Block *Cipher* ini cukup menghasilkan variasi key internal yang baik, dengan contoh seperti berikut.

Fig. 18. Eksekusi enkripsi untuk analisis Ruang Kunci (1)

Fig. 19. Eksekusi enkripsi untuk analisis Ruang Kunci (2)

Dapat dilihat, perubahan 1 huruf saja pada kunci yaitu dari "Ayo" menjadi "Aya" menyebabkan perubahan yang cukup signifikan pada kunci internal, namun pada kunci internal pertama tidak terlalu berubah dikarenakan untuk kunci internal pertama langsung mengubah kunci ekternal menjadi hexadesimal

Kunci Ekternal	Ruang Kunci
Ayo Buat Cipher Block	['0x41796f204275617420436970686572204426c6f636b21', '0x6acda18f235d529a97f44f7b8888dbeb', '0x787d0b01af9b367b806aa69b4661ae02', '0x1b7ac161bcda334aab7c7ef76e60edd1', '0x61e6575ef861d410f68ef617766afcd8', '0x01c480209042dbd17b6f329aeb128e60', '0x4fa50f54593e8d60cd007f56c5c73f1d', '0x4f7f3bcd9fa33141dace455bbaa529b7', '0x6f0e05a962557246e9a2e93da9e9ffe2', '0x9b4aece73fd8f35efc163256520cf3aa', '0xb967695c2986b1b0bdeb4a7ba30de498', '0x97c1b009bcee98d63d1fc913ec13cda6', '0x859b29a34b4ba6ed3a4b6b6b2d8d681e', '0x3f1103731c07cd537b61f2737cd72d0d', '0xe972727522053163fdc5c3f1fbc04f45', '0x4865cd65653e52dd04e6abb48f10f598', '0xe10f2d7f31e9b0ea11d23150c04b6ab9']
Aya Buat Cipher Block	['0x417961204275617420436970686572204426c6f636b21' '0xb79b6066dc9994871491abd5291dc88e', '0xc2e75d310b1d21901392cfebefe99d84', '0x797766a8295f7154529cf27877513013', '0x960185aab800bbd364f132a5e595e908', '0x95c462d34e8be6909ec3b3449a7a9b87', '0x5422e219e49af516d1d79975af9769ea', '0xfd61d0ecbcbf8d5875d50dccb4f76807',
	'0x6d349d4a3b18d01b306bb996a85c5818', '0x10e300b2899e96c2cf0e3af2560852be', '0xf971ddba97e18392883cfa70ddfbfcf8', '0x01a74fbfb26862db586084c9bec76d3b', '0x0e821fdc839182eda6fdcfe1d5a9a46b', '0xd276a129aa8e64e4d13c8912e2b38599', '0x4af3a133b6a514209d2b9432549fa37b', '0x35e0a5357c0d9a0e87dcf0164ce1c3a7', '0x75b2f8c07ef21be5adac67447bafd627']

Fig. 20. Perbandingan pembangkitan kunci internal akibat perubahan 1 huruf kunci eksternal

D. Analisis Serangan Bruteforce

Algoritma Go-Block *Cipher* menggunakan kunci dengan Panjang 128-bit dalam proses enkripsi dan dekripsinya sehingga akan terdapat total $2^{128} \approx 3.4 \times 10^{38}$ kemungkinan kunci yang harus dicoba apabila ingin melakukan *bruteforce attack* (*exhaustive search*). Jika diasumsikan komputer saat ini bisa menghasilkan 10^9 (1 miliar) kunci per detik, maka total akan dibutuhan waktu sekitar 1.072×10^{21} tahun untuk menghasilkan semua kemungkinan kunci. Waktu tersebut bahkan jauh melebihi usia alam semesta yang diperkirakan baru berumur 13.8 miliar tahun sehingga dapat disimpulkan *bruteforce attack* tidak mungkin bisa memecahkan kunci dari algoritma ini.

Selain itu, karena terdapat berbagai operasi substitusi dan permutasi yang beragam, akan cukup sulit untuk menemukan pola enkripsi/dekripsinya dengan hanya menggunakan *bruteforce.*

V. KESIMPULAN DAN SARAN

Dari hasil eksperimen dan pembahasan, algoritma Go-Block *Cipher* sudah dapat dikatakan cukup baik dalam melakukan enkripsi. Dari sisi waktu enkripsi, algoritma yang diajukan dapat menyelesaikan enkripsi dengan waktu yang singkat untuk berbagai ukuran yaitu sekitar 0.01 - 0.17 detik untuk melakukan enkripsi dan 0.06 - 0.21 detik untuk melakukan dekripsi. Dari sisi efek longsoran yang dberikan, algoritma ini juga memberikan hasil yang baik. Perubahan satu kata pada plain teks atau kunci dapat menghasilkan *cipher* yang sangat berbeda pula. Dari sisi ruang kunci juga, algoritma ini menghasilkan ruang kunci yang sangat berbeda hanya dengan perubahan 1 kata pada kunci.

Saran pengembagan pada sisi pembuatan kunci internal, kunci internal pertama sebaiknya tidak langsung menggunakan kunci eksternal yang diberikan oleh pengguna yang diubah menjadi hexadecimal, tetapi diolah dan dibentuk terlebih dahulu. Sementara itu, dari sisi jaringan Feistel (fungsi putaran), sebaiknya S-Box dan P-Box yang digunakan tidak statis, melainkan dapat di-*generate* menggunakan *seed* tertentu, misalnya kunci internal pada saat iterasi tersebut. Namun, pasti akan terdapat *trade off* pada waktu eksekusi.

UCAPAN TERIMA KASIH

Pertama-tama, penulis ingin memanjatkan puji dan syukur kepada Tuhan Yang Maha Esa sehingga penulis bisa menyelesaikan pembuatan makalah ini. Penulis juga ingin mengucapkan terima kasih yang sebesar-besarnya kepada Bapak Dr. Ir. Rinaldi Munir, M.T. selaku dosen mata kuliah IF4020 Kriptografi atas ilmu dan bimbingan yang telah diberikan selama pembelajaran di kelas. Penulis juga ingin meminta maaf apabila terdapat kekurangan dalam penulisan makalah ini. Semoga isi dari makalah ini dapat bermanfaat dan bisa menambah wawasan pembaca.

REFERENSI

[1] Munir, R. (2023). 13-Cipher blok (block cipher) - Bagian 1 [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/13-Block-Cipher-Bagian1-2023.pdf

[2] Munir, R. (2023). 14-Perancangan cipher blok (block cipher) [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/14-Prancangan-block-cipher-2023.pdf

[3] Munir, R. (2023). 15-Review beberapa cipher blok (block cipher) - Bagian 1: DES, Triple DES [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/15-Beberapa-block-cipher-bagian1-2023.pdf

[4] Munir, R. (2023). 16-Review beberapa cipher blok (block cipher) - Bagian 2: GOST, RC5, RC6, Blowfish, Twofish, Mars, Serpent [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/16-Beberapa-block-cipher-bagian2-2023.pdf

[5] Munir, R. (2023). 17-Review beberapa cipher blok (block cipher) - Bagian 3: AES [Presentasi PowerPoint]. Diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/17-Beberapa-block-cipher-bagian3-2023.pdf