Algoritma Block Cipher YEA

Yoncé Encryption Algorithm

Garmastewira (13514068)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
ranggarmaste@gmail.com

Muhammad Gumilang
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
muhammadgumilang11@gmail.com

***Abstrak*—Makalah ini memperkenalkan *block cipher* baru bernama YEA (Yoncé Encryption Algorithm) yang merupakan modifikasi *block cipher* DES. Rancangan *block cipher* YEA memanfaatkan dua buah jaringan Feistel, permutasi rotasi matriks, penggunaan barisan Fibonacci untuk pembangkitan upakunci, dan pembangkitan S-Box secara acak semu. Berdasarkan hasil eksperimen dan analisis keamanan, *block cipher* YEA memenuhi prinsip *confusion* dan *diffusion* dengan baik.**

Kata kunci—block cipher; confusion; decryption; diffusion; encryption; Fibonacci; Feistel; permutation; pseudorandom; S-Box; seed.

I. PENDAHULUAN

Pada zaman ini, penggunaan internet semakin berkembang pesat. Tentunya, keamanan menjadi salah satu aspek penting yang harus ditangani ketika pengguna mentransmisikan data rahasia miliknya ke pengguna lain. Salah satu caranya adalah dengan menyandikan pesan agar pesan tersebut hanya dapat dibaca oleh penerima dan tidak dapat dibaca oleh pihak ketiga. Ilmu dan keahlian yang mempelajari tentang teknik menyandikan pesan disebut dengan kriptografi.

Kriptografi sudah dikenal sejak zaman dahulu. Bangsa Yunani menggunakan alat *scytale* pada tahun 400 SM, dan Nazi Jerman menggunakan cipher Enigma untuk mengenkripsi pesan pada perang dunia II [1]. Pendekatan ini dinamakan kriptografi klasik karena algoritma beroperasi pada tingkat karakter. Setelah era komputer dimulai, kriptografi kemudian beroperasi pada tingkat bit-bit data, atau disebut juga kriptografi modern. Algoritma kriptografi modern dapat beroperasi pada tingkat bit (*stream cipher*) atau pada tingkat blok bit (*block cipher*). Banyak sekali algoritma *block cipher* terkenal, seperti DES (Data Encryption Standard) dan AES (Advanced Encryption Standard).

Makalah ini memperkenalkan rancangan algoritma *block cipher* baru yang diberi nama YEA (Yoncé Encryption Algorithm). Karena algoritma *block cipher* kami menggunakan berbagai macam operasi, kami menamai algoritma kami Yoncé berdasarkan artis Beyoncé yang memiliki berbagai macam bakat. YEA pada dasarnya merupakan modifikasi algoritma DES. Rancangan *block cipher* YEA memanfaatkan penggunaan dua jaringan Feistel, permutasi dengan rotasi matriks, penggunaan barisan Fibonacci untuk pembangkitan upakunci, dan pembangkitan S-Box secara acak semu. Diharapkan rancangan *block cipher* YEA menjadi algoritma *block cipher* yang memenuhi prinsip *confusion* dan *diffusion* dengan baik.

II. DASAR TEORI

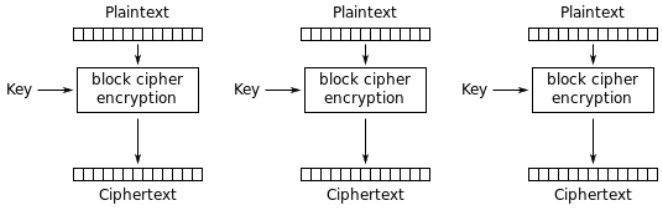
A. Block Cipher

Block cipher merupakan algoritma enkripsi yang beroperasi pada sekumpulan bit dengan suatu panjang yang disebut blok. Awalnya, plainteks dibagi menjadi blok-blok dengan panjang n bit. *Block cipher* kemudian mengenkripsi setiap blok plainteks menjadi blok cipherteks dengan panjang yang sama.

Pada *block cipher*, mode operasi adalah cara terdapat lima mode operasi [2]. Rinciannya adalah sebagai berikut.

1. Electronic Codebook (ECB)'

   Mode operasi ECB merupakan mode operasi yang paling sederhana. Pesan dibagi menjadi blok-blok plainteks, dan setiap blok plainteks diproses secara independen. Kelemahan dari mode operasi ini adalah blok plainteks yang sama akan dienkripsi menjadi blok cipherteks yang sama. Di sisi lain, enkripsi dan dekripsi untuk setiap blok dapat dilakukan secara paralel. Ilustrasi enkripsi mode ECB dapat dilihat pada Gambar 1.

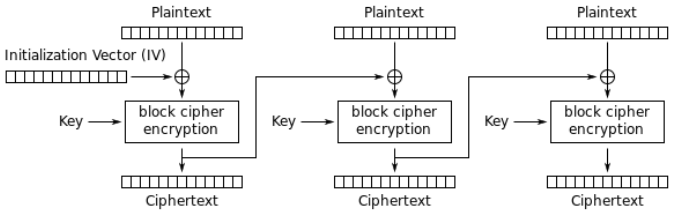


Gambar 1 Ilustrasi Enkripsi Mode ECB [2]

2. Cipher Block Chaining (CBC)

   Mode operasi CBC dirancang oleh Ehrsam, Meyer, Smith, dan Tuchman pada tahun 1976 [3]. Pada mode CBC, setiap blok plainteks di-XOR-kan dengan blok cipherteks sebelumnya sebelum dienkripsi. Khusus untuk blok plainteks pertama, blok akan di-XOR-kan dengan vektor inisialisasi (IV) yang dikonstruksi secara acak semu. Berbeda dengan ECB, blok plainteks yang sama mungkin menghasilkan cipherteks yang berbeda karena enkripsi suatu blok plainteks bergantung kepada

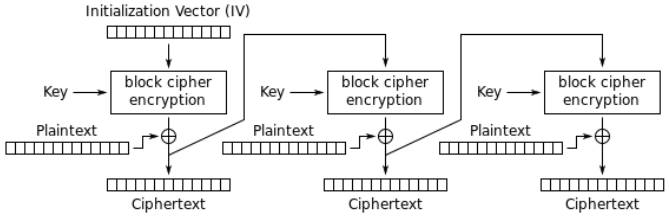
blok cipherteks sebelumnya. Ilustrasi enkripsi mode ECB dapat dilihat pada Gambar 2.



Gambar 2 Ilustrasi Enkripsi Mode CBC

3. Cipher Feedback (CFB)

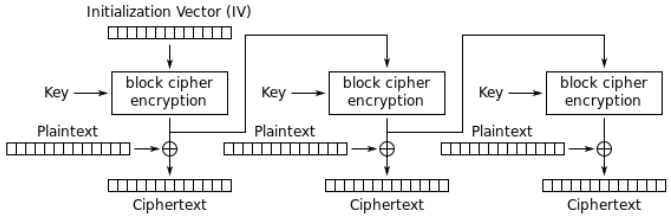
   Mode operasi CFB pada dasarnya mirip dengan CBC. Perbedaannya, data dienkripsikan dalam unit yang lebih kecil dari ukuran blok, yaitu sebesar n-bit, $n < m$ (m merupakan ukuran blok). Mode ini membutuhkan sebuah antrian sepanjang blok masukan. Setelah enkripsi blok ke-i, antrian digeser ke kiri sebanyak n kali, kemudian antrian paling kanan diisi dengan cipherteks. Ilustrasi enkripsi mode CFB dapat dilihat pada Gambar 3.



Gambar 3 Ilustrasi Enkripsi Mode CFB

4. Output Feedback (OFB)

   Mode operasi OFB mirip dengan mode CFB. Jika pada CFB n-bit antrian paling kanan dimasukkan dengan cipherteks, pada OFB, n-bit yang dimasukkan adalah n -bit hasil enkripsi terhadap antrian. Ilustrasi enkripsi mode OFB dapat dilihat pada Gambar 4.

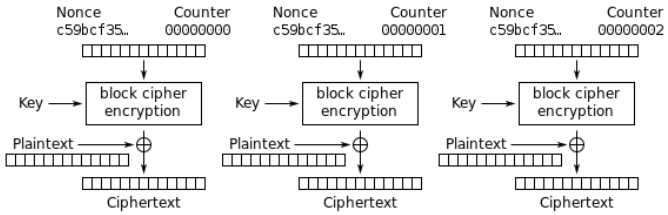


Gambar 4 Ilustrasi Enkripsi Mode OFB

   Mode operasi ini

5. Counter (CTR)

   Berbeda dengan mode operasi sebelumnya, mode CTR tidak melakukan perantaian terhadap blok sebelumnya. Blok yang dienkripsi merupakan blok *counter*. Untuk blok pertama, nilai *counter* diinisialisasi dengan suatu nilai, misalnya 0. Untuk enkripsi/dekripsi blok selanjutnya, nilai *counter* ditambah dengan satu. Cipherteks merupakan hasil operasi XOR antara plainteks dengan hasil enkripsi nilai *counter* ke-i. Ilustrasi enkripsi mode CTR dapat dilihat pada Gambar 5.



Gambar 5 Ilustrasi Enkripsi Mode CTR

B. S-Box

S-Box merupakan sebuah matriks berukuran $n \times m$ yang digunakan untuk mensubstitusi sekumpulan bit menjadi sekumpulan bit lainnya [4]. Panjang hasil substitusi dapat berbeda dengan panjang awal. Pada beberapa algoritma *block cipher*, S-Box umumnya merupakan matriks statis, seperti pada DES dan AES. Namun, terdapat juga algoritma *block cipher* yang membangkitkan matriks-matriks S-Box secara otomatis berdasarkan kunci yang diberikan pengguna. S-Box yang tepat dapat menjamin properti *confusion* pada hasil enkripsi.

C. Jaringan Feistel

Jaringan Feistel merupakan sebuah struktur simetrik yang digunakan untuk membangun *block cipher* [5]. Jaringan ini dinamakan berdasarkan penemunya, Horst Feistel. Banyak sekali *block cipher* yang menggunakan jaringan Feistel, salah satunya DES.

Perancang jaringan Feistel dapat mendefinisikan pembangkitan upakunci untuk enkripsi/dekripsi pada putaran ke-i serta fungsi putaran. Kelebihan dari jaringan Feistel adalah perancang dapat mendefinisikan fungsi putaran F yang tidak reversibel. Jaringan Feistel digunakan pada berbagai *block cipher* untuk menjamin properti *diffusion*.

Gambar 6 memperlihatkan mekanisme jaringan Feistel secara umum. Misalkan F adalah fungsi putaran dan $K_0, K_1, \ldots, K_n$ merupakan upakunci untuk putaran ke-0, 1, ..., n. Maka berikut adalah langkah-langkah untuk menghasilkan blok cipherteks dari blok plainteks.

1. Bagi plainteks menjadi dua blok sama panjang (L_0, R_0)
2. Untuk setiap putaran ke-$i = 0,1,\ldots,n$, hitung L_{i+1} dan R_{i+1} dengan cara sebagai berikut.

$$L_{i+1} = R_i \quad (1)$$
$$R_{i+1} = L_i \oplus F(R_i, K_i) \quad (2)$$

3. Cipherteks merupakan gabungan dua blok (R_{n+1}, L_{n+1})

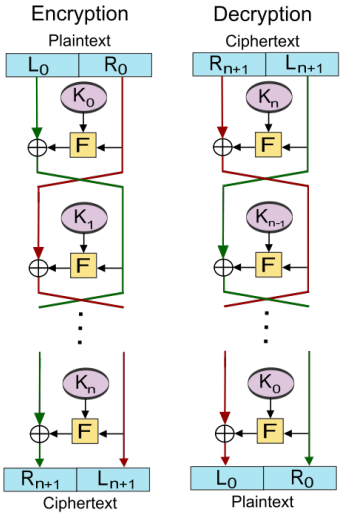
Berikut adalah langkah-langkah untuk melakukan dekripsi dengan jaringan Feistel.

1. Bagi cipherteks menjadi dua blok sama panjang (R_{n+1}, L_{n+1})

2. Untuk setiap putaran ke-$i = n, n-1, \ldots, 0$, hitung R_i dan L_i dengan cara sebagai berikut.

$$R_i = L_{i+1} \quad (3)$$
$$L_i = R_{i+1} \oplus F(L_{i+1}, K_i) \quad (4)$$

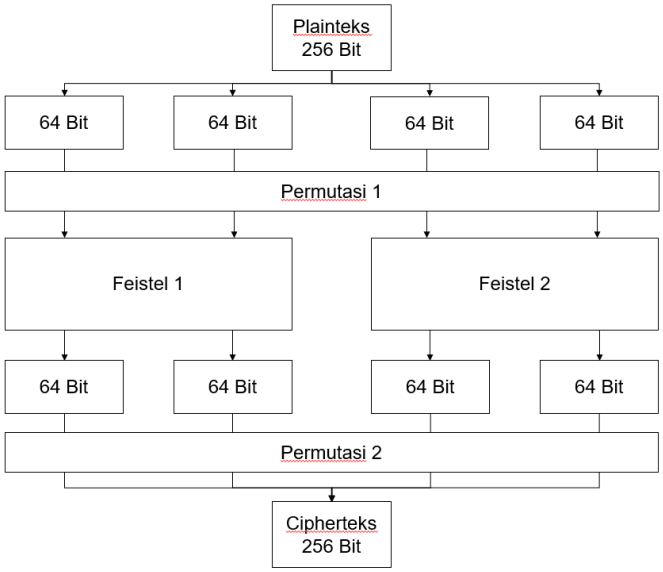
3. Plainteks merupakan gabungan dua blok (L_0, R_0).



Gambar 6 Ilustrasi Jaringan Feistel

III. RANCANGAN BLOCK CIPHER YEA

Rancangan *block cipher* YEA dapat dilihat pada Gambar 7. Algoritma *block cipher* ini merupakan modifikasi dari *block cipher* yang memanfaatkan jaringan Feistel. *Block cipher* YEA beroperasi pada blok bit dengan panjang 256 bit. Pengguna dapat memasukkan kunci dengan panjang bebas. Secara garis besar, berikut adalah tahapan dari algoritma *block cipher* YEA.



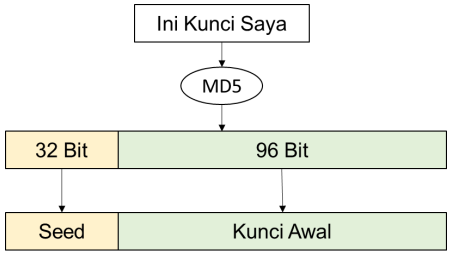
Gambar 7 Rancangan Algoritma *Block Cipher* YEA

1. Membagi blok plainteks 256-bit menjadi empat buah blok 64-bit, kemudian melakukan permutasi terhadap masing-masing blok 64 bit.
2. Dua blok 64-bit pertama menjadi masukan jaringan Feistel pertama, sedangkan dua blok sisanya menjadi masukan jaringan Feistel kedua.
3. Melakukan permutasi terhadap masing-masing blok 64-bit keluaran jaringan Feistel, dan menggabungkan seluruh blok menjadi blok cipherteks 256-bit.

Pada *block cipher* YEA, terdapat empat buah hal unik yang kami ajukan, yaitu konstruksi kunci awal dan *seed*, permutasi dengan rotasi matriks, pembangkitan upakunci, dan fungsi putaran jaringan Feistel.

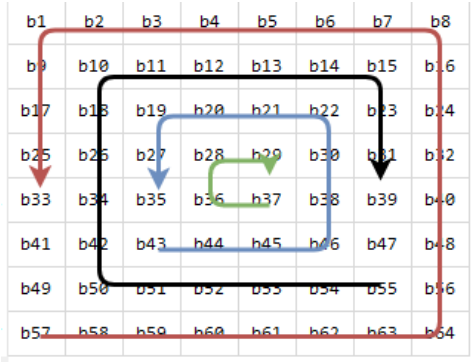
A. *Pembangkitan Kunci Awal dan Seed*

Pengguna awalnya memberikan kunci dengan panjang bebas. Kunci tersebut kemudian ditransformasi oleh fungsi *hash* MD5 menjadi sebuah teks 128 bit. *Seed* yang digunakan untuk membangkitkan jumlah rotasi pada permutasi rotasi matriks dan pembangkitan upakunci merupakan 32-bit awal teks. Kunci awal yang akan digunakan untuk pembangkitan upakunci setiap putaran adalah 96-bit terakhir. Ilustrasi pembangkitan kunci awal dan *seed* dapat dilihat pada Gambar 8.



Gambar 8 Ilustrasi Pembangkitan Kunci Awal dan *Seed*

B. *Permutasi Rotasi Matriks*



Gambar 9 Ilustrasi Permutasi Matriks

Permutasi rotasi matriks beroperasi pada blok 64-bit. Awalnya, blok yang akan dipermutasi ditransformasi menjadi matriks 8x8, di mana setiap anggota matriks merupakan bit dari blok. Kemudian, berdasarkan *seed* dari kunci masukan pengguna, akan dibangun empat buah bilangan *pseudorandom* yang menyatakan jumlah.

Gambar 9 memperlihatkan ilustrasi permutasi matriks, di mana setiap garis menyatakan arah rotasi elemen matriks pada "kulit" tertentu. Setiap garis juga memiliki jumlah rotasi yang berbeda sesuai dengan angka yang dibangkitkan dari *seed*. Setelah permutasi dilakukan, matriks hasil permutasi kemudian dibentuk kembali menjadi blok 64-bit. Untuk proses dekripsi,

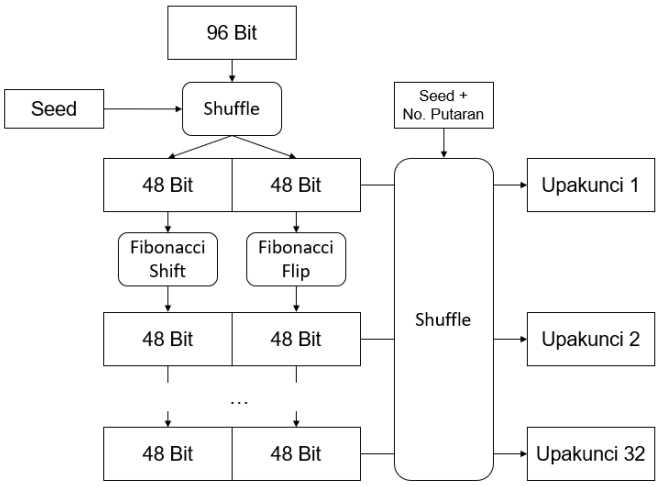
arah permutasi rotasi matriks diubah menjadi arah kebalikan proses enkripsi.

C. *Pembangkitan Upakunci*

Karena terdapat dua buah jaringan Feistel, dan setiap jaringan Feistel membutuhkan 16 putaran, maka dibutuhkan 32 buah upakunci untuk setiap putaran jaringan Feistel. Upakunci ke-1 hingga ke-16 akan digunakan untuk jaringan Feistel pertama, sedangkan upakunci sisanya digunakan untuk jaringan Feistel kedua. Pembangunan upakunci memanfaatkan kunci awal sepanjang 96-bit yang dihasilkan tahap pembangkitan kunci awal. Setiap upakunci akan memiliki panjang 96-bit juga.

Pembangkitan upakunci pada dasarnya mirip dengan pembangkitan upakunci pada algoritma DES, namun terdapat perbedaan pada tahap permutasi dan pergeseran. Ilustrasi pembangkitan upakunci dapat dilihat pada Gambar 10. Berikut adalah langkah-langkahnya.

1. Mengacak bit-bit pada blok 96-bit dengan fungsi acak dengan *seed* yang dibangkitkan berdasarkan kunci.
2. Memecah blok 96-bit menjadi dua buah blok 48-bit sama panjang.
3. Untuk suatu putaran, blok kiri dan blok kanan dikonkatenasi, kemudian diacak lagi dengan fungsi acak dengan *seed* bernilai (*seed* + nomor putaran). Hasilnya adalah sebuah upakunci.
4. Sebelum ke putaran selanjutnya, sistem melakukan operasi Fibonacci *shift* terhadap blok kiri, dan Fibonacci *flip* terhadap blok kanan.
5. Ulangi tahapan hingga putaran ke-32.

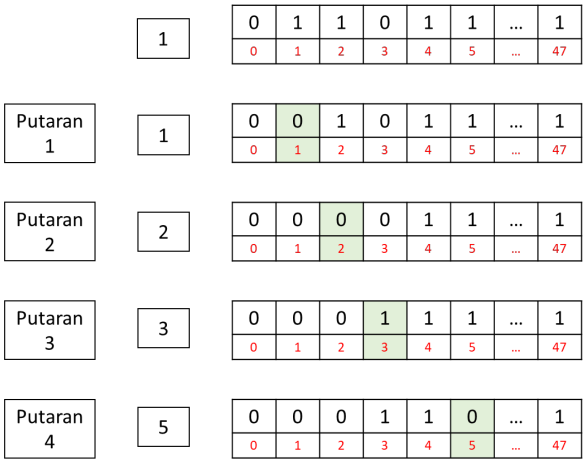


Gambar 10 Ilustrasi Pembangkitan Upakunci

Misalkan terdapat bilangan x sebagai berikut.

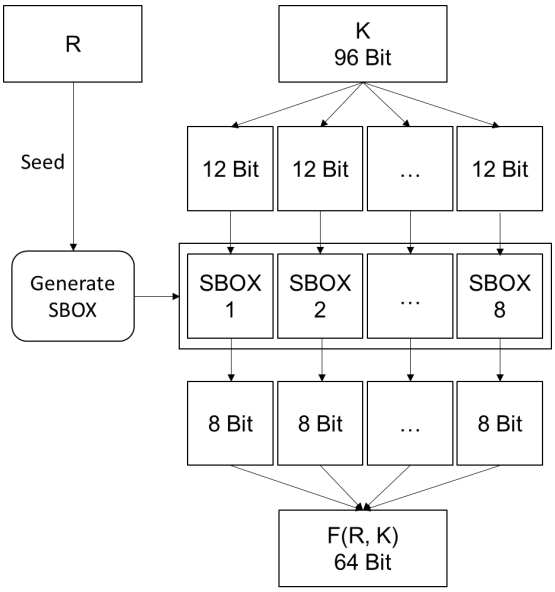
$$x = fibonacci(\mathrm{i})\ mod\ 48 \qquad (5)$$

Di mana i adalah nilai putaran ke-i. Operasi Fibonacci *flip* adalah proses mengganti nilai bit pada indeks ke-x dengan bit kebalikannya, sedangkan operasi Fibonacci *shift* adalah proses menggeser secara sirkular ke kiri sebanyak x kali. Ilustrasi proses Fibonacci *flip* dapat dilihat pada Gambar 11.



Gambar 11 Ilustrasi Fibonacci *Flip*

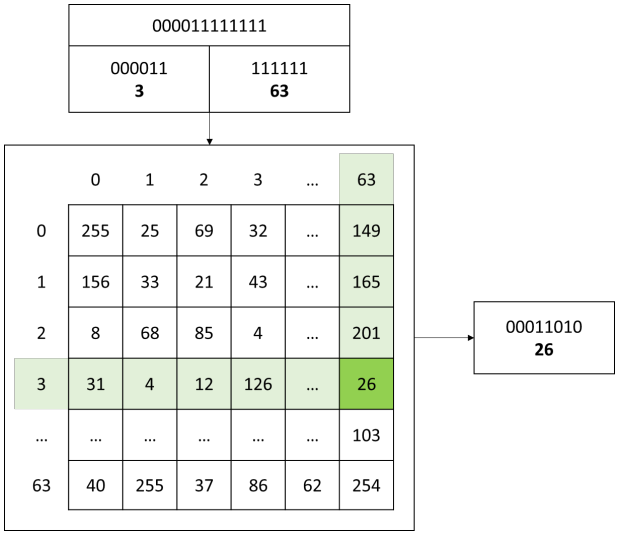
D. *Fungsi Putaran*



Gambar 12 Fungsi Putaran Jaringan Feistel *Block Cipher* YEA

Gambar 12 memperlihatkan fungsi putaran yang digunakan pada setiap jaringan Feistel. Awalnya, akan dibangun 8 buah matriks S-Box berdasarkan secara acak semu. *Seed* yang digunakan adalah hasil seluruh penjumlahan seluruh bit pada blok R. Dengan demikian, nilai 8 buah matriks S-Box pada setiap putaran pasti berbeda. Ukuran matriks S-Box adalah 64 x 64, di mana elemen matriks hanya dapat memiliki nilai 0-255. Setiap elemen muncul berulang 16 kali pada matriks.

Setelah S-Box dibangun, upakunci dengan panjang 96-bit dipecah menjadi 8 buah blok 12-bit. Masing-masing blok 12-bit kemudian akan disubstitusi menjadi 8-bit menggunakan satu matriks S-Box. Caranya adalah dengan memecah blok 12-bit menjadi blok kiri dan blok kanan masing-masing dengan panjang 6-bit. Nilai desimal blok kiri merupakan nomor baris, sedangkan nilai desimal blok kanan adalah nomor kolom. Hasil substitusi blok adalah elemen matriks sesuai nomor baris dan kolom dengan panjang 8 bit. Ilutrasi proses substitusi dapat dilihat pada Gambar 13.



Gambar 13 Ilustrasi Proses Substitusi pada Fungsi Putaran

Hasil masing-masing substitusi kemudian digabungkan membentuk sebuah blok 64-bit. Blok ini merupakan keluaran fungsi putaran.

IV. Eksperiman dan Pembahasan Hasil

Block cipher Yoncé Encryption Algorithm (YEA) diimplementasikan dengan bahasa Python. Untuk melakukan eksperimen terhadap YEA, dilakukan pengujian dengan 5 mode operasi *block cipher*; Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), dan Counter (CTR). Data yang digunakan merupakan teks mars informatika dengan panjang teks sebesar 240 karakter yang akan dibaca dari *file mars.txt*. Kunci yang digunakan adalah '**INFORMATIKA2018**'.

```
Informatika berjiwa satria
Tidak pernah mengenal keluh kesah
Tugas yang melimpah bukanlah rintangan
Lautan ujian sudahlah biasa

Hidup! Hidup! Hidup Informatika!
ITB almamater tercinta
Hidup! Hidup! Hidup Informatika!
ITB almamater tercinta
```

A. Hasil Pengujian dengan ECB

Mode operasi ECB melakukan enkripsi pada plainteks secara independen, di mana setiap blok plainteks akan dienkripsi secara terpisah. Untuk mempermudah pembacaan, representasi heksadesimal digunakan untuk menunjukkan hasil enkripsi. Berikut adalah hasilnya.

```
41 E4 23 3E 29 18 E0 E4 F9 19 94 D1 52 EA
D5 53 70 A4 C9 7B FA 26 CD 94 98 17 82 8A
56 8C 53 46 BA C8 D1 F8 7E 84 7C 30 F5 67
7F 7B F9 70 AF ED 48 B7 17 08 E1 55 E1 61
3F 5F 23 45 50 10 DC 80 AD DD 3C 60 C9 16
F1 EE C6 1C B9 B6 DB B9 BA 09 A7 3C 15 F4
3D 50 B3 FB CF 1D D2 9A C1 D4 0B 85 E0 CA
5A A8 13 EF A5 6E DC E1 27 D0 82 C1 E3 CA
D6 14 45 7B F2 DF BB D5 32 46 C7 35 A0 52
CF 94 85 BA 2B F8 91 24 14 BF 55 B2 26 E2
B5 B2 93 1B 5D C6 DF A5 B2 02 5C 68 AE 2A
89 A2 BD 32 69 95 AE 4E 4D 4F 52 A1 09 2B
02 96 C2 CE 96 8E 63 13 2D E4 6C D3 78 DB
EE 00 D7 6D 5B 43 C3 7B AD 66 13 6D A1 C6
49 95 84 91 1C CC DB E2 83 6B B0 90 C5 1D
B2 90 24 1B D9 F8 FD F9 64 6C BB A9 3D 85
7F B2 D4 24 52 3F 37 6F D0 61 5A 6C 9D E6
59 40 41 65 B0 B1 24 65 85 AC F1 76 95 22
59 DD 87 DB
```

B. Hasil Pengujian dengan CBC

Mode operasi CBC melakukan enkripsi pada plainteks dengan membuat ketergantungan antar blok plainteks yang akan dienkripsi. Blok yang dienkripsi akan dipengaruhi oleh hasil enkripsi dari blok sebelumnya. Karena tidak ada blok sebelumnya pada blok plainteks pertama, maka digunakan IV (*initialization vector*) sebagai blok yang mempengaruhi blok pertama plainteks. Pada pengujian ini, IV yang digunakan adalah blok dengan nilai bit 1 seluruhnya (2^{256}-1). Berikut adalah hasilnya.

```
25 9F 7C 7F F2 07 49 6F CD DD 66 F2 81 9E
4D 5C D0 64 C1 62 BE E8 99 FA B9 1F 0F 23
17 ED DE ED 67 95 A5 F0 A2 DE 02 BC 5A D0
D5 02 CB 01 F7 0B 9A FA B3 B0 7C E6 C9 C4
33 7E 25 EA 49 82 DD 0B 7D AD 56 BD 41 30
F6 56 A7 51 E7 0B E3 18 EA F9 F2 DC 11 1B
0E CE 3A 0D 72 80 60 41 0D 4A 08 49 DF 3B
5B 0D 4E 78 32 AE 47 C1 2C ED E3 E3 42 36
FE 5E DA 71 5C 56 4E 36 23 D2 B6 DA 42 90
59 5C 7C 62 EA 4A 6C CA F2 9A 58 7A 86 85
23 36 FF 25 A1 07 C4 C6 BA 7A 04 B6 4C 15
03 82 59 DB 4D D8 6A F1 22 13 A5 2A 66 83
2A 22 78 61 2B 02 80 D5 6C 0E 8E 66 4E EA
05 12 97 C2 26 DE 0C EE 8F 4F 92 F0 3D 39
A4 02 71 7D D1 7D 6B 0F 97 4F B3 CB AB CC
8D DA 00 68 73 8F 13 C5 B0 BB A9 F3 27 6D
C5 A4 95 8C 31 7C 50 A4 1B B6 1B A6 36 3B
F4 91 37 2D BE B9 30 91 41 F7 AE DE D6 1D
66 1F A4 01
```

C. Hasil Pengujian dengan CFB

Mode operasi CFB menggunakan blok antrian yang akan dienkripsi untuk diproses dengan blok enkripsi. Hasil enkripsi dari blok antrian tersebut lalu diambil sepanjang blok enkripsi dari *most significant bits* dilakukan operasi XOR dengan blok enkripsi. Blok antrian lalu digeser ke kiri sebanyak jumlah bit pada blok enkripsi yang lalu disisipkan hasil proses sebagai *least significant bits*. Berikut adalah hasil enkripsi dengan panjang antrian 256-bit serta IV dengan nilai (2^{256}-1).

```
A8 C9 24 61 17 4C 62 D3 86 90 AB 76 52 8D
7A D9 98 4F 8D D5 72 21 5C 1B 05 6B 38 89
A3 D5 31 2A 36 9D F5 0A 02 59 74 73 1F 96
23 21 1C 97 31 A0 A4 ED A1 AC DF 31 17 63
0A B0 0D 61 AE 1A 0E 26 E9 23 69 01 25 25
```

```
CA 69 21 90 E3 C6 42 3F CF 8D 97 2F C4 1A
B3 52 90 D2 0A AA 4F 12 51 31 B6 CB C9 02
2C 95 BB 99 6E 8C 1C D0 30 A9 09 44 52 71
D4 F4 A9 56 1A 10 80 CA 41 A5 54 7D C9 01
9E 97 0A 52 74 63 8C C4 2B 4B 9D 3E DD 9D
76 1B 0B 95 42 49 29 23 33 A2 70 45 9C C4
91 02 37 B0 45 23 9A 70 94 52 E0 7B 5A 46
92 AE 1A 67 25 1B ED AB 4E F4 6F 72 52 CB
45 82 98 6C A9 66 23 A6 13 92 B7 7F 5F 94
AC 73 52 E5 64 8C BB BF BD CC CA F0 E2 8F
55 F9 25 A9 43 0F 66 BC F9 9C C2 E0 6D 5F
5A 13 03 A6 72 64 6D 4F 70 86 50 FA 8D E9
E1 A4 E0 13 11 4C 17 DE AA EF B0 E5 A2 A6
0F D4 A3 BF
```

D. *Hasil Pengujian dengan OFB*

Pada mode operasi OFB, enkripsi blok plainteks dilakukan hampir sama dengan CFB. Bedanya adalah, pada OFB saat blok antrian digeser, bit yang disisipkan adalah bit yang digunakan untuk melakukan operasi XOR dengan blok enkripsi. Berikut adalah hasil enkripsi dengan panjang antrian 256-bit dan IV dengan nilai (2^{256}-1).

```
A8 C9 24 61 17 4C 62 D3 86 90 AB 76 52 8D
7A D9 98 4F 8D D5 72 21 5C 1B 05 6B 38 89
A3 D5 31 2A C3 84 29 3D E0 B4 13 98 0D A8
46 AA D9 5C 68 AF 61 93 62 1E 0F 58 83 E6
02 CC 88 2D 65 05 76 7C 2C 66 61 33 C3 AB
F3 E2 33 83 B6 90 EE 6F AC 5B 30 84 08 41
32 36 CE 9C E5 AC AC EF 5F 9B AB FE F8 67
80 08 91 A3 E2 A8 DE 01 C8 47 35 AF D7 2B
6C B6 25 2B A7 47 59 33 46 42 F4 77 0E 68
DA B0 BC BE EE 50 DA 90 57 B3 E7 FA 26 F4
DB 74 F5 4E 50 15 A4 31 ED 46 3B 56 DD 4D
B4 15 AB 96 9F AE B2 26 7F DF BF B2 FD 2D
7A 02 34 D5 04 B4 8D DE 3D 90 49 66 B3 EF
F3 5A FD B9 2E 95 37 79 35 A7 68 15 B1 F4
6B 11 80 2C D9 9E 40 5D 65 BD CE F8 97 4E
A4 6D B7 EB C8 68 DD 68 C1 DA 18 71 1A F3
8A B0 75 5B BF EC 67 25 90 47 18 01 96 51
0B 2A 31 F2 11 8D 3E C1 E8 E9 0C 2C 98 BE
3D 83 C7 D5
```

E. *Hasil Pengujian dengan CTR*

Pada metode CTR, terdapat *counter* yang merupakan sebuah nilai berupa blok *bit* yang ukurannya sama dengan ukuran blok plainteks, yaitu 256. Nilai blok *counter* pertama yang kami gunakan adalah jumlah bit-bit dari kunci awal. Nilai blok *counter* kami tambahkan dengan 1 untuk enkripsi blok plainteks selanjutnya. Berikut adalah hasil enkripsinya.

```
63 3A E0 92 8F 35 14 7A 43 0B FF 8F E3 CF
08 08 28 10 C3 31 D7 06 FB 10 98 15 BF 56
32 BB 62 6B 0A 24 E3 8F 93 39 1D 2E 47 05
F0 C8 E4 C4 1B 0E 61 0E D5 DD 51 8D A5 33
94 05 F4 4A 53 09 F2 2C 4B 27 A6 84 9C 36
12 2E 47 05 F2 C6 EC DA 1B 0A 61 05 D5 7A
45 8B E1 39 99 54 C7 6B 37 29 E6 24 4D 35
```

```
E8 F7 B1 39 00 7A 4B 0E BE DA EB C3 1B 0C
61 16 C7 D5 C5 8F EB 3B 99 56 F7 4B 3A 2C
E2 43 20 1C EF 99 88 28 54 2E 62 09 FA DA
F1 8B 5A 2A 28 03 D7 61 84 AE E1 3C 9E 06
D8 63 2F 36 E8 29 0B 5E CF A9 BF 78 14 62
47 01 F3 CE F5 CF 08 42 35 00 C2 D2 4D 0B
71 A5 FB 3E FC 46 2C AD A6 3F 62 3D E2 88
8D 79 55 46 43 04 EB DF A1 E3 14 04 2E 15
CD 70 50 0C 66 A5 D0 7E FC 56 1B FD E6 72
47 35 EB 9C 89 3D 07 2E 5E 05 EC CC E8 C4
0E 03 41 65 B2 B1 A4 67 07 C6 F1 76 95 22
5B DF 83 1D
```

V. ANALISIS KEAMANAN

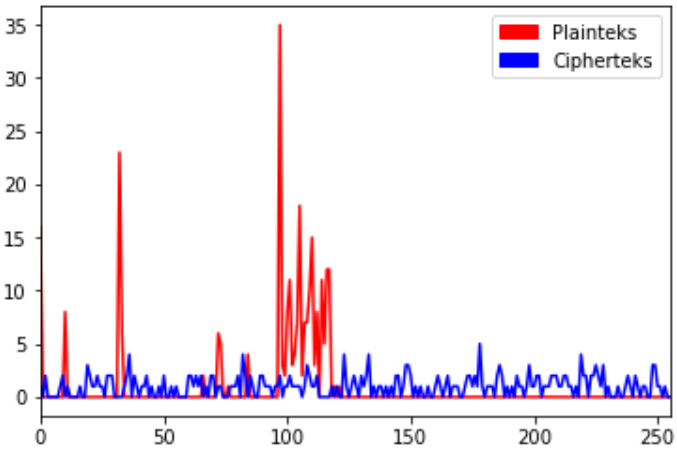
A. *Serangan Brute Force*

Penyerang dapat melakukan serangan *brute force* percobaan kunci untuk mendapatkan hasil dekripsi yang tepat. Secara teoretis, karena pengguna dapat memasukkan kunci dengan panjang berapapun, maka jumlah kemungkinan kunci yang dapat ditebak tidak terbatas. Kunci yang berbeda pun memiliki kemungkinan besar untuk menghasilkan keluaran *hash* MD5 yang sama.

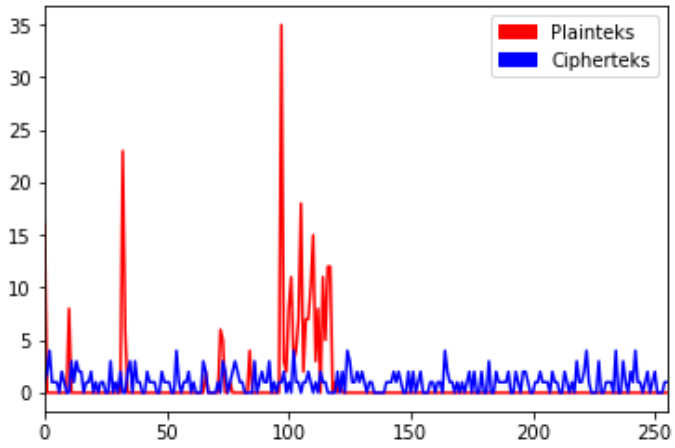
Jika penyerang mengetahui bahwa fungsi MD5 digunakan untuk membangkitkan kunci, maka untuk menebak kunci yang tepat, penyerang harus menggunakan *rainbow table* yang berisi 2^{128} atau 3×10^{38} keluaran fungsi *hash* MD5. Pembangunan tabel dengan ukuran tersebut mahal dan waktu untuk melakukan *brute force* sangat lama.

B. *Analisis Frekuensi*

Dalam kriptografi, serangan umum yang dilakukan salah satunya adalah dengan analisis frekuensi [6]. Misalkan untuk Bahasa Inggris, analisis frekuensi dilakukan untuk menentukan huruf E, T, A, dan O dengan melihat frekuensi *byte* yang sering muncul pada cipherteks, karena keempat huruf tersebut merupakan huruf yang sering muncul dan Bahasa Inggris. Dengan informasi huruf yang sering muncul, penyerang dapat memecahkan cipherteks dengan informasi yang telah disediakan. Gambar 14 dan Gambar 15 memperlihatkan analisis frekuensi dengan mode operasi ECB dan CBC.



Gambar 14 Analisis Frekuens Mode Operasi ECB



Gambar 15 Analisis Frekuensi Mode Operasi CBC

Berdasarkan analisis tersebut, dapat disimpulkan bahwa teknik analisis frekuensi akan sulit untuk diterapkan pada algoritma YEA. Hal ini dapat dilihat dari distribusi frekuensi *byte* pada cipherteks yang merata bahkan pada mode operasi ECB. Terbukti bahwa YEA memenuhi prinsip *diffusion* dengan baik.

C. Sedikit Perbedaan Cipherteks saat Dekripsi

Saat akan melakukan dekripsi dengan mengubah karakter pertama menjadi huruf 'a' pada cipherteks, perubahaan yang terjadi tidak begitu signifikan pada plainteks yang dihasilkan untuk seluruh mode. Berikut adalah hasil dekripsi dengan metode CFB. Dapat dilihat, baris yang berubah hanya baris pertama dan kedua saja.

```
□nformatika berjiwa satria
Tidak§q$woah =do&afal keluh kesah
Tugas yang melimpah bukanlah rintangan
Lautan ujian sudahlah biasa

Hidup! Hidup! Hidup Informatika!
ITB almamater tercinta
Hidup! Hidup! Hidup Informatika!
ITB almamater tercinta
```

Berikut adalah hasil dekripsi dengan metode CTR. Dapat dilihat, dari hasil yang diberikan, hanya huruf pertama saja yang berubah. Dengan demikian, sedikit kerusakan pada cipherteks tidak akan merusak plainteks dengan buruk.

```
Knformatika berjiwa satria
Tidak pernah mengenal keluh kesah
Tugas yang melimpah bukanlah rintangan
Lautan ujian sudahlah biasa

Hidup! Hidup! Hidup Informatika!
ITB almamater tercinta
Hidup! Hidup! Hidup Informatika!
ITB almamater tercinta
```

D. Sedikit Perbedaan Kunci

Apabila kunci diubah dari '**INFORMATIKA2018**' menjadi '**INFORMATIKA2017**' pada saat melakukan dekripsi, hasil yang dari dekripsi sangat berbeda jauh dari plainteks sebelumnya. Hal ini berlaku untuk seluruh metode, baik yang terdapat *chaining* maupun tidak. Terbukti bahwa YEA memenuhi prinsip *confusion* dengan baik. Berikut adalah hasil dekripsi degan kunci yang diubah pada metode CBC.

```
¥#ŠF&fù+aä□´LñMŸPHØKõìNµ"s¥7à˜g2ÕÛïè.×ª–
)Œ=ÃpC
™□%}î|î4ãÈ"|c?OšSn™Ò□[»-
â*È‹õ<7Ÿ;œ¯>"C2×¹□"„²˜yŠæ°çYÁ□™vŠW%§Í?ð)
•¤E□æÕ›ÎUW4/Nª'L□&¦úZ□ñ0Â¥
```

VI. Simpulan dan Saran

Berdasarkan hasil eksperimen dan analisis keamanan, terlihat bahwa *block cipher* YEA menerapkan prinsip *confusion* dan *diffusion* dengan baik. Algorima ini memperlihatkan hasil yang baik untuk berbagai mode operasi *block cipher*. Pengembangan selanjutnya yang mungkin adalah untuk melakukan enkripsi dan dekripsi secara paralel. Misalnya. Pemrosesan pada dua buah jaringan Feistel dapat dilakukan secara paralel. Proses permutasi untuk empat blok 64-bit juga dapat dilakukan secara paralel. Selain itu, dibutuhkan pembangunan S-Box yang menjamin properti *confusion* secara otomatis.

Ucapan Terima Kasih

Penulis memanjatkan rasa syukur kepada Tuhan YME karena atas kuasa-Nya penulis dapat menyelesaikan makalah ini. Penulis mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir selaku dosen mata kuliah Kriptografi karena atas bimbingannya penulis mendapatkan bekal ilmu yang cukup untuk menulis makalah ini. Terakhir, penulis juga mengucapkan terima kasih kepada pihak-pihak yang secara langsung maupun tidak langsung telah membantu menyelesaikan makalah ini.

Referensi

[1] Munir, Rinaldi. 2015. Slide Kuliah IF4020 Kriptografi: Pengantar Kriptografi.

[2] Dworking, Morris. 2001. Recommendation for Block Cipher Modes of Operation. National Institute of Standards and Technology Special Publication.

[3] William F. Ehrsam, Carl H. W. Meyer, John L. Smith, Walter L. Tuchman. 1976. "Message verification and transmission error detection by block chaining". US Patent 4074066.

[4] A. Menezes, P. van Oorschot, dan S. Vanstone. 1996. Handbook of Applied Cryptography. CRC Press.

[5] Munir, Rinaldi. 2015. Slide Kuliah IF4020 Kriptografi: Algoritma Kriptografi Modern.

[6] Stallings, William. 2014. Cryptography and Network Security (6th ed.). Upper Saddle River, N.J.: Prentic Hall.

Pernyataan

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

Bandung, 14 Maret 2018

Garmastewira (13514068)

Muhammad Gumilang (13514092)