Composite Block Cipher

Aufar Gilbran (13513015)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Jl. Ganesha 10 Bandung, Indonesia
aufargilbran@gmail.com

Kevin Yauris (13513015)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Jl. Ganesha 10 Bandung, Indonesia
kyauris@gmail.com

***Abstract*— Dalam era modern ini kebutuhan akan suatu metode untuk dapat mengirimkan pesan dengan aman semakin dibutuhkan. Salah satu metode untuk dapat melakukan komunikasi dan mengirimkan pesan dengan aman adalah dengan menggunakan algoritma enkripsi untuk mengenkripsi pesan yang ada sehingga walaupun dapat dicuri tidak dapat dimengerti oleh pengambil pesan. Hal ini memotivasi kami untuk membuat suatu *cipher* yang menggabungkan beberapa teori kriptografi dan algoritma enkripsi yang sudah pernah ditemukan. Teori yang kami pakai antara lain adalah properti *diffusion* dan *confusion.* Sedangkan algoritma yang kamu pakai diantaranya adalah algoritma *hash MD5. E*nkripsi yang dilakukan juga memanfaatkan *feistel network* dengan *round key* berupa hasil transformasi dan subtitusi serta digabungkan dengan algoritma MD5. Makalah ini akan membahas mengenai algoritma yang kami namakan *Composite Block Cipher* ini.**

Keywords—kriptografi, cipher, block cipher, enkripsi

I. Introduction

Di era modern ini penggunaan internet semakin luas digunakan, semakin hari orang semakin mudah melakukan komunikasi maupun mendapatkan informasi yang diinginkan. Suatu komunikasi merupakan proses transmisi data atau pesan dari satu pihak ke pihak lainnya. Dalam prosesnya, bisa jadi pesan atau data tersebut melewati beberapa komputer sebelum akhirnya sampai kepada tujuan pesan tersebut. Hal ini tidak menjadi masalah apabila data atau pesan yang dikirim tidak mengandung suatu informasi rahasia atau berkaitan dengan privasi seseorang, namun akan sangat berbahaya apabila sebaliknya. Oleh karena itu untuk dapat melindungi pesan atau data yang dikirimkan itu dibutuhkan suatu cara untuk membuat pesan tersebut hanya dapat diakses atau dimengerti oleh penerima pesan tersebut. Salah satu cara yang dapat digunakan adalah dengan menggunakan cipher, yaitu algoritma untuk melakukan enkripsi suatu pesan, dari plainteks ke cipherteks. Pesan yang ingin dikirim akan dienkripsi dengan cipher dan kunci tertentu, kemudian pesan dikirimkan lewat saluran publik sedangkan kunci dikirimkan secara rahasia. Hal ini akan memastikan pesan aman, karena walaupun mungkin dapat dicuri datanya tetapi pesan atau data tidak dapat dimengerti isinya karena telah terenkripsi. Pada tulisan ini akan dipaparkan sebuah cipher yang dibuat dengan menggabungkan beberapa prinsip serta algoritma enkripsi yang telah ada.

Nama cipher yang akan dibahas ini adalah *Composite Block*. Struktur utama cipher ini menggunakan struktur pada jaringan *Feistel*, dimana *round key* yang dipakai dibuat dengan menggunakan algoritma MD5 dan dikombinasikan dengan transformasi serta subtitusi terhadao *plainteks*. Dilakukan juga subtitusi serta transformasi agar algoritma enkripsi ini memiliki properti *diffusion and confusion.*

II. Landasan Teori

A. *Properti Diffussion* dan *Confusion*

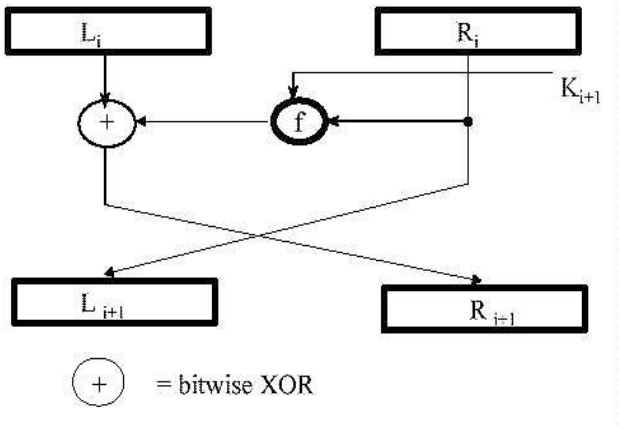
Syarat algoritma kriptografi modern yang aman harus memenuhi properti *diffusion* dan *confusion* yang dikemukakan oleh Claude Shannon pada tahun 1945[2]. Properti *diffusion* dan *confusion* mempersulit analisa statistik dan metode lain yang umum digunakan untuk melakukan *cryptanalysis* pada cipherteks.

Properti *diffusion* berarti menyebarkan pengaruh satu bit pada plainteks dan kunci ke sebanyak mungkin bit cipherteks. Perubahan satu bit pada plainteks dapat mengubah banyak bit pada cipherteks. Properti *confusion* memasikan hubungan antara cipherteks dengan kunci simetrik harus saling terkait sebanyak mungkin. Kedua properti ini dapat didapatkan dengan cara melakukan substitute dan transformasi secara berurutan.

B. *Feistel Network*

Jaringan Feistel ditemukan oleh Horst Feistel 1970. Jaringan Feistel bekerja pada dengan membagi plainteks menjadi blok-blok yang berukuran N bit. Model jaringan Feistel adalah sebagai berikut:

1. Bagi blok yang panjangnya N bit menjadi dua bagian sama panjang, mari kita definisikan blok kiri sebagai L_0 dan blok kanan sebagai R_0.
2. Definisikan cipher blok berulang dimana hasil dari putaran ke-i_{+1} ditentukan dari hasil putaran ke-i . Secara formal, cipher blok (L_{i+1}, R_{i+1}) adalah:
3. Hasil enkripsi dengan menggunakan jaringan *Feistel* adalah cipher blok (L_n, R_n).



Gambar 1. Jaringan Feistel

Round function F pada setiap perulangan ditentukan oleh perancang algoritma. *F* harus memiliki mekanisme transformasi dan substitusi dikarenakan properti *diffusion* dan *confusion* yang menentukan kekuatan algoritma didapatkan dengan mekanisme tersebut.

Jaringan *Feistel* dilakukan dengan beberapa putaran (umumnya 16 putaran agar tidak terlalu lambat). Putaran tersebut bersama dengan mekanisme transformasi dan substitusi pada *round function* akan menghasilkan properti *diffusion* dan *confusion* sehingga memperkuat kekuatan algoritma enkripsi.

C. *Properti Diffussion dan Confusion*

MD5 adalah salah satu dari serangkaian algoritma message digest yang didesain oleh Profesor Ronald Rivest dari MIT (Rivest, 1994). Saat kerja analitik menunjukkan bahwa pendahulu MD5 mulai tidak aman. MD5 kemudian didesain pada tahun 1991 sebagai pengganti dari MD4 (kelemahan MD4 ditemukan oleh Hans Dobbertin).

Algoritma MD5 yang utama beroperasi pada kondisi 128-bit, dibagi menjadi empat word 32-bit, menunjukkan A, B, C dan D. Operasi tersebut diinisialisasi dijaga untuk tetap konstan. Algoritma utama kemudian beroperasi pada masing-masing blok pesan 512-bit, masing-masing blok melakukan pengubahan terhadap kondisi. Pemrosesan blok pesan terdiri atas empat tahap, batasan putaran; tiap putasan membuat 16 operasi serupa berdasar pada fungsi non-linear F, tambahan modular, dan rotasi ke kiri. Gambar satu mengilustrasikan satu operasi dalam putaran. Ada empat macam kemungkinan fungsi F, berbeda dari yang digunakan pada tiap-tiap putaran:

$$F(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$$
$$G(X,Y,Z) = (X \wedge Z) \vee (Y \wedge \neg Z)$$
$$H(X,Y,Z) = X \oplus Y \oplus Z$$
$$I(X,Y,Z) = Y \oplus (X \vee \neg Z)$$

$\oplus, \wedge, \vee, \neg$ menununjukan operasi logika XOR, AND, OR dan NOT.

D. *Hexdump*

Algoritma enkripsi *Composite Block* beroperasi pada blok berukuran *128*-bit. Blok berukuran *128*-bit akan memiliki *8* buah byte. Nilai dari sebuah byte berkisar dari *0* sampai dengan *255*. Tidak semua nilai pada kisaran tersebut dapat direpresentasikan sebagai karakter yang dapat dibaca/dikenali oleh manusia, sebagai contoh *NULL byte* atau byte bernilai *0* yang digunakan untuk penanda data kosong pada komputer.

Untuk itu, diperlukan teknik untuk memeriksa nilai suatu byte. Teknik yang umum digunakan untuk melakukan pemeriksaan adalah *hexdump*. Teknik ini mengubah representasi byte menjadi sebuah string yang memiliki dua buah karakter *0-9A-F*, yaitu nilai pada *hexadecimal* (basis-16). Maka, suatu byte yang bernilai *154* akan memiliki representasi *9A* pada *hexdump.*

Hexdump mempermudah pembacaan ketimbang menuliskan nilai pada byte pada *decimal* (basis-10) dikarenakan untuk merepresentasikan suatu byte hanya memerlukan *2* buah karakter saja dan tidak menggunakan banyak angka *0* untuk byte yang memiliki nilai kecil.

III. Rancangan Algoritma

Algoritma enkripsi *Composite Block* menerima masukan berupa pesan berukuran berapapun dan kunci enkripsi yang memiliki panjang 128-bit. Cipher *Composite Block* memiliki keunikan yaitu pada mekanisme substitusinya, tidak dilakukan dengan cara yang biasanya dilakukan. Proses pembangkitan kunci internal dan enkripsi adalah sebagai berikut.

A. *Pembangkitan Round Key*

Round key pada putaran ke-*i+1* didapatkan dengan mengaplikasikan MD5 kepada *round key* pada putaran ke-*i*. Hal ini dilakukan karena alasan sebagai berikut:

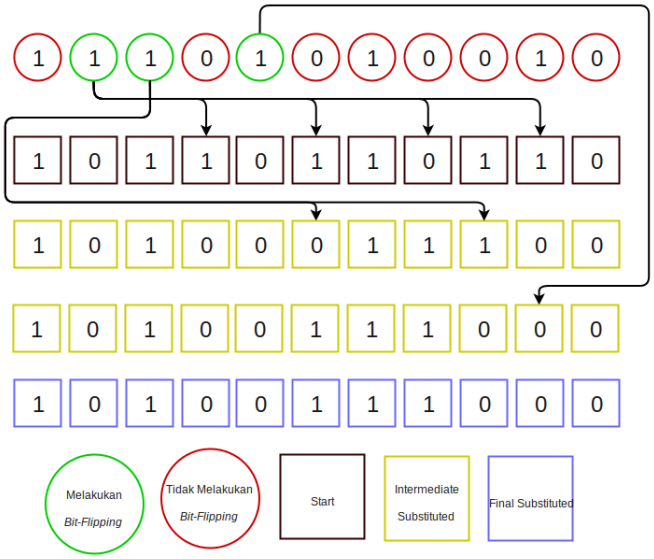
1. MD5 akan selalu menghasilkan *message digest* sepanjang 128-bit. Properti ini memenuhi persyaratan *round key* dari cipher *Composite Block*.
2. MD5 memiliki *collision* yang sangat kecil sehingga kunci yang berbeda hampir tidak mungkin menjadi *round key* yang sama.
3. MD5 sudah memiliki properti *diffusion* dan *confusion*, sehingga memperkuat properti *diffusion* dan *confusion* yang dimiliki cipher *Composite Block*.
4. MD5 memiliki kompleksitas yang linear terhadap *message* yang akan diproses, sehingga proses enkripsi cipher *Composite Block* efisien dan tidak terhambat pada pembentukan *round key*.

B. *Enkripsi Pesan*

Enkripsi pesan dilakukan dengan menggunakan jaringan *Feistel*. Cipher *Composite Block* mengimplementasikan *round function* yang berbeda. *Round function* yang digunakan memiliki dua mekanisme utama, yaitu:

1. Substitusi

   Substitusi pada *composite block* dilakukan dengan melakukan *bit-flipping* dari bit yang merupakan kelipatan bit prima yang bernilai *1* (lihat Gambar 2). *Bit-flipping* adalah proses mengubah nilai sebuah bit menjadi nilai komplemennya. Mekanisme substitusi ini mengambil struktur pembentukan bilangan prima dengan menggunakan metode *Sieve of Erathosthenes[3]*. Mekanisme substitusi ini digunakan untuk memastikan sebuah nilai pada kunci dapat mengakibatkan perubahan yang menyebar pada hasil enkripsinya. Selain itu setiap bit pada plainteks dapat berubah lebih dari satu kali apabila posisi bit tersebut memiliki banyak faktor. Dengan demikian, perubahan plainteks tidak satu-ke-satu terhadap kunci.



Gambar 2. Mekanisme substitusi pada round function

2. Transformasi

   Transformasi pada cipher *Composite Block* dilakukan dengan melakukan pertukaran byte pada blok yang posisinya ditunjuk oleh *round key*. Posisi bit ditentukan dengan melakukan modulo pada *round key* yang jumlah kedua indeks *round key* tersebut berjumlah *15*. Mekanisme transformasi bersama dengan mekanisme substitusi memunculkan propersi *diffusion*. Pada proses substitusi, perubahan hanya terjadi pada pesan dengan nilai indeksnya merupakan bilangan komposit dan apabila bit pada kunci yang indeksnya merupakan faktor prima bilangan tersebut bernilai *1*. Untuk itu, diperlukan pengacakan urutan sebelum substitusi pada *round* berikutnya agar indeks prima dan bilangan komposit yang indeks faktor primanya tidak menunjuk bit bernilai *1* pada *round key*.

IV. Eksperimen dan Pembahasan Hasil

Bab ini akan memperlihatkan dan memberikan penjelasan detil mengenai enkripsi *Composite* Block. Pada setiap bab akan diberikan hasil eksperimen yang dilakukan dengan cara memberikan *hexdump* plainteks pada Tabel 1 dan kunci pada Tabel 2 ke dalam algoritma enkripsi *Composite Block.*

```
Lorem ipsum dolor sit amet, consectetur
adipiscing elit. Integer maximus, lacus
eu porta molestie, quam orci aliquam
nisl, eu rutrum metus enim eu nisi.
Vivamus augue leo, pellentesque nec
turpis.

4c 6f 72 65 6d 20 69 70 73 75 6d 20 64
6f 6c 6f 72 20 73 69 74 20 61 6d 65 74
2c 20 63 6f 6e 73 65 63 74 65 74 75 72
20 61 64 69 70 69 73 63 69 6e 67 20 65
6c 69 74 2e 20 49 6e 74 65 67 65 72 20
6d 61 78 69 6d 75 73 2c 20 6c 61 63 75
73 20 65 75 20 70 6f 72 74 61 20 6d 6f
6c 65 73 74 69 65 2c 20 71 75 61 6d 20
6f 72 63 69 20 61 6c 69 71 75 61 6d 20
6e 69 73 6c 2c 20 65 75 20 72 75 74 72
75 6d 20 6d 65 74 75 73 20 65 6e 69 6d
20 65 75 20 6e 69 73 69 2e 20 56 69 76
61 6d 75 73 20 61 75 67 75 65 20 6c 65
6f 2c 20 70 65 6c 6c 65 6e 74 65 73 71
75 65 20 6e 65 63 20 74 75 72 70 69 73
2e 20 0a
```

Tabel 1. Plainteks beserta hexdump plainteks masukan

```
1234567890123456
```

Tabel 2. Kunci enkripsi

Plainteks yang digunakan untuk pengujian adalah teks Lorem Ipsum berbahasa Latin yang sering digunakan sebagai sampel uji. Pemilihan teks ini dilakukan karena mudahnya pembentukan sampel uji dan miripnya frekuensi kemunculan karakter pada teks Lorem Ipsum dengan kemunculan karakter pada teks bahasa Inggris.

Algoritma enkripsi *Composite Block* dapat berkerja dalam salah satu dari tiga modus operasi yang diimplementasikan, yaitu *Electronic Codebook (ECB)*, *Cipher Block Chaining (CBC)*, dan *Cipher Feedback(CFB)*.

Berikut adalah hasil dan penjelasan dalam masing-masing modus operasi:

A. *Electronic Codebook (ECB)*

ECB adalah modus operasi paling sederhana yang dapat dilakukan oleh jaringan *Feistel*. Berikut adalah *hexdump* cipherteks hasil enkripsi plainteks dalam modus ECB.

```
5a 7a af 0d 05 09 9e da 70 15 29 0d ee 3a
9c 35 72 34 ef 01 1b 09 94 db 66 14 68 0d
e9 3a 9e 29 61 67 ad 5d 11 40 8a 8c 62 04
2d 5d e3 26 93 33 2b 4e fe 59 05 48 8a 99
23 29 2a 59 ef 32 95 28 69 2d bd 51 06 5e
9d 96 2f 40 28 4c e9 20 83 7a 20 78 ff 54
06 47 9b cd 23 0d 2b 41 ef 26 84 33 6f 3e
```

```
f3 50 59 46 9a 8c 6c 12 27 44 aa 34 9c 33
78 39 f1 40 59 08 80 c3 6f 4c 64 48 ff 75
82 2f 64 61 e5 40 42 42 93 91 76 13 64 48
e4 3c 9d 7a 20 43 f9 50 04 58 87 98 23 36
2d 5b eb 38 85 29 20 6e e9 0f 09 46 d7 c5
66 0f 68 0d fa 30 9c 36 20 60 a1 4e 5f 43
9b d2 23 0e 21 4e aa 21 85 28 15 09 c3 66
2c 46 9b c5 03 60 44 2d 8a 55 f0 5a
```

Tabel 3. Enkripsi mode ECB

Cipherteks hasil enkripsi mode ECB tidak menunjukkan hubungan apapun dengan kunci enkripsi maupun plainteks yang dienkripsi. Namun, mode ECB memperlihatkan bahwa blok yang sama pada plainteks akan mengakibatkan hasil enkripsi yang sama, seperti pada contoh berikut:

```
Lorem ipsum dolor
```

```
4c 6f 72 65 6d 20 69 70 73 75 6d 20 64 6f
6c 6f 72 0a
```

Tabel 4. Blok subset plainteks

```
5a 7a af 0d 05 09 9e da 70 15 29 0d ee 3a
9c 35 17 60 b0 48 0c 46 9b c5 03 60 44 2d
8a 55 f0 5a
```

Tabel 5. Cipherteks yang sama

Perhatikan bahwa cipherteks pada Tabel 5 memiliki awal yang sama dengan sama dengan cipherteks pada Tabel 3. Perbedaan pada bagian belakang terjadi dikarenakan *padding* yang ditambah ketika panjang blok tidak memenuhi syarat panjang yang blok algoritma enkripsi *Composite Block*, yaitu *128*-bit.

Kesamaan yang terjadi disebabkan karena *round function* hanya bekerja pada sebuah blok dan tidak melibatkan blok lain ketika melakukan enkripsi pada sebuah blok. Kelemahan ini diatasi pada kedua mode operasi berikutnya.

B. *Cipher Block Chaining (CBC)*

Modus operasi *CBC* digunakan untuk mengatasi kelemahan yang dimiliki oleh modus operasi *ECB*. Berikut adalah *hexdump* cipherteks hasil enkripsi plainteks dalam modus *CBC*.

```
80 50 3a 53 b0 05 2a 99 d2 6e b9 aa f8 01
2d 07 a8 1e 7a 5f ae 05 20 98 c4 6f f8 aa
ff 01 2f 1b bb 4d 38 03 a4 4c 3e cf c0 7f
bd fa f5 1d 22 01 f1 64 6b 07 b0 44 3e da
81 52 ba fe f9 09 24 1a b3 07 28 0f b3 52
29 d5 8d 3b b8 eb ff 1b 32 48 fa 52 6a 0a
b3 4b 2f 8e 81 76 bb e6 f9 1d 35 01 b5 14
66 0e ec 4a 2e cf ce 69 b7 e3 bc 0f 2d 01
a2 13 64 1e ec 04 34 80 cd 37 f4 ef e9 4e
33 1d be 4b 70 1e f7 4e 27 d2 d4 68 f4 ef
f2 07 2c 48 fa 69 6c 0e b1 54 33 db 81 4d
bd fc fd 03 34 1b fa 44 7c 51 bc 4a 63 86
c4 74 f8 aa ec 0b 2d 04 fa 4a 34 10 ea 4f
2f 91 81 75 b1 e9 bc 1a 34 1a cf 23 56 38
99 4a 2f 86 a1 1b d4 8a 9c 6e 41 68
```

Tabel 6. Enkripsi mode CBC

Pada modus operasi CBC, enkripsi blok ke-*i+1* dilakukan tidak hanya dengan *round function*, namun setelah itu dilakukan XOR dengan *Initialization Vector*. *Initialization Vector* dibangkitkan secara acak ketika program mulai berjalan. Setelah itu, *Initialization Vector* diganti dengan hasil enkripsi blok ke-*i*. Dengan demikian, plainteks yang hanya berbeda sedikit, akan mempengaruhi hasil enkripsi blok berikutnya yang mungkin seharusnya sudah diketahui plainteks aslinya dari hasil enkripsinya.

Perbedaan dapat ditemukan ketika membandingkan hasil enkripsi masukkan pada Tabel 1 menjadi cipherteks pada Tabel 6 dengan hasil enkripsi masukkan pada Tabel 4 menjadi cipherteks pada Tabel 7 dengan *Initialization Vector* yang sama. Terlihat kesamaan pada bagian awal enkripsi tidak terjadi lagi pada modus operasi *CBC*.

```
43 e2 4a df 96 70 50 78 9c 92 8a e4 51 ed
2f e2 0e f8 55 9a 9f 3f 55 67 ef e7 e7 c4
35 82 43 8d
```

C. *Cipher Feedback (CFB)*

Modus operasi *CFB* mempermudah pengenkripsian apabila *round function* yang digunakan untuk enkripsi dan dekripsi berbeda sangat jauh. Berikut adalah *hexdump* cipherteks hasil enkripsi plainteks dalam modus CFC.

```
f3 25 57 73 d4 6a 46 f6 d2 6e b9 aa f8 01
2d 07 cd 6a 56 7f cd 6a 4e eb c4 6f f8 aa
ff 01 2f 1b da 29 51 73 cd 3f 5d a6 c0 7f
bd fa f5 1d 22 01 d1 2d 05 73 d5 23 5b a8
81 52 ba fe f9 09 24 1a 9f 27 44 6e d0 27
5a f5 8d 3b b8 eb ff 1b 32 48 da 3f 05 66
d6 38 5b e7 81 76 bb e6 f9 1d 35 01 da 66
05 67 cc 2b 42 a6 ce 69 b7 e3 bc 0f 2d 01
ce 3f 44 7b 99 24 46 f5 cd 37 f4 ef e9 4e
33 1d cb 38 50 7b 99 27 4a f2 d4 68 f4 ef
f2 07 2c 48 da 3f 05 78 d0 39 46 a8 81 4d
bd fc fd 03 34 1b 9f 2b 50 71 cc 2f 0f ea
c4 74 f8 aa ec 0b 2d 04 da 24 51 73 ca 3b
5a e3 81 75 b1 e9 bc 1a 34 1a cf 23 56 38
99 4a 2f 86 a1 1b d4 8a 9c 6e 41 68
```

Tabel 8. Enkripsi mode CFB

Modus operasi *CFB* memberikan keuntungan kemudahan pendekripsian ulang apabila cipherteks rusak, yakni apabila ada bagian cipherteks yang rusak, maka hanya bagian itu saja yang rusak. Meskipun secara singkat *CBC* memiliki sifat yang lebih *resilient* dibanding *CFB*, namun apabila diamati *CFB* jauh lebih praktis untuk digunakan.

V. ANALISIS

Untuk menentukan kekuatan enkripsi yang dilakukan oleh enkripsi *Composite Block*, kita perlu melakukan *cryptanalisis*. Hal ini dapat dilakukan dengan beberapa cara antara lain:

A. *Bruteforce Attack*

Salah satu teknik lain adalah dengan cara mencoba semua kemungkinan subtitusi huruf yang ada atau mencoba dengan menggunakan seluruh kemungkinan kunci yang dapat dibentuk. Jumlah kemungkinan kunci yang dapat dibuat adalah 26^{128}. Apabila diasumsikan bahwa dalam 1 detik dapat diciptakan 10^6 kunci maka waktu yang dibutuhkan untuk menciptakan seluruh kemungkinan kunci adalah 1.307942×10^{175} detik atau 1.513822×10^{170} hari. Waktu yang sedemikian lamanya bahkan hanya memberikan seluruh kemungkinan kunci umum, bukan kunci privat. Kunci privat harus dicari kembali dengan cara yang lain yaitu mengambil salah satu kunci umum yang ada dan menciptkan kunci khusus atau privat dengan metode tertentu. Oleh karena itu sulit untuk dapat melakukan serangan *bruteforce* terhadapa algoritma *Composite Block* ini. Apabila tetap ingin dicoba akan membutuhkan waktu yang lama atau *resource* yang besar dan tidak akan sebanding dengan nilai informasi dari pesan yang dienkripsi.

B. *Serangan dengan Kunci Dekripsi Berdekatan*

Analisis ini dilakukan dengan mencoba melakukan dekripsi dengan kunci yang berdekatan, kemudian dilihat hasil dekripsinya apakah dapat ditemukan suatu pola atau kemiripan dengan pesan asli yang sebelumnya dienkripsi. Dengan percobaan dengan mengganti 1 karakter dari kunci tersebut, dapat dilihat pada tabel di bawah bahwa pesan yang dihasilkan sangat berbeda dan tidak ditemukan pola yang dapat dianalisa lebih lanjut. Oleh karena itu dapat disimpulkan bahwa algoritma *composite block cipher* ini bebas dari bentuk serangan analisis kunci berdekatan.

1234567890123451

Tabel 8. Kunci dekripsi

5a	7a	af	0d	05	09	9e	da	70	15	29	0d	ee
3a	9c	35	72	34	ef	01	1b	09	94	db	66	14
68	0d	e9	3a	9e	29	61	67	ad	5d	11	40	8a
8c	62	04	2d	5d	e3	26	93	33	2b	4e	fe	59
05	48	8a	99	23	29	2a	59	ef	32	95	28	69
2d	bd	51	06	5e	9d	96	2f	40	28	4c	e9	20
83	7a	20	78	ff	54	06	47	9b	cd	23	0d	2b
41	ef	26	84	33	6f	3e	f3	50	59	46	9a	8c
6c	12	27	44	aa	34	9c	33	78	39	f1	40	59
08	80	c3	6f	4c	64	48	ff	75	82	2f	64	61
e5	40	42	42	93	91	76	13	64	48	e4	3c	9d
7a	20	43	f9	50	04	58	87	98	23	36	2d	5b
eb	38	85	29	20	6e	e9	0f	09	46	d7	c5	66
0f	68	0d	fa	30	9c	36	20	60	a1	4e	5f	43
9b	d2	23	0e	21	4e	aa	21	85	28	15	09	c3
66	2c	46	9b	c5	03	60	44	2d	8a	55	f0	5a

Tabel 9. Hasil dekripsi mode ECB yang salah

Hasil dekripsi dengan menggunakan kunci yang berdekatan menghasilkan *hexdump* yang sangat berbeda dengan *hexdump* milik plainteks. Hal ini disebabkan oleh properti *diffusion* yang didapatkan melalui mekanisme transformasi dan substitusi pada *round function*.

C. *Known plainteks Attack*

Serangan dengan jenis ini memerlukan analisis yang lebih mendalam yang berbeda untuk setiap cipher blok. Keamanan *composite block cipher* ditentukan oleh struktur utamanya, yaitu skema *feistel network* beserta modus operasinya dan fungsi round yang terdapat di dalamnya. Bagaimana membangkitkan kunci internal juga menjadi faktor dalam hal ini.

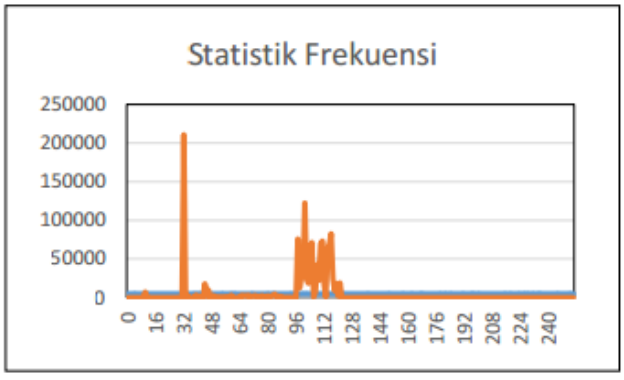
Dapat dilihat pada modus operasi *ECB*, blok pada plainteks yang sama akan menjadi blok cipherteks yang sama pula. Dengan demikian, apabila sebuah blok pada plainteks sudah diketahui hasil enkripsinya, dekripsi algoritma ini pada modus operasi *ECB* akan sangat cepat. Hal ini dapat diatasi dengan menggunakan modus operasi yang berbasis *stream cipher* seperti *CBC* dan *CFB*.

Fungsi yang digunakan untuk dalam fungsi *round* yang ada hanya memanfaatkan transformasi dan subtitusi sederhana serta peran dari algoritma MD5 juga membuat *cipher* ini memberikan keamanan yang baik walaupun cukup sederhana Jikalau seseorang mengetahui masukan dan keluaran dari satu round pada skema utama *feistel network*, menemukan kuncinya sangatlah sulit.

Walaupn belakangan ini MD5 sudah dikenal tidak aman lagi karena menemukan collision dari MD5 mudah, tetapi keluaran dari tiap round function telah disubtitusi dan ditransformasi yang dipengaruhi oleh *plainteks* dan kunci utamanya, sehingga sulit dipecahkan. Mengetahaui *plainteks* dari suatu *round* saja tidak akan cukup untuk dapat memecahkan keseluruhan isi dari pesan yang dienkripsi.

D. *Analisis Frekuensi*

Berikut akan dilakukan analisis frekuensi terhadap enkripsi plainteks Lorem Ipsum yang diperoleh dari *Open Data Character Frequency.* Berikut adalah statistik frekuensi kemunculan tiap byte pada plainteks yang cukup panjang dan *cipherteks* hasil enkripsi.



Gambar 3. Frekuensi kemunculan huruf dan angka pada plainteks dan ciperteks.

Angka pada sumbu x adalah byte (0 – 255), dan angka pada sumbu y adalah kemunculannya. Frekuensi pada plainteks ditunjukkan oleh garis oranye, dan frekuensi pada cipherteks

ditunjukkan oleh garis biru. Dapat dilihat bahwa pada plainteks Lorem Ipsum, frekuensi kemunculan huruf mendekati kemunculan huruf pada teks bahasa Inggris.

Pada plainteks, kemunculan tertinggi adalah pada karakter spasi dan karakter-karakter yang umum muncul pada bahasa Inggris seperti E, T, dan A. Sedangkan pada cipherteks, kemunculan tiap byte merata. Hal ini menyimpulkan bahwa algoritma ini aman dari analisis frekuensi.

VI. Kesimpulan dan Saran

Berdasarkan hasil eksperimen dan analisis keamanan, dapat disimpulkan bahwa algoritma enkripsi *Composite Block* berhasil mengenkripsi dan menghilangkan hubungan antara kunci, plainteks, dan cipherteks, hal ini terbukti lewat analisa dan percobaan yang menyimpulkan bahwa algoritma ini aman dari berbagai serangan yang umum dilakukan yaitu *bruteforce attack*, analisis kunci berdekatan, *known plainteks attack* dan juga analisa frekuensi. Rancangan algoritma *Composite Block* cipher dapat dikembangkan lebih lanjut, salah satunya adalah membuat panjang kunci lebih fleksibel.

References

[1] https://www.wolfssl.com/wolfSSL/Home.html, diambil pada tanggal 23 Maret 2016 pukul 17.00

[2] Shannon, Claude (1949). "Communication Theory of Secrecy Systems" (PDF). *Bell System Technical Journal* 28 (4): 656–715.

[3] Jonathan Sorenson, An Introduction to Prime Number Sieves, Computer Sciences Technical Report #909, Department of Computer Sciences University of Wisconsin-Madison, January 2, 1990

[4] *van Tilborg, Henk C. A.; Jajodia, Sushil, eds. (2011). Encyclopedia of Cryptography and Security. Springer. ISBN 978-1-4419-5905-8.*, p. 455.K.

[5] Chakraborty, D. & Rodriguez-Henriquez, F. (2008). "Block Cipher Modes of Operation from a Hardware Implementation Perspective". In Koc, Cetin K. *Cryptographic Engineering*. Springer. p. 321. ISBN 9780387718163.

[6] Paar, Cristof et al. (2010). *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer. p. 30. ISBN 9783642041006.

[7] Matsui, M. and Yamagishi, A. "A new method for known plainteks attack of FEAL cipher".*Advances in Cryptology - EUROCRYPT 1992*.