Cipher 1399: Algoritma Block Cipher

Anwar Ramadha
Program Studi Teknik Informatika
Institut Teknologi Bandung
Jl. Ganesha 10 Bandung 40132, Indonesia
anwar.ramadha@gmail.com

Drestanto M. Dyasputro
Program Studi Teknik Informatika
Institut Teknologi Bandung
Jl. Ganesha 10 Bandung 40132, Indonesia
dyas@live.com

***Abstrak*—Enkripsi pesan banyak digunakan untuk menjaga keamanan komunikasi. Sudah banyak algoritma enkripsi yang telah dikembangkan. Algoritma tersebut harus dirancang serumit mungkin agar sukar untuk dipecahkan. Block Cipher merupakan algoritma yang beroperasi pada sekumpulan bit dengan panjang sama yang disebut block. Algoritma enkripsi yang akan dikembangkan merupakan algoritma block cipher yang menawarkan solusi dengan keamanan yang berlapis namun tetap sederhana. 1399 chiper menerapkan jaringan feistel pada proses enkripsi dan dekripsinya. Algortima ini memiliki S-Box berukuran 16 x 16 seperti algoritma AES yang dibangkitkan dinamik secara pseudorandom berdasarkan seed tertentu. Algoritma ini juga memiliki jumlah perputaran (round) yang berbeda bergantung pada kunci eksternal.**

Kata kunci—kriptografi; enkripsi; block cipher; jaringan feistel

I. Latar Belakang

Kriptografi adalah ilmu yang menjaga kerahasiaan pesan dengan menyandikan pesan menjadi bentuk yang terlihat tidak bermakna. Ilmu ini sudah ada dan digunakan sejak zaman Mesir kuno dan terus berkembang hingga sekarang. Saat ini, kriptografi banyak digunakan misalnya untuk mengamankan pengiriman pesan, sistem pengamanan gedung, bahkan untuk mengamankan telepon seluler.

Pada awal kemunculannya, kriptografi hanya digunakan untuk mengenkripsi pesan sehingga tidak dapat dibaca oleh pihak lain. Namun, seiring dengan perkembangannya, kriptografi juga memberikan aspek-aspek keamanan yang lain, yaitu kerahasiaan, integritas data, otentikasi, dan nirpenyangkalan. Perkembangan tersebut juga telah melahirkan berbagai algoritma kriptografi modern baru yang menggunakan berbagai fungsi matematika, diantaranya fungsi permutasi dani substitusi, sehingga prinsip confusion dan diffusion terpenuhi. Contohnya di bidang komunikasi setiap kartu sim telah diimplementasikan algoritma A5. Sayangnya regulasi di Indonesia tidak memperbolehkan penggunakan enkripsi pada panggilan telepon.

Makalah ini akan membahas rancangan algoritma block cipher yaitu Chipher 1399.

II. Dasar Teori

A. Block Cipher

Dalam kriptografi, *block cipher* adalah algoritma deterministik yang beroperasi pada kelompok bit dengan panjang yang telah ditentukan, disebut *block*, dengan transformasi yang tidak berubah yang dispesifikasikan dengan kunci simetri. *Block cipher* adalah komponen penting dalam perancangan kriptografi dan secara luas digunakan untuk enkripsi dari data yang besar.

Setiap algoritma *block cipher* mempunyai enkripsi (E) dan dekripsi (D). Kedua alur ini mempunyai dua input, yaitu blok berukuran n-bit, dan kunci dengan panjang k-bit. Algoritma dekripsi pada *block cipher* di deskripsikan sebagai invers dari algoritma enkripsi, sehingga $D = E^{-1}$.

Pada algoritma *block cipher,* ada empat mode yang digunakan yaitu *Electronic Code Book* (ECB), *Cipher Block Chaining* (CBC), *Cipher Feedbock* (CFB), dan *Output Feedback* (OFB). Mode yang paling sederhana adalah ECB. Pesan akan dibagi menjadi beberapa blok, kemudian akan dienkripsi secara independen.

Kerugian menggunakan ECB adalah kurangnya difusi. Hal ini dikarenakan hasil enkripsi akan identik dengan blok plainteksnya dengan kata lain tidak menyembunyikan data dengan baik.

Pada tahun 1976, Ehrsam, Mayer, Smith, dan Tuchman mengenalkan *Cipher Block Chaining* (CBC). Pada CBC, setiap blok plainteks di XOR dengan blok cipherteks sebelumnya. Kemudian hasil dari operasi tersebut akan dienkripsi. Dengan begitu, setiap blok akan bergantung pada blok lainnya. Mode ini memang lebih kuat daripada ECB, namun memiliki kelemahan, yaitu jika terjadi kesalahan satu bit, maka akan berdampak pada bit-bit selanjutnya. Tetapi, hal ini berkebalikan dengan proses dekripsinya. Kesalahan satu bit pada blok cipherteks hanya memengaruhi blok plainteks yang berkoresponden dan pada satu bit plainteks berikutnya.

Pada mode CFB, data dienkripsi dalam unit yang lebih kecil daripada ukuran blok. Unit yang dienkripsi dapat berupa bit per bit (seperti *stream cipher*), dua bit, dan seterusnya. Bila unit yang dienkripsi satu karakter setiap kali, maka disebut

CFB 8-bit. CFB membutuhkan antrian dengan ukuran yang sama dengan ukuran blok.

Mode OFB mirip dengan CFB, kecuali n-bit daru hasil enkripsi terhadap antrian disalin menjadi elemen posisi paling kanan di antrian. Dekripsi dilakukan sebagai kebalikan dari proses enkripsi.

B. *Shannon's Confusion and Diffusion*

Pada tahun 1949, Claude Shannon memperkenalkan prinsip *confusion* dan *diffusion* dalam karyanya yang berjudul *Communication Theory of Secrecy System*. Kedua prinsip ini mengacu pada prinsip operasi penyandian. Tujuan prinsip ini adalah menggagalkan kriptanalis dalam memecahkan *cipherteks* dengan metode analisis statistik.

Shannon mendefinisikan *confusion* sebagai proses perancangan plainteks, kunci dan cipherteks yang memiliki hubungan serumit mungkin. Contohnya pada Caesar cipher setiap huruf di subtitusi dengan huruf yang sama, sehingga cipher ini mudah untuk dipecahkan. *Confusion* digunakan agar kriptanalis kesulitan untuk menemukan pola-pola statistik yang muncul pada cipherteks. Sedangkan *diffusion* adalah prinsip yang digunakan untuk menyebarkan pengaruh bit plainteks atau kunci ke seluruh cipherteks. Perubahan satu atau dua bit pada plainteks akan membuat perubahan yang tidak dapat diprediksi pada cipherteks. Prinsip ini mempunyai tujuan yang sama dengan *confusion*. Untuk memperoleh tingkat keamanan cipher yang tinggi, kedua prinsip ini diterapkan secara berulang pada sebuah blok tunggal dengan kombinasi yang berbeda.

C. Jaringan Feistel

Jaringan Feistel merupakan struktur simetris pada suatu algoritma *block cipher*. Model ini sudah umum digunakan pada algoritma *cipher* modern. Nama *feistel* berasal dari nama fisikawan dan kriptografer asal Jerman, Horst Feistel yang menjadi pelopor riset saat bekerja di IBM. Penggunaan model ini mempunyai kenuntungan yang terletak pada kemiripan operasi enkripsi dan dekripsi, bahkan identik untuk beberapa kasus. Sehingga hanya membutuhkan kunci yang dibalik.

Pada jaringan feistel, blok plainteks harus dibagi dua, bagian kanan dan kiri. Bagian kanan blok plainteks akan menjadi bagian kiri blok cipherteks, sementara bagian kiri blok plainteks akan di-XOR-kan dengan hasil operasi bagian kanan blok plainteks. Operasi tersebut berbeda-beda bergantung pada jenis algoritmanya. Setelah itu, bagian kiri blok plainteks yang telah di-XOR-kan akan menjadi bagian kanan dari blok cipherteks.

Jaringan feistel ini sering digunakan karena memiliki sifat reversible sehingga untuk proses dekripsi, dapat digunakan algoritma yang sama dengan algoritma enkripsinya. Dalam satu kali proses enkripsi, blok bit dapat melewati jaringan feistel berkali-kali untuk menambah kerumitan proses.

III. Rancangan block cipher

A. Penggunaan Struktur Feistel

Struktur yang digunakan dalam block cipher ini adalah struktur Feistel. Sehingga enkripsi dan dekripsi akan memiliki struktur yang sama. Feistel yang kami gunakan membagi plainteks sama rata menjadi plainteks kiri dan plainteks kanan.

Plainteks akan dibagi menjadi L (plainteks kiri) dan R (plainteks kanan). Plainteks kanan dimasukkan ke dalam sebuah fungsi F (akan dijelaskan pada halaman berikutnya) dan menjadi plainteks kiri untuk iterasi berikutnya. Plainteks kiri di-XOR-kan dengan plainteks kanan yang sudah dimasukkan ke dalam fungsi F dan menjadi plainteks kanan untuk iterasi berikutnya.

Sehingga, formula feistel yang kami gunakan adalah

$$L' \leftarrow R$$

$$R' \leftarrow L \oplus F(R)$$

Untuk dekripsi, cukup membalik formula yaitu

$$R \leftarrow L'$$

$$L \leftarrow R' \oplus F(L')$$

(Keterangan : L, R, L', dan R' berturut-turut adalah plainteks kiri, plainteks kanan, cipherteks kiri, dan cipherteks kanan)

B. Penentuan jumlah iterasi

Key awal yang dimiliki berbentuk string. Setiap character dari string akan direpresentasikan dalam angka-angka dan dijumlahkan. Hasil penjumlahan akan di-mod dengan panjang string dan ditambah 1 untuk mendapatkan nilai n.

Contoh :

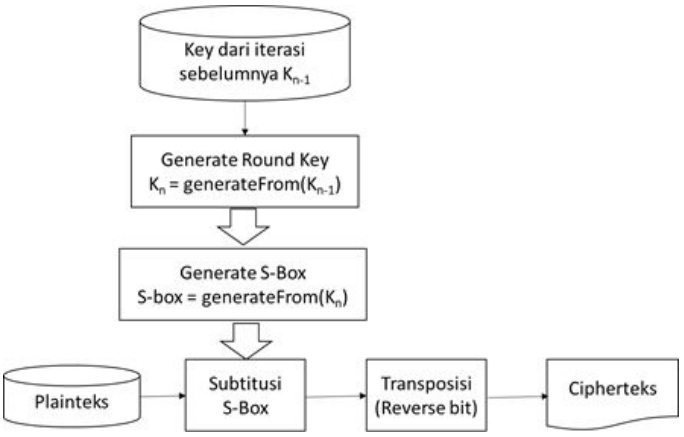
Key = "tes" -> panjang kunci = 3

$$n = ('t' + 'e' + 's') \bmod 3 + 1$$

$$= 116+101+115 \bmod 3 + 1 = 3$$

Maka iterasi dilakukan sebanyak 3 kali.

Fungsi F dan pembangkitan key K_n pada struktur Feistel di-detilkan dengan gambar di bawah ini



Gambar 1 : Rancangan Block Cipher

C. *Generate round key*

Key pertama menjadi *seed* untuk *pseudo random generator*. Kemudian melakukan 3 kali *generate random* 0-255 sehingga didapatkan 3 buah karakter. 3 buah karakter ini akan digunakan untuk menjadi *round key* yang akan men-*generate* S-Box sendiri.

Untuk putaran Feistel berikutnya, *key* yang digunakan untuk men-*generate round key* adalah menggunakan *round key* sebelumnya.

D. *Generate* S-Box

Ukuran S-Box adalah 16 x 16 heksadesimal.

S-Box *di-generate* dengan menggunakan *round key* sebagai *seed* dari *pseudo random generator*. Dilakukan *generate* bilangan random 0-15.

Hasil pertama *pseudo random* dijadikan nomor baris pertama, hasil kedua menjadi nomor kolom pertama. Sehingga baris dan kolom pada S-Box tersebut bernilai "00" heksadesimal. Setelah itu, random kedua dan ketiga menjadi baris dan kolom untuk "01" heksa, begitu seterusnya. Jika didapat hasil *random* dengan nomor baris dan nomor kolom yang sudah terisi, maka sisa S-Box yang belum terisi akan diisi secara sekuensial.

Contoh

Key = "baru".

Menggunakan *pseudo random generator* dari python masukkan kata "baru" sebagai *seed random*. S-Box awalnya kosong. Saat dilakukan random, didapatkan bilangan *random* pertama adalah 4, dan bilangan *random* kedua adalah 15. sehingga, S-Box pertama yang diisi adalah pada baris ke-4 kolom ke-f (heksadesimal dari 15). Sel ini akan diisi dengan "00" heksa.

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0																
1																
2																
3																
4																00
5																
6																
7																
8																
9																
a																
b																
c																
d																
e																
f																

Setelah itu, dilanjutkan dengan men-*generate* bilangan *random* kembali untuk baris dan kolom berikutnya, didapatkanlah angka 11 dan 0. Sehingga sel pada baris ke-b dan kolom ke-0 akan diisi dengan "01" heksa. Begitu seterusnya hingga terbentuk S-Box seperti ini.

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0		33												09		
1	19	2c		2f								26				
2			22	0c					14							
3	12		0a			06					1d		1a			
4		2b								1c						00
5	2a						17					04			31	
6												13	27			
7	10	0e		29							15			0b	20	
8				1e	25	21	02	03								2e
9		0d			2d											
a	08						0f			32						
b	01	16	28	1b			23			07			05		30	
c		1f			18											
d		11														
e																
f													24			

Setelah bilangan heksa 33 terisi, di-*generate* lagi 2 bilangan *random*. Bilangan random yang dihasilkan adalah 5 dan 14. Namun, baris ke-5 kolom ke-e sudah terisi dengan bilangan sebelumnya. Sehingga, pengisian secara acak dihentikan, dilanjutkan sengan pengisian secara sekuensial. S-Box yang dihasilkan secara sekuensial adalah sebagai berikut :

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

E. Subtitusi S-Box

Untuk setiap blok, dikelompokkan menjadi pasangan heksadesimal. Pasangan tersebut menjadi baris dan kolom pada S-Box, kemudian dilakukan subtitusi seperti biasa.

Contoh :

Misalnya digunakan S-Box di atas untuk mengenkripsi kata “aku”. Kata “aku” apabila dijadikan pasangan heksadesimal menjadi 61, 6b, dan 75.

Apabila dilakukan subtitusi S-Box di atas, maka 61 menjadi 80, 6b menjadi 13, dan 75 menjadi 8f atau ASCII ke-128, ke-19, dan ke-143. Dalam bentuk text, diperlihatkan sebagai berikut :

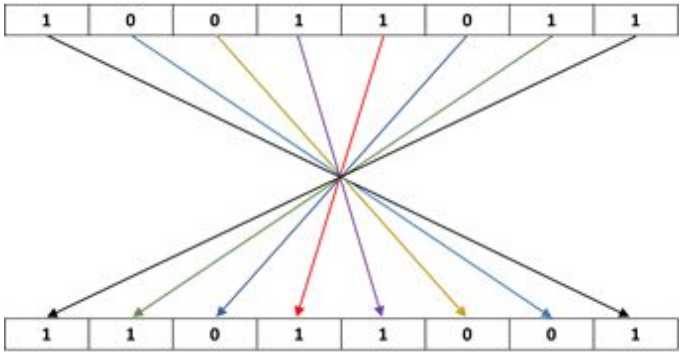
“aku” -> “Ç□Å”

F. Transposisi

Untuk setiap blok-nya, dilakukan reverse bit-bit nya. Bit paling depan dalam blok menjadi bit paling belakang, bit paling belakang menjadi bit terdepan. Bit kedua menjadi bit kedua terakhir, dan seterusnya.

Contoh :

Misalnya blok hanya terdiri dari 8 bit (pada kenyataannya, akan digunakan ukuran blok 64 bit, 8 bit hanya penyederhanaan untuk mempermudah penjelasan). Blok berisi 10011011. Maka dilakukan reverse seperti gambar berikut



Gambar 2 : Visualisasi transposisi

Maka, hasil yang didapatkan untuk blok tersebut adalah 11011001. Setiap blok diproses secara independen sehingga ukuran blok yang digunakan akan sangat berpengaruh pada hasil untuk mendapatkan cipherteks.

IV. Simulasi dan Pembahasan Hasil

A. Simulasi berbagai mode (ECB, CBC, CFB, OFB, dan counter)

Kunci : ITB – Institut Teknologi Bandung

Plainteks :

Red: These walls are funny. First you hate 'em, then you get used to 'em. Enough time passes, you get so you depend on them. That's institutionalized.

Heywood: Shit. I could never get like that.

Ernie: Oh yeah? Say that when you been here as long as Brooks has.

Red: Goddamn right. They send you here for life, and that's exactly what they take. The part that counts, anyway.

Cipherteks (ECB) :

®pÓÝÄ

¾Óp╔Ïçt╩W─áq*ÿ,ê GW1ù|-Ê°fÿãÚDJ81võ
¼▒¯w5Ú▀ÎF┴zÖGÓ╗E¡ü┴┤æÜ╗▒¯ç

õ■ÿ▒ÂH └ÁKÇF┴%½Ï>ÊÙáQÌÄ~nE;1┬Ýó(═´ä<Á3û├u6âÖÏÑÑa
À▄Á*▒µb▀Ü¬■²X³QéuÓ▄ V─îõÏwæÅíÐøVëcü╩EÃb1╠▒k■ï7O)e
b=®yýð*─:_X¥▒ª'ÓXyÍÃÜ):ù&yV₁¼┼"ÐPý>Ôk>oÉ^ªâê╝ \ýÍ┌├Ü*ÍïïR*õ
Ò_8w┤Ê┤°®è ┘P¯éHú5éù¿TçÍÕí®ãïDm{

÷[pgÞ─>sX=_Ñð╣9╔╪┐\<þ2\Å}■¡4T><qGø├<½5{╬ïj┌®wv
╩▒qÚ6ù

Á╝>

Waktu eksekusi enkripsi : 0.172000169754 sekon

Waktu eksekusi dekripsi : 0.176999807358 sekon

Cipherteks (CBC) :

®pÓÝÄ

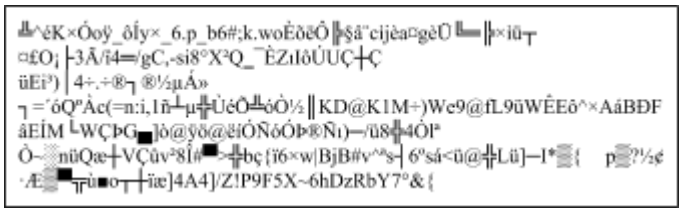
i<Z.P₁Ã╬'■2Ü· ┐J£ └Tý╣ìÚP}}3═ãSðß┘ 0²;ò<oÑy}ð │0┼Ï¼¾k┼í2Zæ¢;
ð¿0Ï╬F4©ô┬«°Æ¨Èá └«|┬ó,2ÎC9¤ã;V ├-Ý8v^┴PRMøà ê½CPIb© ╚├ã]
à ûÀ┼éã■cµÉ®¡á ╟FÝ²:┼Ñ¬éeı╝ w■*8Ó └╮~ÿ9>┤"
╬é'■'═■Ï¼)aë_ÔM┤ mîk Úð¾Æª

■)k°ÊuFÁLÞ0nÎ1&·¯Ø╢JYHO╢Ê╬pÿj┴xð,´ý■0ºØøH s|»,}4ðïd5Êûäé
ßlêÜ0b&┴═

Waktu eksekusi enkripsi : 0.335000038147 sekon

Waktu eksekusi dekripsi : 0.321000099182 sekon

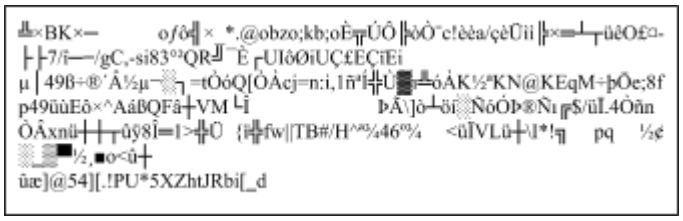
Cipherteks (CFB) :



Waktu eksekusi enkripsi : 0.953999996185 sekon

Waktu eksekusi dekripsi : 0.722999811172 sekon

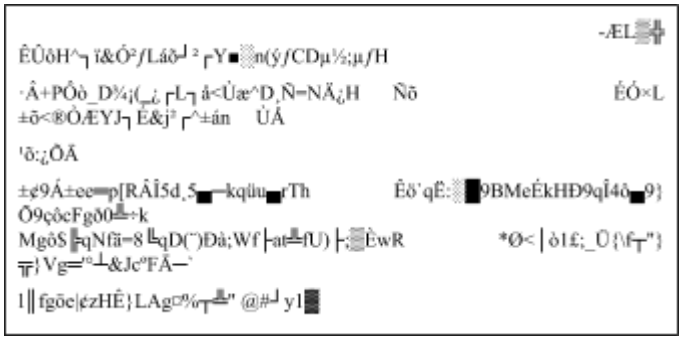
Cipherteks (OFB) :



Waktu eksekusi enkripsi : 0.745000123978 sekon

Waktu eksekusi dekripsi : 0.711999893188 sekon

Cipherteks (counter) :



Waktu eksekusi enkripsi : 0.114000082016 sekon

Waktu eksekusi dekripsi : 0.114000082016 sekon

Terlihat bahwa mode yang paling hemat waktu adalah mode counter. Hal ini dikarenakan enkripsi dan dekripsi dilakukan bukan pada plainteks, namun pada counter. Pada algoritma yang diimplementasi, counter yang digunakan cukup sederhana sehingga enkripsi pun memakan waktu yang cepat.

Untuk analisis-analisis berikutnya dalam melakukan pengujian keamanan algoritma, digunakan mode yang paling ringkih yaitu mode ECB.

B. Analisis frekuensi

Untuk melakukan analisis *confusion* dan *diffusion*, akan dilihat distirbusi huruf yang dihasilkan. Untuk itu, dilakukan penghitungan unigram dengan menggunakan tools online yaitu http://guidetodatamining.com/ngramAnalyzer

Data yang dimasukkan berupa text dalam representasi heksadesimal karena tools ini tidak dapat melakukan proses seluruh tipe ASCII.

Kunci : ITB = Institut Teknologi Bandung

Plainteks :

Red: These walls are funny. First you hate 'em, then you get used to 'em. Enough time passes, you get so you depend on them. That's institutionalized.

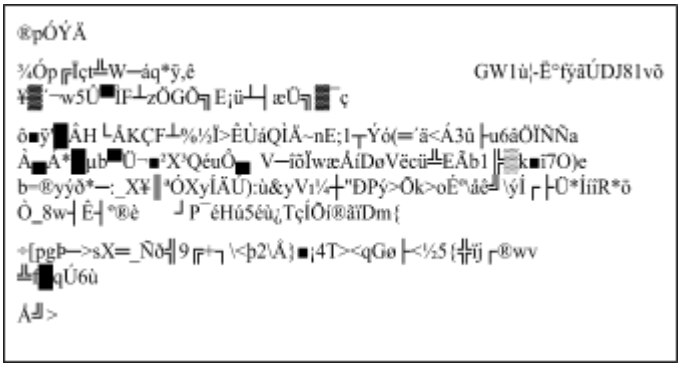
Heywood: Shit. I could never get like that.

Ernie: Oh yeah? Say that when you been here as long as Brooks has.

Red: Goddamn right. They send you here for life, and that's exactly what they take. The part that counts, anyway.

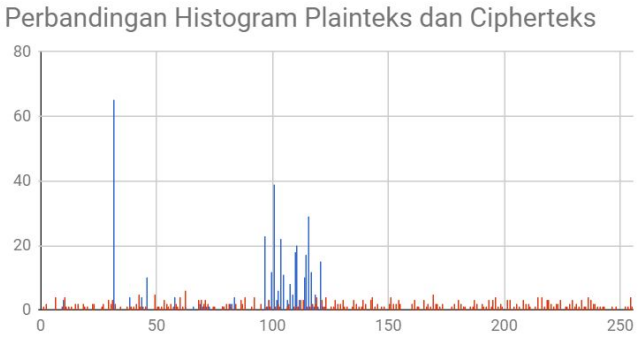
ngram	count	frequency
20	65	17.379679144385
65	39	10.427807486631
74	29	7.7540106951872
61	23	6.1497326203209
68	22	5.8823529411765
6f	20	5.3475935828877
6e	18	4.8128342245989
73	17	4.5454545454545
79	15	4.0106951871658
75	12	3.2085561497326
64	12	3.2085561497326
69	11	2.9411764705882
72	10	2.6737967914439
2e	9	2.4064171122995

Cipherteks :



ngram	count	frequency
80	6	1.6129032258065
2c	5	1.3440860215054
77	5	1.3440860215054
83	5	1.3440860215054
e8	5	1.3440860215054
10	5	1.3440860215054
5f	5	1.3440860215054
67	5	1.3440860215054
6e	4	1.0752688172043
a7	4	1.0752688172043
a6	4	1.0752688172043
4e	4	1.0752688172043
95	4	1.0752688172043
fb	4	1.0752688172043

List frekuensi kemunculan karakter plainteks dan cipherteks dapat dengan mudah dibedakan dengan melihat histogram berikut (histogram plainteks berwarna biru sedangkan histogram cipherteks berwarna merah) :



Terlihat bahwa frekuensi dari cipherteks lebih merata dibandingkan dengan plainteks. Sehingga, analisis frekuensi tidak dapat dilakukan untuk memecahkan Cipher 1399.

C. Analisis pengubahan kecil kunci

Algoritma kriptografi yang aman akan memberikan hasil cipherteks yang sangat berbeda dengan diubahnya kunci sedikit. Berikut adalah hasil pengujian untuk pengubahan 1 karakter pada kunci.

Plainteks :

Red: These walls are funny. First you hate 'em, then you get used to 'em. Enough time passes, you get so you depend on them. That's institutionalized.

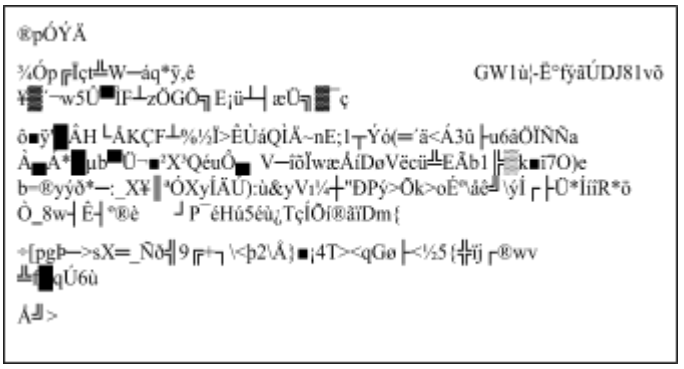
Heywood: Shit. I could never get like that.

Ernie: Oh yeah? Say that when you been here as long as Brooks has.

Red: Goddamn right. They send you here for life, and that's exactly what they take. The part that counts, anyway.

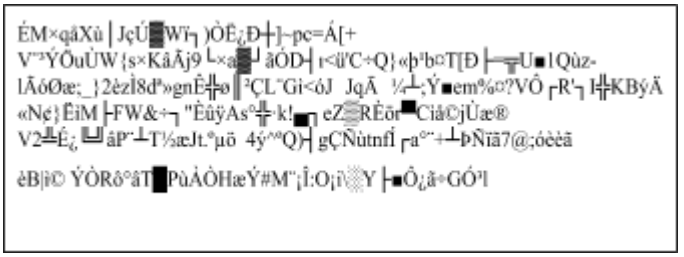
Kunci : ITB = Institut Teknologi Bandung

Cipherteks :



Kunci (i menjadi o) : ITB = Institut Teknologo Bandung

Cipherteks :



Terlihat bahwa 1 karakter pengubahan pada kunci menghasilkan cipherteks yang sama sekali berbeda

D. Analisis pengubahan kecil plainteks

Algoritma kriptografi yang aman akan memberikan hasil cipherteks yang sangat berbeda dengan diubahnya plainteks sedikit. Berikut adalah hasil pengujian untuk pengubahan 1 karakter pada plainteks.

Kunci : ITB = Institut Teknologi Bandung

Plainteks :

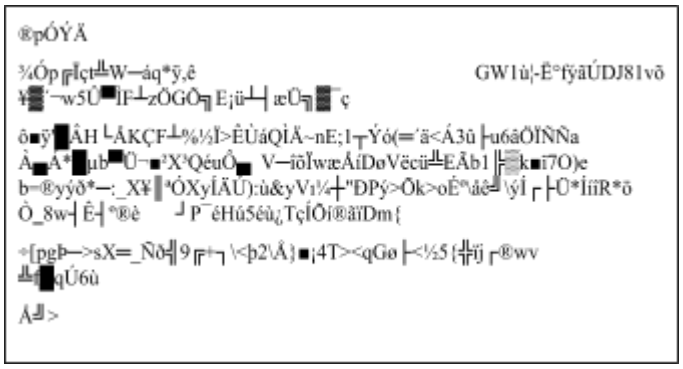
Red: These walls are funny. First you hate 'em, then you get used to 'em. Enough time passes, you get so you depend on them. That's institutionalized.

Heywood: Shit. I could never get like that.

Ernie: Oh yeah? Say that when you been here as long as Brooks has.

Red: Goddamn right. They send you here for life, and that's exactly what they take. The part that counts, anyway.

Cipherteks :



Plainteks (menghapus titik di akhir) :

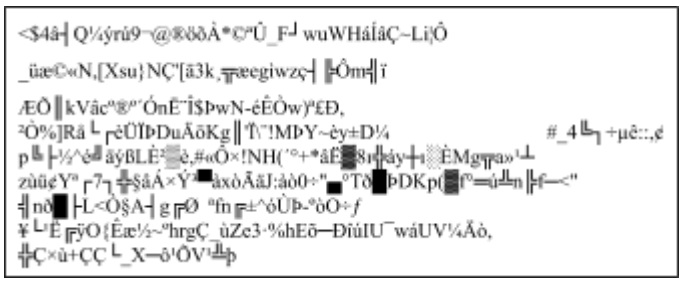
Red: These walls are funny. First you hate 'em, then you get used to 'em. Enough time passes, you get so you depend on them. That's institutionalized.

Heywood: Shit. I could never get like that.

Ernie: Oh yeah? Say that when you been here as long as Brooks has.

Red: Goddamn right. They send you here for life, and that's exactly what they take. The part that counts, anyway

Cipherteks :



E. Estimasi kriptanalisis

Kriptanalisis tidak dapat dilakukan dengan analisis frekuensi, karena frekuensi kemunculan bilangan heksadesimal lebih merata pada cipherteks.

Kriptanalisis dapat dilakukan dengan menggunakan *brute force attack*. Dengan struktur yang cukup rumit, *attacker* tidak dapat melakukan *brute force* pada S-Box ataupun *round key*, paling logis adalah dengan menebak kunci awal. Algoritma ini menggunakan kunci dengan panjang yang tidak ditentukan, sehingga, sangat sulit untuk menebak kunci yang digunakan.

Apabila kunci memiliki panjang n, maka ada 256^n kemungkinan kunci. Jika digunakan panjang kunci yang wajar, yaitu 16 karakter, maka brute force attack harus mencoba 256^{16} kunci. Dengan komputer yang memiliki kemampuan menebak satu juta kunci perdetik, maka dibutuhkan $256^{16} / 10^6$ detik atau sekitar 1.08×10^{25} tahun.

V. Kesimpulan dan Saran Pengembangan

A. Kesimpulan

Setelah dilakukan analisis-analisis yang dilakukan yaitu terkait frekuensi kemunculan huruf, serta pengubahan kecil kunci maupun plainteks, terbukti bahwa Cipher 1399 cukup aman dalam melakukan enkripsi. Karena penggunaan struktur feistel yang menyebabkan enkripsi dan dekripsi memiliki struktur yang sama, maka dekripsi pun tidak mudah ditebak.

Kompleksitas waktu yang dibutuhkan untuk melakukan brute force attack pada algoritma ini eksponensial terhadap panjang kunci. Sehingga, semakin panjang kunci yang digunakan, semakin aman pula algoritma ini.

B. Kelemahan dan saran pengembangan

Kelemahan dari Cipher 1399 terletak pada saat dilakukan *generate* dari S-Box dari *round key*.

"Jika didapat hasil *random* dengan nomor baris dan nomor kolom yang sudah terisi, maka sisa S-Box yang belum terisi akan diisi secara sekuensial"

Hal ini hampir pasti terjadi sehingga box (sel tabel) dengan heksadesimal baris f dan kolom f akan didekripsi dengan ff (tidak berubah).

Pengembangan yang dapat dilakukan adalah pada langkah *generate* S-Box. Apabila baris dan kolom yang didapat sama, sebaiknya ada penanganan sehingga sisa S-Box yang kosong diisi secara acak (dan bukan secara sekuensial). Misalnya, yang diisi adalah box setelahnya, apabila setelahnya sudah terisi, maka diisi setelahnya lagi, dan seterusnya.

Pengembangan lain yang dapat dilakukan berkaitan dengan struktur feistel yang digunakan. Cipher 1399 menggunakan feistel yang membagi plainteks menjadi dua bagian sama besar. Apabila ukuran plainteks ganjil, maka byte yang berada di tengah plainteks tidak dienkripsi, tetap berada di posisi tengah. Pengembangan yang dapat dilakukan adalah dengan melakukan penanganan apabila ukuran plainteks ganjil. Misalnya, menggunakan struktur feistel yang *unbalanced* (tidak harus membagi dua bagian sama rata).

References

[1] A. Menezes, P. van Oorschot, dan S. Vanstone. 2001. *Handbook of Applied Cryptography (5th ed.)*. CRC Press

[2] Munir, Rinaldi. 2018. *Slide Kuliah IF4020 Kriptografi: Algoritma Kriptografi Modern*

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

Pernyataan

Dengan ini kami menyatakan bahwa makalah yang kami tulis ini adalah tulisan kami berdua, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 16 Maret 2018

Anwar Ramadha - 13514013

Drestanto M. Dyasputro - 13514099