PlayUnfair: Block Cipher dengan Kotak S yang Sulit Ditebak

Muhammad Al Terra, **Aidil Rezjki Suljztan Syawaludin**.

[1,2] Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika (STEI), Institut Teknologi Bandung (ITB), Jalan Ganesha 10, Bandung 40132
E-mail: 13517145@std.stei.itb.ac.id, 13517070@std.stei.itb.ac.id

Abstract. Keamanan informasi adalah hal yang penting pada masa modern ini. Teknik kriptografi adalah salah satu teknik perlindungan informasi yang dapat digunakan. Teknik-teknik kriptografi ini bisa jadi adalah teknik kriptografi simetris dan asimetris. Beberapa teknik kriptografi simetris yang digunakan di antara lain adalah: DES, AES, Blowfish dan Twofish. Jurnal ini membahas kemungkinan untuk mengkonstruksi sebuah algoritma blok cipher yang mengadaptasi kekuatan Twofish untuk membangkitkan S-box yang dinamis tetapi juga masih memanfaatkan proses substitusi yang kompleks. Kami memilih algoritma Playfair dan pada akhirnya mengkombinasikannya dengan S-box statis milik Rijndael. Hasilnya adalah algoritma memiliki nilai Avalanche Effect yang diinginkan, 53.9% dan memiliki korelasi kuat antara kunci dan cipherteks . **Keywords**: Playfair, Block Cipher, Extended Playfair, Cryptography

1. Introduction

1.1 Latar Belakang

Seiring dengan berkembangnya teknologi dapat diamati bahwa informasi memiliki peranan yang semakin penting dan kegiatan-kegiatan yang memiliki hal krusial, seperti dokumen kerja, dokumen perjanjian bisnis dan dokumen-dokumen penting yang didalamnya memiliki informasi yang dapat membahayakan orang, perusahaan dan instansi pemerintahan selalu diedarkan dengan saluran-saluran yang bisa jadi diintai, dibaca atau bocor ke pihak-pihak yang tidak bertanggung jawab. Kesadaran orang-orang juga telah meningkat tentang pentingya perlindungan terhadap informasi tersebut. Pendekatan untuk melindungi informasi ini dapat dengan 2 cara, informasi dapat disembunyikan dengan tujuan utama agar orang tidak menyadari bahwa di suatu pesan ada suatu rahasia dengan metode steganografi, atau informasi dapat disembunyikan dengan menggunakan sandi yang membuat teks menjadi teks yang seolah-olah tidak memiliki makna dengan metode kriptografi. Pada jurnal ini akan dibahas metode penyembunyian pesan dengan teknik kriptografi.

1.2 Kriptografi Simetris dan Asimetris

Ada dua teknik kriptografi yang dikenal yaitu adalah metode kriptografi asimetris dan kriptografi simetris. Kriptografi simetris adalah teknik kriptografi yang menggunakan kunci dekripsi yang sama dengan kunci enkripsi algoritma-algoritma enkripsi yang menggunakan pendekatan ini antara lain Vigenere, Playfair, Hill dan Affine [1]. Kriptografi asimetris atau kriptografi kunci publik berbeda dengan kriptografi simetris di aspek kuncinya, kunci yang digunakan untuk melakukan enkripsi dan

dekripsi berbeda pada kriptografi asimetris, apabila diibaratkan dengan sederhana, pengirim adalah seseorang yang mengirimkan kotak kosong dengan gembok yang akan mengunci apabila ditekan, kunci dari gembok itu disimpan oleh pengirim, sehingga saat ada seseorang yang mengirim suatu objek pada kotak tersebut, orang tersebut cukup menekan gembok dan kotak itu terkunci. Hanya pengirim kotak yang bisa membuka gembok dengan kunci [2].

1.3 Block Cipher

Pada kriptografi modern ada yang disebut sebagai *block cipher*. *Block cipher* bekerja beda dengan *cipher* pada umumnya, jika *cipher* pada umumnya beroperasi pada 1 bit data, *block cipher* bekerja pada blok-blok data yang nantinya akan menghasilkan data dengan panjang yang sama. *Block cipher* sendiri memiliki mode-mode tergantung dari cara pemakaiannya, mode-mode ini diantaranya adalah *CBC (Cipher Block Chaining)*, *EBC (Electronic Book Code)*, *Counter Mode* dan *Cipher Feedback*. Perbedaan dari kelimanya terletak pada bagaimana *cipher* tersebut digunakan untuk menentukan bagaimana pembagian blok yang akan dienkripsikan seperti apa.

1.4 Analisis Block Cipher lain

Beberapa algoritma block cipher yang terkenal antara lain adalah AES, DES, Blowfish dan Twofish. Algoritma yang kami usulkan mencoba untuk mengkombinasikan beberapa kekuatan dari algoritma-algoritma yang sudah disebutkan di atas. Pemikiran pertama kami dapat ditinjau melalui bagaimana implementasi AES dilakukan, kami mendapati bahwa AES menggunakan implementasi S-box yang bersifat statik, ide ini berbeda dengan prinsip desain yang digunakan oleh algoritma Twofish, algoritma Twofish memperkenalkan 4 S-box yang dibangkitkan secara dinamis mengikuti suatu algoritma yang apabila dicari di internet sulit untuk ditemukan. Karena S-box pada dasarnya mengimplementasikan metode substitusi pada plainteks, kami ingin menambahkan aspek transposisi pada proses enkripsi dengan mengenalkan Extended Playfair untuk mengonstruksi S-box dinamik untuk S-box pertama dan kedua lalu melanjutkannya dengan substitusi dengan S-box ketiga berdasarkan S-box yang diberikan oleh Rijndael sebelum memasuki jaringan Feistel yang digunakan pada penerapan AES.

2. Teori Dasar

2.1. Playfair Cipher

Playfair cipher adalah cipher poligram yang dikembangkan oleh Charles Wheatstone dan Lyon Playfair pada 1854. *Playfair cipher* ini digunakan berbeda pada umumnya karena *Playfair cipher* memanfaatkan bigram bukan hanya karakter per karakter. *Playfair cipher* ini digunakan dengan pertama menyusun suatu matriks dengan huruf-huruf yang berasal dari kunci setelah duplikatnya dihilangkan dan cipher ini tidak memiliki huruf j. Dalam melakukan enkripsi cipher playfair mengikuti aturan yang ditentukan berdasarkan kolom atau baris yang ditempati bigram yang berada pada plainteks dan pada kunci, huruf-huruf tersebut ditukar -tukar mengikuti aturan sebagai berikut [3]:

1. Jika dua huruf terdapat pada baris kunci yang sama maka tiap huruf diganti dengan huruf di kanannya (bersifat siklik)
2. Jika dua huruf terdapat pada kolom kunci yang sama maka tiap huruf diganti dengan huruf di bawahnya (bersifat siklik)
3. Jika dua huruf tidak pada baris yang sama atau kolom yang sama maka
    a. huruf pertama diganti dengan huruf pada perpotongan baris huruf pertama dengan kolom huruf kedua
    b. huruf kedua diganti dengan huruf pada titik sudut keempat dari persegi panjang yang dibentuk dari tiga huruf yang digunakan

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Contoh gambar kotak Playfair (sumber: Cryptography and Network Security)

Kelemahan *Playfair cipher* ada pada ukuran poligram yang berada pada matriksnya, selain itu *Playfair cipher* dapat dipecah dengan analisis frekuensi dengan kata-kata pada bahasa Inggris. *Playfair cipher* pada dasarnya terlihat seperti sebuah S-box yang memiliki aturan unik dalam melakukan substitusi, kelemahan yang dialami Playfair ini akan menjadi sukar apabila ukuran dari S-box dibesarkan dan pembentukan S-box Playfair tidak hanya akan bergantung pada susunan awal kunci tetapi juga akan dilakukan berbagai pergeseran dan pertukaran yang membuat S-box bervariasi tinggi [3].

2.2. Extended Playfair Cipher

Algoritma Extended Playfair merupakan perluasan dan penguatan dari algoritma Playfair sederhana. Algoritma ini menggunakan alfabet *extended* ASCII sebagai karakter yang mengisi matriks 16x16. Algoritma Extended Playfair dapat mengatasi kelemahan algoritma playfair sederhana dengan menambah kompleksitas yang menyukarkan kriptanalisis dengan frekuensi. Extended Playfair juga menyulitkan dekripsi menggunakan kunci yang diketahui hanya sebagian karena kata kuncinya akan jauh lebih panjang [4]. Selain itu, Extended Playfair juga akan menerapkan aspek *confusion* Shannon karena aturan substitusi yang digunakan lebih kompleks dibandingkan S-box pada umumnya. Kemungkinan S-box yang dihasilkan adalah sangat tinggi, apabila kunci yang digunakan sangat panjang jumlah S-box dapat mencapai 256!. Menggunakan metode Stirling maka dapat diperkirakan batas atas jumlah digit dari S-box tersebut adalah:

$$Approximation\ of\ Upper\ Bound\ for\ Digits = \sum_{i}^{256} log(i) = 506.9333950412657$$

Atau dengan representasi biner angka tersebut akan mencapai kisaran:

$$Approximation\ of\ Bit\ Length\ Upper\ Bound = \sum_{i}^{256} log2(i) = 1683.9962872242136$$

Angka yang dihasilkan tidaklah kecil dan kemungkinan untuk menghasilkan kotak S yang bisa bermacam-macam untuk melakukan enkripsi menerapkan prinsip *confusion* Shannon.

P	l	a	y	f	i	r	.	(	S	m	p	e	)	NUL	☺
☻	♥	♦	♣	♠	•	◘	○	◙	♂	♀	♪	♫	☼	►	◄
↕	‼	¶	§	▬	↨	↑	↓	→	←	∟	↔	▲	▼	Space	!
“	#	$	%	&	‘	*	+	,	-	/	0	1	2	3	4
5	6	7	8	9	:	;	<	=	>	?	@	A	B	C	D
E	F	G	H	I	J	K	L	M	N	O	Q	R	T	U	V
W	X	Y	Z	[	\	]	^	_	`	b	c	d	g	h	j
k	n	o	q	s	t	u	v	w	x	z	{	|	}	~	DEL
Ç	ü	é	â	ä	À	å	ç	ê	ë	è	ï	î	ì	Ä	Å
É	æ	Æ	ô	ö	Ò	û	ù	ÿ	Ö	Ü	¢	£	¥	Pts	*f*
á	í	ó	ú	ñ	Ñ	ª	º	¿	⌐	¬	½	¼	¡	«	»
░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	▉	▄	▌	▐	▀
α	ß	Γ	π	Σ	Σ	µ	T	Φ	Θ	Ω	δ	∞	φ	ε	∩
≡	±	≥	≤	⌠	⌡	÷	≈	°	∙	·	√	n	2	■	

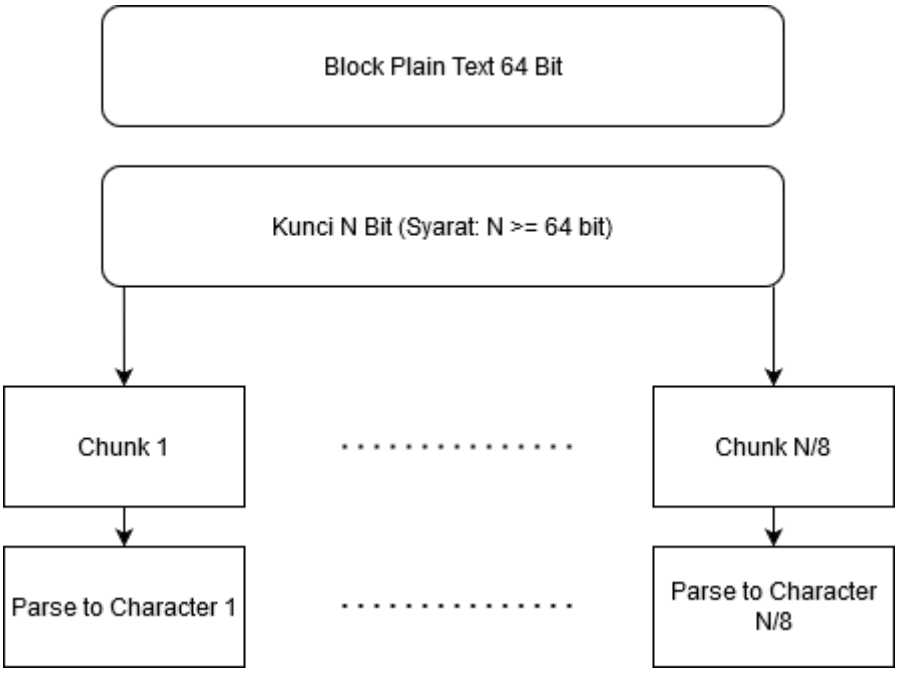
Contoh kotak Playfair extended (Sumber:[3])

3. Desain Cipher

3.1. Konstruksi S-box Pertama

Algoritma diawali dengan melakukan konstruksi S-box yang mengikuti aturan *Playfair cipher* 16x16 dengan mengisikan kunci lalu menghilangkan duplikat dari kunci tersebut. Proses ini diawali dengan membagi plainteks menjadi blok-blok berukuran 64 bit. Ukuran kunci yang digunakan harus minimal

sebesar 64 bit pula. Kunci yang berukuran 64 bit tersebut dibagi menjadi blok-blok yang berukuran 8 bit sesuai dengan ukuran karakter *extended* ASCII dan dilakukan konversi menjadi karakter ASCII.

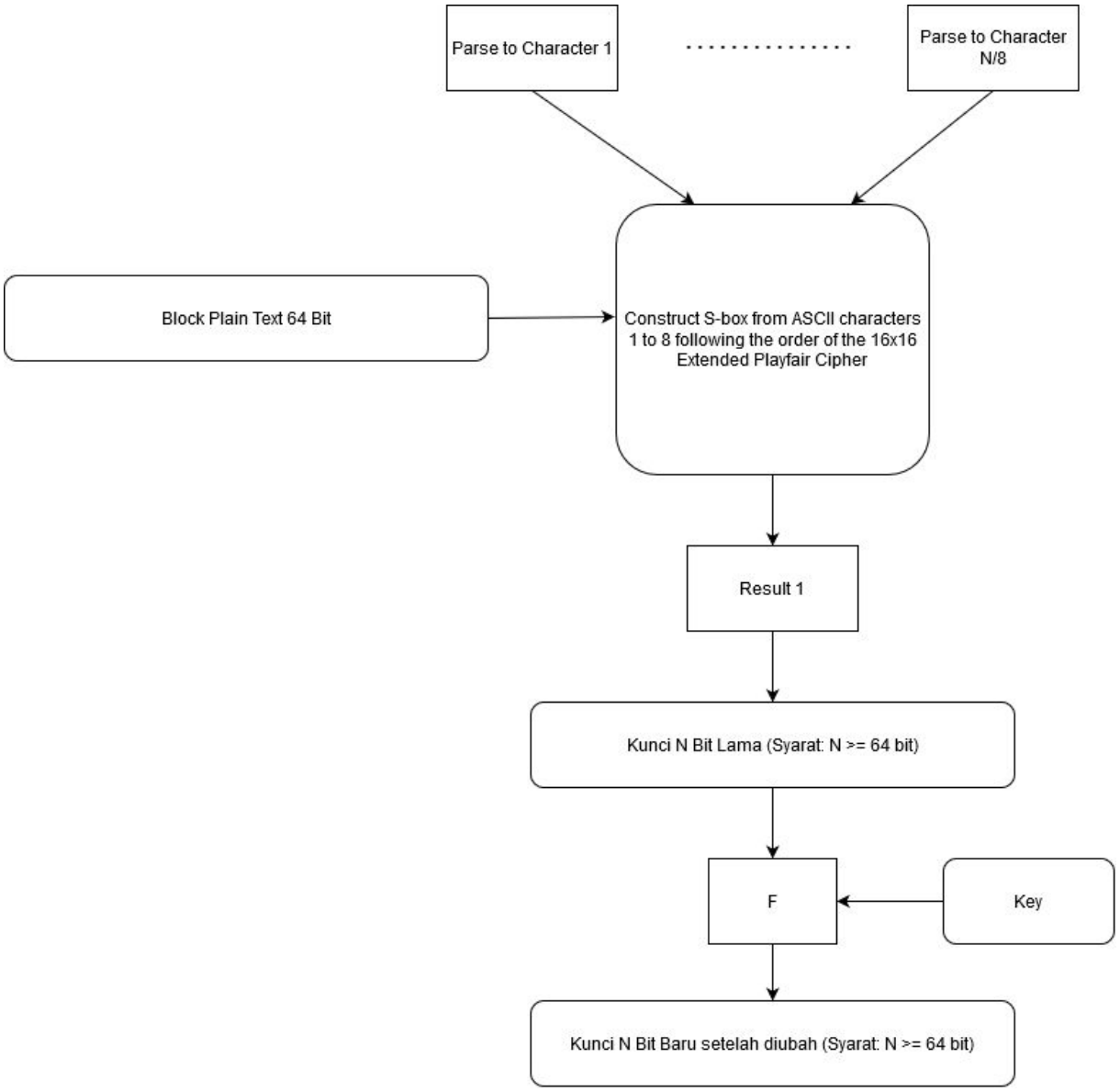


S-box lalu dihasilkan dengan mengisi S-box dengan karakter-karakter ASCII hasil konversi sesuai urutan tersebut. Mengacu pada gambar, maka pada kasus tersebut didapatkan bahwa blok 8 bit tersebut apabila dikodifikasikan menjadi ASCII akan mengeja Playfair.

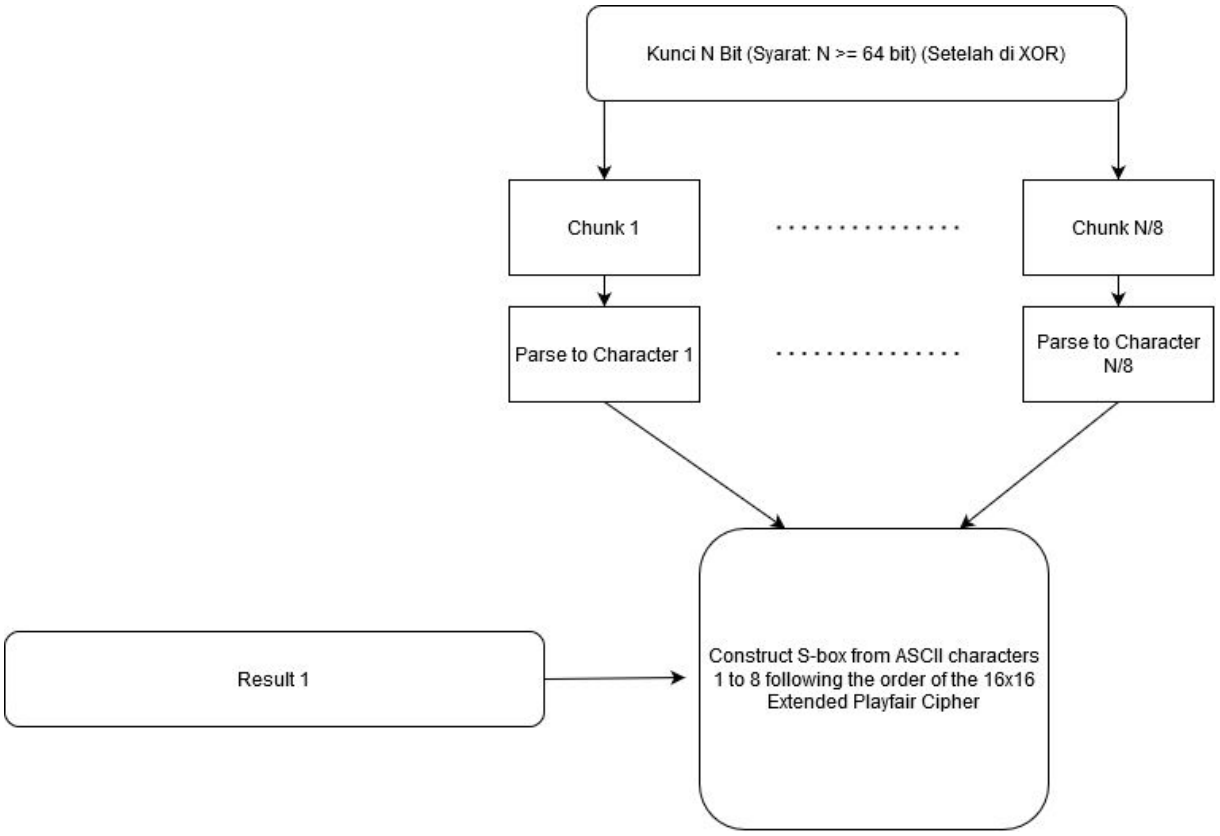
3.2. Konstruksi S-box Kedua

S-box kedua dikonstruksi dengan melakukan manipulasi pada kunci yang nantinya mengisi S-box kedua. Hal ini dilakukan dengan melakukan pergeseran-pergeseran pada S-box pertama berdasarkan nilai dari kunci tersebut. Karena kunci adalah blok yang berukuran 64 bit maka perubahan pada S-box akan dilakukan 8 kali. Peraturan untuk mengubah susunan S-box pertama adalah:

1. Apabila hasil modulo 2 dari nilai ASCII genap, maka dilakukan pergeseran baris ke bawah dan dilakukan secara siklis
2. Apabila hasil modulo 2 dari nilai ASCII ganjil, maka dilakukan pergeseran kolom kekanan dan dilakukan secara siklis



Hasil modifikasi tersebut lalu digunakan untuk membuat kotak Playfair yang baru sesuai dengan aturan Playfair 16x16.



Hasil cipherteks 1 lalu melewati S-box kedua dan menghasilkan cipherteks 2.

S-box ketiga adalah S-box statis sesuai dengan rekomendasi Rijndael, dilakukan substitusi hingga cipherteks 3 dapat dihasilkan.

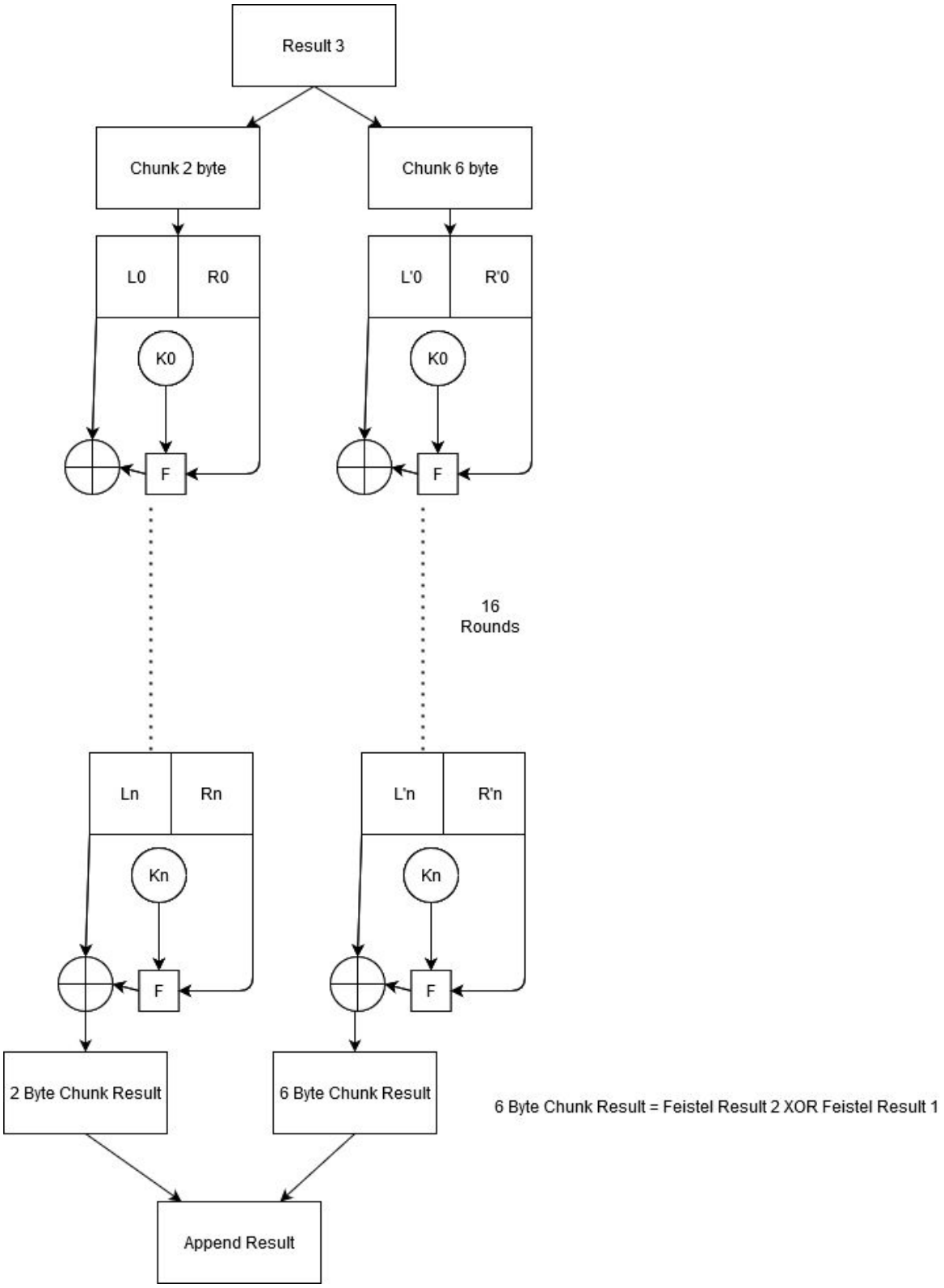
AES S-box

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

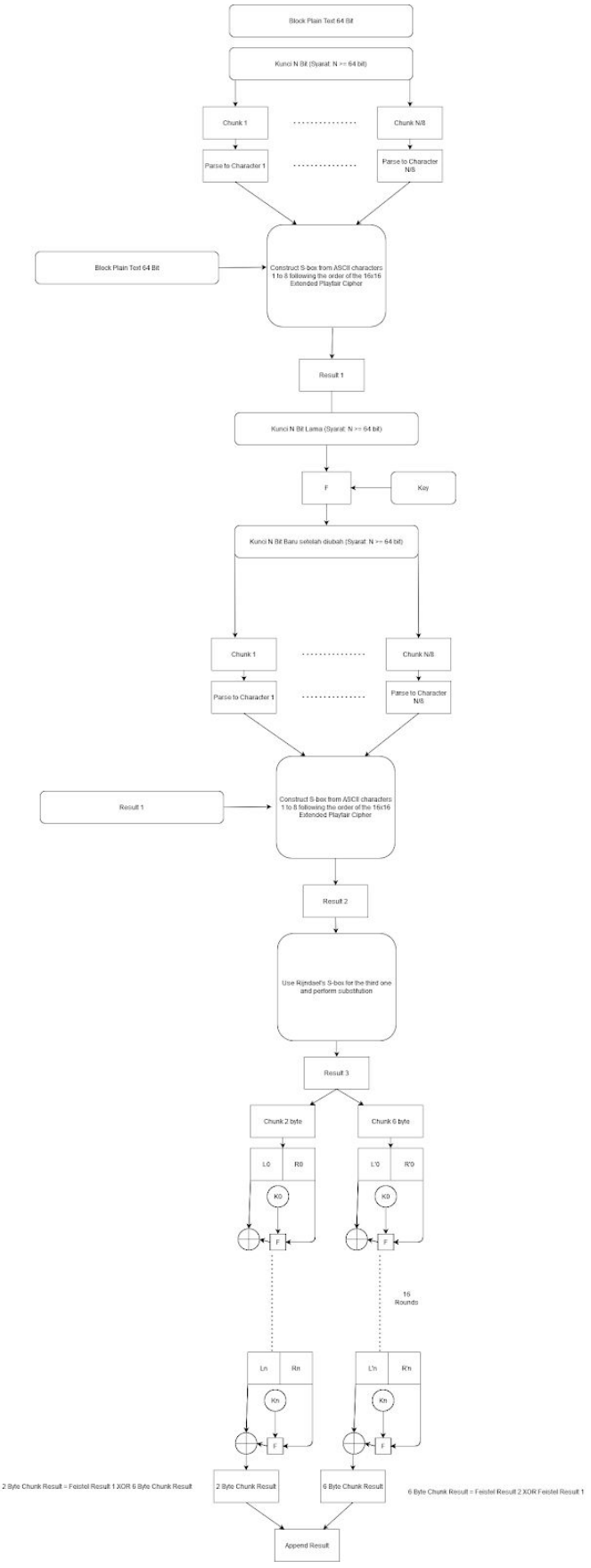
(Sumber: https://en.wikipedia.org/wiki/Rijndael_S-box)

3.3. Jaringan Feistel

Cipherteks 3 akan melewati jaringan Feistel. Jaringan Feistel ini akan mengambil 8 byte data dan akan digunakan modifikasi dari varian *Unbalanced Feistel*. Digunakan 2 jaringan Feistel, 2 byte data akan masuk ke jaringan Feistel 1 dan 6 byte akan masuk ke jaringan Feistel 2. Pembangkitan upa kunci dan fungsi menggunakan fungsi yang digunakan pada DES. Hasil dari kedua jaringan Feistel pertama akan diXOR dengan jaringan Feistel pertama dan sebaliknya. Teks akan disambungkan lalu dikembalikan.



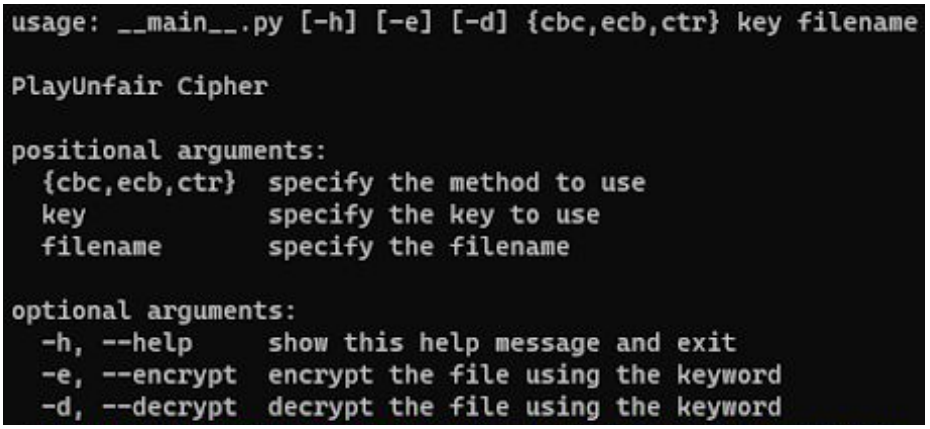
Sehingga diagram lengkap dari cipher adalah sebagai berikut:


Block Plain Text 64 Bit
Kunci N Bit (Syarat: N >= 64 bit)
Chunk 1
Chunk N/8
Parse to Character 1
Parse to Character N/8
Block Plain Text 64 Bit
Construct S-box from ASCII characters 1 to 8 following the order of the 16x16 Extended Playfair Cipher
Result 1
Kunci N Bit Lama (Syarat: N >= 64 bit)
F
Key
Kunci N Bit Baru setelah diubah (Syarat: N >= 64 bit)
Chunk 1
Chunk N/8
Parse to Character 1
Parse to Character N/8
Result 1
Construct S-box from ASCII characters 1 to 8 following the order of the 16x16 Extended Playfair Cipher
Result 2
Use Rijndael's S-box for the third one and perform substitution
Result 3
Chunk 2 byte
Chunk 6 byte
L0
R0
L'0
R'0
K0
K0
F
F
16 Rounds
Ln
Rn
L'n
R'n
Kn
Kn
F
F
2 Byte Chunk Result = Feistel Result 1 XOR 6 Byte Chunk Result
2 Byte Chunk Result
6 Byte Chunk Result
6 Byte Chunk Result = Feistel Result 2 XOR Feistel Result 1
Append Result

4. Hasil Eksperimen

4.1. Tampilan Program

Program yang dibuat berdasarkan metode enkripsi ini adalah sebuah program CLI yang menerima beberapa argumen. Adapun tampilan program adalah sebagai berikut.



Tampilan menu bantuan program

Sesuai dengan panduan yang ada pada menu bantuan program, maka program menerima beberapa argumen yaitu metode enkripsi yang digunakan (cbc, ecb, atau ctr), kunci yang akan digunakan, nama file yang ingin dienkripsi atau didekripsi, dan operasi yang ingin dilakukan (enkripsi atau dekripsi).

Berikut adalah contoh pemanggilan program yang dilakukan ketika ingin mengenkripsi suatu *file*.

```
python3 -m playunfair -e cbc KUNCI
test.txt
```

Contoh pemanggilan program melalui CLI untuk melakukan enkripsi

Setelah dilakukan pemanggilan program tersebut, maka program akan berjalan tanpa menampilkan output apapun pada layar, namun akan langsung menuliskan file hasil enkripsi dengan tambahan nama `.cipher` atau *file* hasil dekripsi dengan tambahan nama `.plain` pada bagian akhir nama *file* baru.

4.2. Eksperimen Pertama

Program menerima masukan input file dan melakukan enkripsi dengan algoritma. Kunci yang digunakan bersifat konstan yaitu adalah:

Kunci = [**255, 255, 255, 1, 13, 69, 210, 155**]

Berikut adalah contoh enkripsi menggunakan file text yang berisi:

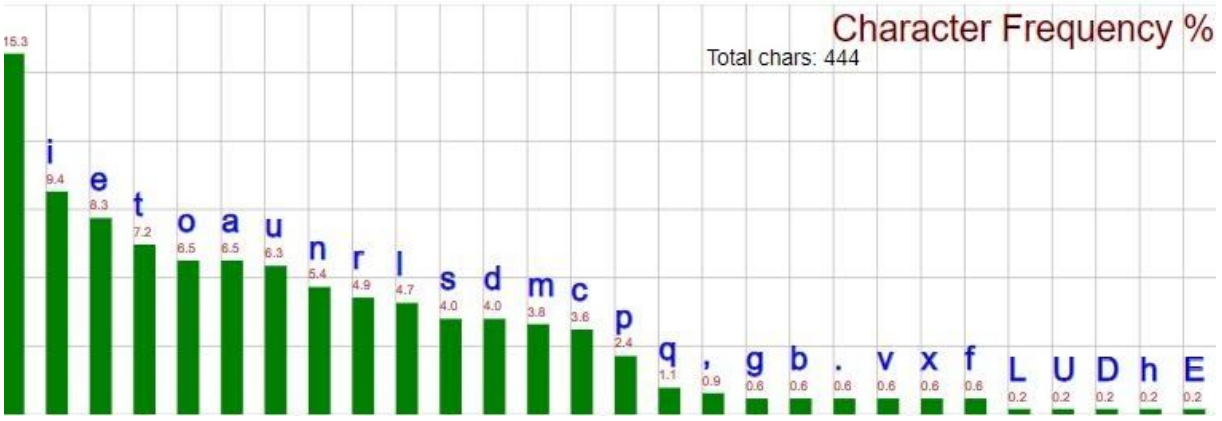
“**Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in**

voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum”

Lalu data dienkripsi menjadi:

“¤Ã·¯'*ãžx•n±ÝäÃBÐîlce5Îž|ü,"Ò"ÅËÿ&úÄDÛü4}xh·O¨mx/ÿòQ,ô?Ö’b_SÛû¸{ö4toïH-îàuÊÀý³
Ô?¿+^ÄÎ}¸ÛHç´Þ*ªn‘r´õ¶ö…€
çP¯vR;o§ÉÇçFÉñ¸p*¼P;C€(ivEA–Ç•*ƒ*@ûmìà#„ÄDz¹t[ò¢u·dE*ƒ*àø=u
Ü7›*¬™ÀÙŸF„—·‘…AÞÀÅXpëiúà,óý)×2pÝ
JqÓDc*åd„ªâ6¼ËÅuW¾¯·¤“\^^Ð¤Ç”G¯x/×ðh¸ë_#
&n!ií$Š"Ç<é
4Š¹Ç¿(×†Âðýcìœš U“š;{™_Ú.óÊ…{öHR“…ÞöF}œlEC9y™c+)¨ŸC«H*ƒ*ûUÙ/àµî¹9ì€ïèi«:ñÒš4¯Vî‡
Ì{ÖÀøõ°¿«æœ½žl”ú£¤®™²=ïÐnKjûÀó¢Ø”’çIç³
<{sj£‰1osŸÿ‰ñ`½ñ×i,‰”

Berikut adalah perbandingan histogram plainteks dan teks yang sudah melewati cipher:



Plain text

Cipher text

4.3. Eksperimen dengan Mengubah Bit pada Teks Asli

Kami lalu menguji dengan mengubah huruf L pada awal kata menjadi M:

“**Morem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum**”

Hasil data yang sudah dienkripsi menjadi:

“**Þ¸·¯'*ãž^fn±ÝäÃB,Ølce5çY|ü,"Òö`Ëÿ&úÄ„‰ü4}xh—ÜO¨mx/¾mQ,ô?Ö'„¸SÛû¸{·¾toïHËÉàu**
ÊÀhåÔ?¿+^þ¾Î}¸ÛHçÁs*ªnÒs´õ¶ö…€¿fP¯vR;coÉÇçFÉáp*¼P;»(ivEAãSÇ•*ƒ*@ûmùk#„ÄDÍCt[ò
¿%ØdE*ƒà*ø=,Ü7›*¬™Þ²ŸF„—·Ï}AÞÀÅX:iúà,óýÀ×2pÝ
W¿ÓDc*ådúâ6¼ËÅ!!W¾¯·¤Y²^^Ð¤Ç”Ìÿx/×ðhbÝ_#
&n!"Yí$Š"Ç<®1
4Š¹Ç¿ô&†Âðýcì È)U“š;{,à™_Ú.\…{öH…ÞöF}œ=ÑEC9y™Aª)¨ŸC«HC)ûUÙ/àµ$9ì€ïèiEñÒš4¯¤l‡
Ì{ÖÀ¿]õ°¿«µ>½žl”vI¤®™²µÐnÿûÀó¿Øå]çIçª
<¾j£‰ŸosŸÿ‰ñ^ñ×i,‰”

Berikut adalah perbandingan histogram plainteks dan teks yang sudah melewati cipher:

Plain text

Cipher text

4.4. *Eksperimen dengan Mengubah Bit pada Kunci*

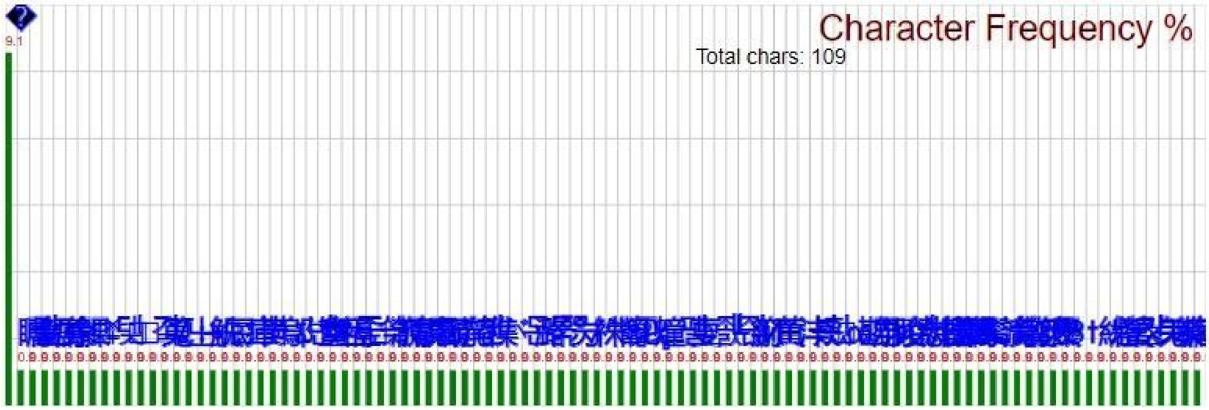
Lalu kami membandingkan hasil apabila kunci diubah 1 bit. 8 bit pertama dengan representasi **11111111** kami ubah menjadi **11111110.** Teks yang digunakan sama seperti teks awal:

Kunci = [**254, 255, 255, 1, 13, 69, 210, 155**]

Hasil Enkripsi:

**“矚킬鯃癘余髁□□텟طي□걱□7□□□底□□봦ᢌ鄎□반畬坦□??줖?□쑛□鳹靚索顯硴滗첸穙バ뭔
潞뤎□峟✓祩□□□ஐ熣?_펠叓ȩ춼╙잕□И黄ĩ沣焸쵔□ɱ□毘洞畯샚憇焨槿犨覉鬃□?潊鐓□筴□Ȣ
□絖砦?蘁킷□羗喋□□駣□ﺑ□鏁헠젰蹬?搉킹잫豕佫ʔ릨ˁd洌h□鷖❗澈?□ȥ췻□歖韭锛
삷앷□ⓧᏋ□奇ɷ槑□瓢□滋徿倜퉂□睡炌屨ч옹□□□鄰猾∈廟□□馹Ṇ칑ჲ쐉矾?□فخم빅ɭ네畲괃榅
퓔돠□欣뗌?뗵출～䆏쭷軮뒣철嘸蘢쩩澛**”

Berikut adalah hasil histogram dari cipher text:



4.5. *Perhitungan Avalanche Effect*

Avalanche Effect adalah salah satu faktor yang diinginkan pada pembuatan block cipher. Pada dasarnya *Avalanche Effect* adalah fenomena yang terjadi apabila ada 1 bit yang diubah pada plainteks atau kunci maka akan menghasilkan perubahan yang signifikan pada hasil enkripsi [5]. Rumus untuk mendapatkan nilai *Avalanche Effect* adalah:

$$Avalanche\ Effect = \frac{Number\ of\ Changed\ bit\ in\ Ciphertext}{Number\ of\ bits\ in\ Ciphertext}$$

Berikut adalah tabel perubahan bit dan nilai *Avalanche Effect*:

	Jumlah bit yang berubah	Jumlah bit	Nilai Avalanche
Cipher 1	1882	3552	0.52984234234

Cipher 2	1893	3552	0.53293918918
Cipher 3	1918	3552	0.53997747747

Jika dibandingkan maka variasi perubahan bit yang dihasilkan adalah 1918 walaupun kunci hanya diubah 1 bit.

5. Analisis

Analisis yang dilakukan berdasarkan hasil dari distribusi frekuensi dan nilai *Avalanche Effect*.

Hal yang menarik dapat diamati pada distribusi frekuensi cipherteks pertama dan kedua. Cipherteks pertama dan kedua apabila diamati menghasilkan perbandingan frekuensi yang sama, namun karakter-karakter yang menempati tempat tersebut berbeda jauh di antara keduanya. Sedangkan perubahan 1 bit saja pada bit kunci menyebabkan perubahan baik pada distribusi dan pada karakter yang mengisi distribusi tersebut. Hal ini menunjukkan ketergantungan yang tinggi dan kompleks antara teks dan kunci, sehingga memenuhi prinsip *confusion* yang diutarakan oleh Shannon.

Shannon juga mengatakan bahwa algoritma enkripsi harus memenuhi prinsip diffusion. Shannon mendefinisikan bahwa *diffusion* berarti adalah menyembunyikan hubungan cipherteks dan plainteks, pada percobaan, hal ini digambarkan dengan penilaian *Avalanche Effect*. Dengan melihat berapa bit yang berubah apabila cipherteks dibandingkan dengan plainteks. Karena nilai ideal yang digunakan untuk mengukur Avalanche Effect adalah apabila berada di atas 50% [6]. Maka algoritma ini dapat dikatakan memiliki Avalanche Effect yang baik karena menghasilkan angka rata-rata berdasarkan 3 kali percobaan pada kisaran 53-54%

6. Kesimpulan dan Saran

Teknik kriptografi diperlukan sebagai cara untuk melindungi data-data penting agar tidak diketahui oleh orang pada umumnya. Pada jurnal ini dibuktikan bahwa S-box tidak perlu memiliki unsur statis dan dapat memanfaatkan S-box yang bersifat dinamis. Hasil yang didapatkan juga masih memenuhi prinsip diffusion dan confusion Shannon, dibuktikan dengan distribusi frekuensi yang menunjukkan ketergantungan tinggi antara cipherteks dan kunci dan adanya perubahan drastis dari cipherteks meskipun hanya ada 1 bit yang berubah.

Saran yang direkomendasikan adalah untuk menginvestigasi kemungkinan kemunculannya sifat one-time pad yang dapat dimunculkan apabila S-box yang dibangkitkan bergantung dengan hasil S-box pertama. Hasil ini menunjukkan perlunya kunci dekripsi yang lebih besar daripada kunci enkripsi.

7. Daftar Pustaka

[1] William Stallings. 2010. Cryptography and Network Security: Principles and Practice (5th. ed.). Prentice Hall Press, USA

[2] Munir, Rinaldi. "Kriptografi Kunci-Publik." IF4020 Kriptografi. 23 Oktober 2020, Institut Teknologi Bandung. Kuliah Tatap Muka

[3] Munir, Rinaldi. "Kriptografi Klasik." IF4020 Kriptografi. 2 September 2020, Institut Teknologi Bandung. Kuliah Tatap Muka

[4] Ss, Dhenakaran & .M, Ilayaraja. (2012). Extension of Playfair Cipher using 16X16 Matrix. International Journal of Computer Applications. 48. 37-41. 10.5120/7363-0192.

[5] Mahindrakar, Manisha. (2014). Evaluation of Blowfish Algorithm based on Avalanche Effect. International Journal of Innovations in Engineering and Technology

[6] Webster, A. F.; Tavares, Stafford E. (1985). "On the design of S-boxes". Advances in Cryptology - Crypto '85. Lecture Notes in Computer Science. 218. New York, NY: Springer-Verlag New York, Inc. pp. 523–534