Algoritma 474

Husnulzaki Wibisono Haryadi-13515005[1] Aya Aurora Rimbamorani-13515098[2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13515005@std.stei.itb.ac.id [2]13515098@std.stei.itb.ac.id

Abstrak—Algoritma 474 merupakan algoritma *block cipher* yang dapat mengenkripsi pesan dengan ukuran blok sebesar 128 bit. Besar ukuran blok kunci yang diperlukan adalah 128 bit yang kemudian dari kunci tersebut aka n terbentuk 32 kunci internal yang digunakan dalam proses enkripsi yang dilakukan sebanyak 32 kali putaran. Pada algoritma ini pula dimanfaatkan beberapa operasi bit yaitu XOR, *shifting*, *inverse* bit, dan substitusi. Selain operasi bit, diterapkan pula operasi matematika dasar yaitu operasi modulo yang berperan dalam menentukan operasi bit yang akan dijalankan. Operasi bit tersebut dilakukan guna mempersulit penyerang dan kriptanalis dalam memecahkan *ciphertext*.

Kata Kunci—Operasi Bit, Jaringan *Feistel*, P-Box Algoritma DES Termodifikasi, dan S-Box Algoritma AES

I. PENDAHULUAN

Kriptografi merupakan cabang ilmu yang mempelajari teknik dan seni dalam menjaga keamanan suatu pesan. Kriptografi ini pula sudah diterapkan dari zaman sebelum masehi atau jauh sebelum adanya komputer digital (kriptografi klasik). Pada zaman tersebut, teknik kriptografi yang digunakan cukup sederhana yaitu mengganti suatu huruf dengan huruf lainnya (algoritma *Caesar Cipher*) [1]. Algoritma kriptografi yang lahir pada zaman sebelum adanya komputer digital tersebut merupakan algoritma yang komputasinya berbasis alfabet. Beberapa teknik yang diterapkan pada kriptografi klasik antara lain substitusi dan transposisi.

Seiring berkembangnya teknologi dan pengetahuan, berbagai algoritma kriptografi baru pun tercipta. Berbeda dengan kriptografi klasik, algoritma pada zaman modern atau setelah adanya komputer digital (kriptografi modern), komputasi yang dilakukan berbasis bit. Pada kriptografi modern, teknik substitusi dan transposisi masih digunakan namun dengan tingkat kerumitan yang lebih tinggi. Komputasi berbasis bit pada kriptografi modern ini terbagi menjadi dua, yaitu *Stream Cipher* [2] dan *Block Cipher* [2]. *Stream Cipher* merupakan algoritma yang melakukan operasi bit pada bit tunggal dan melakukan enkripsi/dekripsi bit per bit. Berbeda dengan *Stream Cipher, Block Cipher* melakukan operasi pada blok bit (misal 128bit/blok) dan melakukan enkripsi/dekripsi blok per blok.

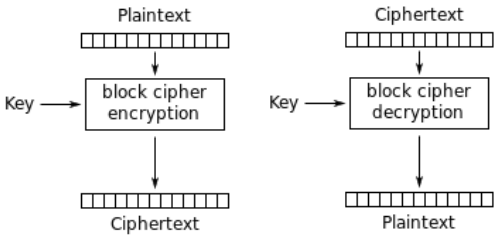
Algoritma 474 termasuk ke dalam golongan algoritma *Block Cipher* (128bit/blok) yang dikembangkan dengan mengkombinasikan beberapa teknik operasi bit dan teknik pada algoritma *Block Cipher* lain yang sudah ada. Algoritma 474 ini juga menerapkan prinsip *confusion* dan *diffusion* dari Shannon [3]. Prinsip *confusion* diterapkan pada teknik operasi bit substitusi dengan menggunakan P-Box pada algoritma DES dan S-Box pada algoritma AES. Sedangkan untuk prinsip *diffusion,* penerapannya dilakukan pada operasi bit *shifting,* dan *inverse bit.* Selain operasi bit, algoritma 474 juga menerapkan jaringan *Feistel* yang digunakan pada saat enkripsi/dekripsi. Jaringan *Feistel* ini dijalankan sebanyak 32 kali. Pada algoritma 474 ini pula, kunci masukan (128 bit) akan diproses untuk mendapatkan 32 kunci internal (masing-masing 64 bit). Kunci internal tersebut kemudian digunakan pada fungsi f jaringan *Feistel*.

II. DASAR TEORI

A. *Mode Operasi Block Cipher*

1) *Electronic Code Book* (ECB)

Pada mode ECB, setiap blok *plaintext Pi* dienkripsi menjadi blok *ciphertext Ci*. Enkripsi setiap blok tersebut dilakukan secara individual tanpa memperhatikan blok maupun bit *plaintext* sebelum atau sesudah *plaintext Pi* tersebut. Oleh karena itu, untuk setiap blok *plaintext Pi* yang sama akan dienkripsi menjadi blok *ciphertext Ci* yang sama pula (gambar 1).

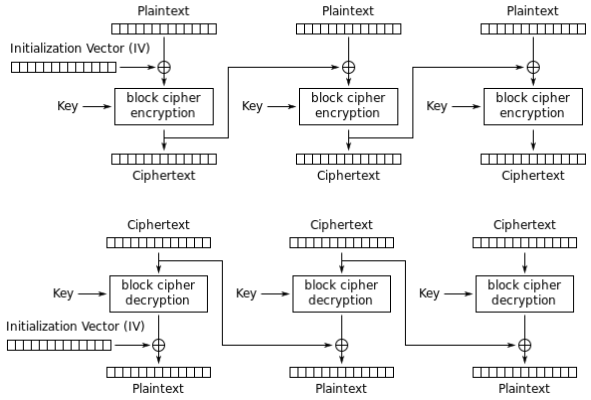


Gambar 1. Diagram ECB

2) *Cipher Block Chaining* (CBC)

Berbeda dengan ECB, pada mode CBC setiap blok *ciphertext Ci* bergantung pada *plaintext Pi* dan *plaintext Pi-1* (*plaintext* sebelumnya). Pada proses enkripsi mode CBC, *ciphertext Ci* diperoleh dengan mengenkripsi hasil operasi XOR antara *ciphertext Ci-1* dan *plaintext Pi*. Pada penanganan blok *plaintext P1* (blok *plaintext* pertama), sebuah *initial*

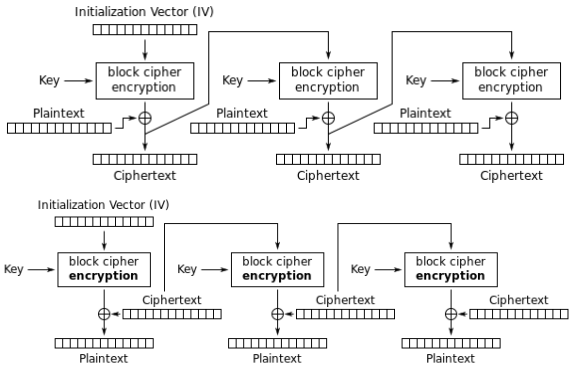
vector (IV) digunakan dalam operasi XOR dilakukan inisialisasi vektor sejumlah panjang blok untuk kemudian di-XOR-kan dengan *plaintext P1* dan dienkripsi untuk menghasilkan *ciphertext C1* (gambar 2).



Gambar 2. Diagram CBC

3) *Cipher Feedback* (CFB)

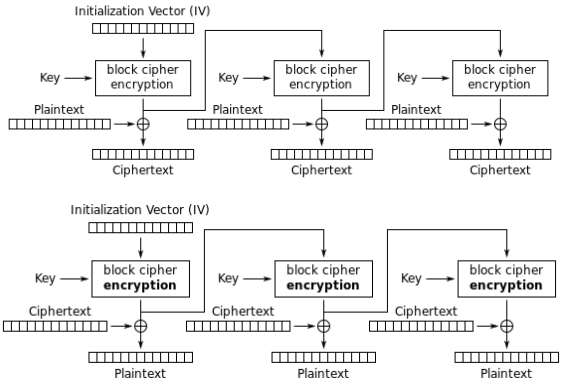
Mode CFB melakukan enkripsi tidak pada blok *plaintext*, melainkan pada blok antrian berupa *initial vector* (IV). Blok antrian tersebut pertama dienkripsi dengan kunci tertentu, dan *left-most byte* dari hasil enkripsi tersebut kemudian digunakan dalam operasi XOR dengan n bit *plaintext* untuk menghasilkan n bit *ciphertext*. Operasi *shifting* sebesar n bit kemudian dilakukan terhadap blok antrian yang belum dienkripsi. Setelah itu, hasil n bit *ciphertext* disambungkan ke bagian kanan blok antrian untuk mendapatkan blok antrian yang baru (gambar 3).



Gambar 3. Diagram CFB

4) *Output Feedback* (OFB)

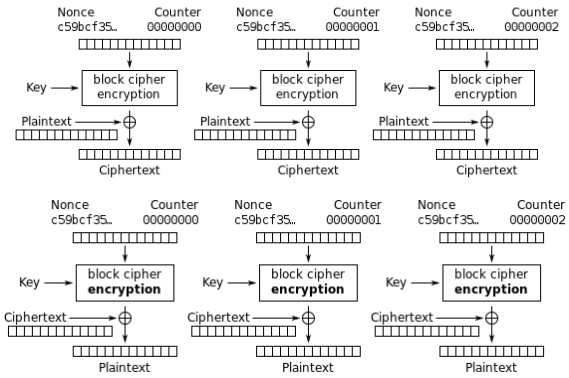
Menyerupai mode CFB, mode OFB juga tidak melakukan enkripsi terhadap blok *plaintext.* Mode OFB ini memiliki mekanisme yang sama dengan mode CFB hanya saja terdapat perbedaan pada saat penyambungan n bit pada blok antrian untuk menghasilkan blok antrian baru. Pada mode OFB, n bit yang disambungkan adalah *left-most byte* dari hasil enkripsi blok antrian dengan kunci tertentu (gambar 4).



Gambar 4. Diagram OFB

5) *Counter*

Pada mode *counter*, blok yang dienkripsi bukanlah blok *plaintext* melainkan blok *counter*. Blok *counter* ini diinisialisasi terlebih dahulu dengan *initial vector* (IV). Hasil dari enkripsi blok *counter* tersebut kemudian dilakukan operasi XOR dengan blok *plaintext* untuk mendapatkan blok *ciphertext.* Pada proses selanjutnya, untuk enkripsi blok-blok berikutnya, nilai blok *counter* dinaikkan nilainya satu (gambar 5).



Gambar 5. Diagram Counter

B. *Algoritma DES*

Algoritma DES [4] merupakan algoritma *block cipher* yang beroperasi dalam ukuran blok 64 bit. Pada algoritma DES, panjang kunci yang digunakan adalah 64 bit, namun hanya 56 bit yang dipakai. Setiap blok dienkripsi dalam 16 putaran dan pada setiap putaran diperlukan kunci internal yang berbeda. Kunci internal ini dibangkitkan dari kunci eksternal. Sebelum dienkripsi, pada setiap blok akan dilakukan permutasi awal dengan menggunakan P-Box (gambar 6).

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Gambar 6. P-Box pada Algoritma DES

C. *Algoritma AES*

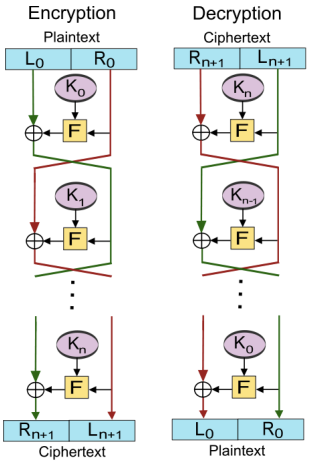
Algoritma AES [5] terlahir karena algoritma DES sudah dianggap tidak aman lagi untuk digunakan. Pada AES panjang kunci memiliki 3 varian, yaitu 128 bit, 192 bit, dan 256 bit. Algoritma AES beroperasi dalam orientasi *byte* yang pada setiap putaran enkripsinya diperlukan operasi permutasi dan substitusi. Banyak putaran yang dilakukan pada algoritma AES adalah 16 kali. Terdapat 4 operasi yang dilakukan pada putaran enkripsi yaitu substitusi *byte* dengan menggunakan tabel substitusi S-Box (gambar 7), pergeseran *array state* baris secara *wrapping*, pengacakan data pada masing-masing kolom *array state*, dan melakukan operasi XOR antara *state* sekarang dengan *round key*.

	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

Gambar 7. S-Box pada Algoritma AES

D. *Jaringan Feistel*

Jaringan *Feistel* banyak diterapkan di berbagai algoritma *block cipher* dikarenakan model jaringan *Feistel* ini bersifat *reversible* untuk proses enkripsi dan dekripsi (gambar 8). Sifat *reversible* tersebut memudahkan perancang *block cipher* untuk tidak perlu merancang algoritma baru untuk melakukan dekripsi *ciphertext* kembali menjadi *plaintext.*



Gambar 8. Jaringan *Feistel*

III. RANCANGAN ALGORITMA

Algoritma 474 melakukan enkripsi dengan menerima masukan blok *plaintext* dan blok kunci eksternal dengan 1 blok terdiri dari 128 bit. Algoritma ini memanfaatkan jaringan *Feistel* dan melakukan sebanyak 32 kali putaran untuk melakukan enkripsi/dekripsi. Sebelum memasuki jaringan *Feistel,* elemen bit pada blok yang akan dienkripsi dipermutasi terlebih dahulu (*initial permutation*) dengan menggunakan P-Box pada algoritma DES yang dimodifikasi. Pada algoritma ini pula terdapat fungsi f (*round function*) dan pembangkitan kunci internal yang memanfaatkan operasi bit berupa substitusi dengan S-Box pada algoritma AES, *shifting,* dan *inverse bit.*

A. *Algoritma Secara Keseluruhan*

Proses pertama yang harus dilakukan dalam menjalankan algoritma 474 adalah menerima kunci eksternal sepanjang 128 bit untuk mendapatkan 32 kunci internal dengan masing-masing kunci internal sepanjang 64 bit. Kunci internal tersebut kemudian dibagi menjadi kunci ganjil dan kunci genap dimana kunci ganjil digunakan pada putaran ganjil dan kunci genap digunakan pada putaran genap.

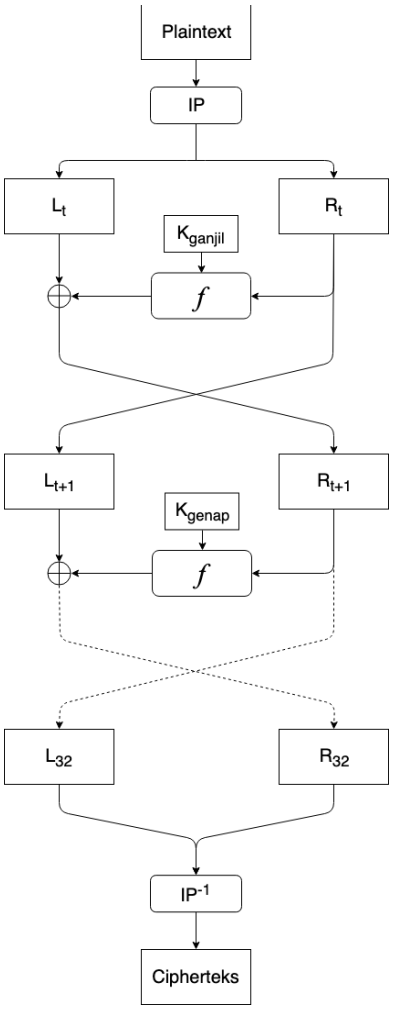
Setelah didapatkan kunci internal, blok masukan sepanjang 128 bit dipermutasi terlebih dahulu (*initial permutation*) dengan menggunakan P-Box pada algoritma DES yang dimodifikasi (gambar 9). Setelah itu, blok hasil permutasi tersebut dibagi menjadi dua, yaitu blok kiri (*Left Block*) yaitu elemen pada index pertama sampai dengan ke-64 dan blok kanan (*Right Block*) yaitu elemen pada index ke-65 sampai dengan index ke-128. Pada tahap selanjutnya, kedua pasang blok tersebut akan masuk ke jaringan *Feistel* yang dilakukan sebanyak 32 kali (gambar 10).

Pada setiap putaran jaringan *Feistel,* hanya blok kanan saja yang diproses dengan fungsi f dan menerima masukan kunci internal. Terdapat 3 operasi bit pada proses pembangkitan kunci internal dan fungsi f yaitu *shifting*, *inverse* bit, dan substitusi dengan S-Box pada algoritma AES (gambar 7). Sebelum melakukan 3 operasi bit tersebut, bit-bit pada blok yang akan diproses dibagi menjadi elemen blok yang terdiri

dari 8 bit. Pada operasi *shifting*, elemen pada blok akan digeser ke kiri sejumlah n tertentu. Berbeda dengan *shifting* pada operasi bit biasanya, sejumlah n bit paling kiri pada elemen dipindahkan ke sebelah kanan elemen agar pada setiap elemen blok nilainya tetap terjaga dari 0-255 (rentang nilai *integer* karakter ASCII). Tahap terakhir yang perlu dilakukan setelah 32 kali putaran jaringan *Feistel* selesai adalah melakukan kebalikan permutasi awal (*inverse initial permutation*) dengan P-Box pada algoritma DES yang dimodifikasi (gambar 9).

```
57, 49, 41, 33, 25, 17, 9,  1, 121, 113, 105,  97, 89, 81, 73, 65,
61, 53, 45, 37, 29, 21, 13, 5, 125, 117, 109, 101, 93, 85, 77, 69,
56, 48, 40, 32, 24, 16,  8, 0, 120, 112, 104,  96, 88, 80, 72, 64,
60, 52, 44, 36, 28, 20, 12, 4, 124, 116, 108, 100, 92, 84, 76, 68,
55, 47, 39, 31, 23, 15, 7, 127, 119, 111, 103, 95, 87, 79, 71, 63,
59, 51, 43, 35, 27, 19, 11, 3, 123, 115, 107,  99, 91, 83, 75, 67,
54, 46, 38, 30, 22, 14, 6, 126, 118, 110, 102, 94, 86, 78, 70, 62,
58, 50, 42, 34, 26, 18, 10, 2,  122, 114, 106, 98, 90, 82, 74, 66
```

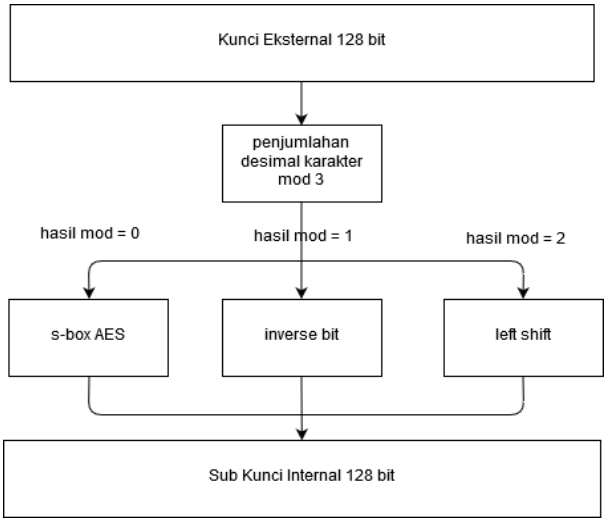
Gambar 9. P-Box Algoritma 474



Gambar 10. Alur Algoritma 474

B. *Pembangkitan Kunci Internal*

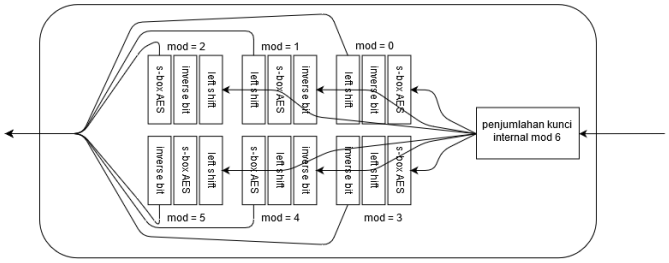
Pada proses pembangkitan kunci internal, hal pertama yang dilakukan adalah membagi blok kunci eksternal menjadi 16 elemen blok dengan masing-masing elemen terdiri dari 8 bit. Pada setiap elemen tersebut didapatkan nilai *integer*-nya dan dilakukan penjumlahan untuk mendapatkan total nilai *integer* dari seluruh elemen pada blok. Pada tahap selanjutnya, total nilai elemen blok tersebut dibagi dengan 3 dan sisa pembagiannya akan menentukan operasi bit apa yang akan dilakukan. Jika sisa pembagian total nilai *integer* elemen blok adalah 0, maka operasi yang dilakukan adalah substitusi elemen dengan S-Box pada algoritma AES, jika sisa baginya adalah 1, maka operasi yang dilakukan adalah *inverse* bit, dan jika sisa baginya adalah 2, maka operasi yang dilakukan adalah *shifting* (gambar 11).



Gambar 11. Alur Pembangkitan Kunci Internal

C. *Fungsi F* (*Round Function*)

Fungsi F atau *round function* merupakan fungsi yang dijalankan pada saat proses enkripsi/dekripsi di dalam jaringan *Feistel.* Pada fungsi f ini, ketiga operasi bit yaitu *shifting, inverse* bit, dan substitusi dengan S-Box pada algoritma AES dilakukan. Berbeda dengan pembangkitan kunci internal, ketiga operasi bit tersebut dijalankan seluruhnya namun dengan urutan yang berbeda-beda. Pada setiap blok yang akan diproses dengan fungsi f, diperlukan satu kunci internal yang berperan dalam menentukan urutan ketiga operasi bit tersebut. Satu blok kunci internal tersebut pertama dibagi menjadi beberapa elemen blok dengan satu elemen terdiri dari 8 bit. Pada setiap elemen tersebut kemudian didapatkan nilai *integer*-nya dan seluruh elemen dijumlahkan untuk diperoleh total nilai *integer* pada blok tersebut. Total nilai *integer* tersebut kemudian dibagi 6 (karena ada 6 variasi urutan ketiga operasi bit) dan sisa baginya dijadikan penentu urutan ketiga operasi bit yang dijalankan (gambar 5).



Gambar 12. Fungsi F (*Round Function*)

IV. EKSPERIMEN DAN ANALISIS

A. *Eksperimen*

Algoritma 474 diimplementasikan menggunakan bahasa Python 3.6. Implementasi dilakukan untuk mode blok cipher CBC, CFB, Counter, ECB, dan OFB. Eksperimen lalu dilaksanakan dengan melakukan dua kali eksperimen terhadap setiap jenis blok cipher. Eksperimen pertama menggunakan *plaintext* berukuran kecil (238 karakter) sedangkan eksperimen kedua menggunakan *plaintext* berukuran besar (2935 karakter). Kedua percobaan menggunakan kunci berukuran 128-bit yang sama, dimana kunci dibangkitkan secara acak dengan menggunakan *library NumPy*. Pada eksperimen ini algoritma hanya akan membaca masukan dengan format teks. Kunci enkripsi eksperimen dapat dilihat pada Tabel 1, sedangkan *plaintext* yang digunakan dapat dilihat pada Lampiran A dan Lampiran B.

Kunci enkripsi 128-bit
0x0 0x29 0x9a 0xf3 0xaf 0xac 0x88 0x4c 0xad 0xe0 0x4b 0xc2 0x28 0xd7 0x69 0x1

Tabel 1. Kunci enkripsi yang digunakan pada eksperimen

B. *Analisis Kebenaran Hasil*

Untuk memvalidasi algoritma 474, dilakukan pengujian dengan menggunakan *plaintext* pada Lampiran A yang berukuran 42 karakter (ditampilkan pada Tabel 3). Dari hasil pengujian pada Tabel 5 dapat terlihat bahwa algoritma berjalan dengan baik pada semua jenis blok cipher. Hasil enkripsi sendiri dapat dilihat di Tabel 4.

Plaintext	all work and no play makes jack a dull boy
Heksadesimal	0x61 0x6c 0x6c 0x20 0x77 0x6f 0x72 0x6b 0x20 0x61 0x6e 0x64 0x20 0x6e 0x6f 0x20 0x70 0x6c 0x61 0x79 0x20 0x6d 0x61 0x6b 0x65 0x73 0x20 0x6a 0x61 0x63 0x6b 0x20 0x61 0x20 0x64 0x75
	0x6c 0x6c 0x20 0x62 0x6f 0x79

Tabel 2. *Plaintext* untuk uji kebenaran hasil

Jenis	*Ciphertext*
CBC	0x98 0xc0 0xb8 0xf 0x12 0xd8 0xe3 0x4e 0x25 0x26 0xa0 0x41 0x33 0x4e 0x37 0xbc 0x84 0xfe 0x83 0xc9 0x92 0x76 0x27 0x77 0x48 0x4a 0x74 0x21 0x12 0x25 0xef 0xde 0x1f 0xd4 0x22 0x49 0x3d 0x9d 0x12 0xfc 0x60 0xb0 0x25 0xcd 0xee 0x17 0xf1 0xe
CFB	0x56 0xa8 0x6d 0x21 0xf5 0xc4 0x1b 0xdc 0x36 0xba 0x35 0xd7 0xc7 0x11 0xed 0xe8 0xe7 0x50 0x47 0xb5 0x21 0xf3 0x41 0x31 0xac 0xd7 0xe4 0x8d 0x5b 0xee 0x13 0xc9 0xd6 0x65 0xb 0x4e 0x98 0xfa 0x3 0xad 0x40 0x8f
Counter	0x4c 0x2c 0x91 0x0 0x72 0x5b 0xca 0xf6 0x14 0xb3 0xa3 0x61 0x73 0x29 0xc1 0xcf 0x1d 0x2d 0xdc 0x58 0x65 0x18 0xd8 0xf6 0x51 0xa1 0xed 0x6f 0x32 0x24 0xc5 0xcf 0xce 0x60 0x9b 0x55 0xe9 0x5a 0x98 0x7f 0x47 0xbf 0xc9 0xa 0x73 0x47 0x86 0xcf
ECB	0xbc 0x2 0x18 0x26 0x4a 0x3e 0x45 0xaa 0x97 0xbf 0x2e 0xc8 0xe6 0xb1 0xfa 0x68 0xfa 0xba 0x8d 0xdb 0x3a 0x9b 0x69 0xd9 0x43 0xdb 0x73 0x3e 0xa4 0x25 0x4b 0xb 0x86 0x92 0x80 0xef 0xf 0xab 0x48 0xd5 0xa2 0x3a 0x31 0xbd 0xa7 0xe5 0xa 0x63
OFB	0xc7 0xfd 0x8c 0xf7 0xdc 0x8 0x4f 0xea 0x9b 0xe5 0x9a 0x3d 0xff 0x60 0xae 0x90 0x86 0x1c 0x49 0x69 0x8 0x26 0xf2 0x97 0xa0 0x8d 0x2 0x39 0x57 0x15 0x34 0x9e 0xd7 0xb0 0xd3 0x38 0xb7 0x78 0x59 0x2f 0x5e 0x97

Tabel 4. *Ciphertext* hasil uji kebenaran hasil

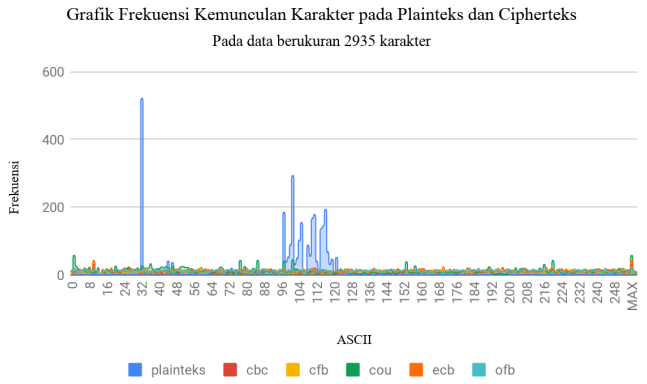
Jenis	*Plaintext*
CBC	0x61 0x6c 0x6c 0x20 0x77 0x6f 0x72 0x6b 0x20 0x61 0x6e 0x64 0x20 0x6e 0x6f 0x20 0x70 0x6c 0x61 0x79 0x20 0x6d 0x61 0x6b 0x65 0x73 0x20 0x6a 0x61 0x63 0x6b 0x20 0x61 0x20 0x64 0x75 0x6c 0x6c 0x20 0x62 0x6f 0x79
CFB	0x61 0x6c 0x6c 0x20 0x77 0x6f 0x72 0x6b 0x20 0x61 0x6e 0x64 0x20 0x6e 0x6f 0x20 0x70 0x6c 0x61 0x79 0x20 0x6d 0x61 0x6b 0x65 0x73 0x20 0x6a 0x61 0x63 0x6b 0x20

	0x61 0x20 0x64 0x75 0x6c 0x6c 0x20 0x62 0x6f 0x79
Counter	0x61 0x6c 0x6c 0x20 0x77 0x6f 0x72 0x6b 0x20 0x61 0x6e 0x64 0x20 0x6e 0x6f 0x20 0x70 0x6c 0x61 0x79 0x20 0x6d 0x61 0x6b 0x65 0x73 0x20 0x6a 0x61 0x63 0x6b 0x20 0x61 0x20 0x64 0x75 0x6c 0x6c 0x20 0x62 0x6f 0x79
ECB	0x61 0x6c 0x6c 0x20 0x77 0x6f 0x72 0x6b 0x20 0x61 0x6e 0x64 0x20 0x6e 0x6f 0x20 0x70 0x6c 0x61 0x79 0x20 0x6d 0x61 0x6b 0x65 0x73 0x20 0x6a 0x61 0x63 0x6b 0x20 0x61 0x20 0x64 0x75 0x6c 0x6c 0x20 0x62 0x6f 0x79
OFB	0x61 0x6c 0x6c 0x20 0x77 0x6f 0x72 0x6b 0x20 0x61 0x6e 0x64 0x20 0x6e 0x6f 0x20 0x70 0x6c 0x61 0x79 0x20 0x6d 0x61 0x6b 0x65 0x73 0x20 0x6a 0x61 0x63 0x6b 0x20 0x61 0x20 0x64 0x75 0x6c 0x6c 0x20 0x62 0x6f 0x79

Tabel 5. *Plaintext* hasil uji kebenaran hasil

C. *Analisis Frekuensi*

Eksperimen analisis frekuensi dilakukan dengan menggunakan *plaintext* pada Lampiran B yang berukuran 2935 karakter yang seluruhnya berbahasa Inggris.



Gambar 6. Grafik analisis frekuensi kemunculan karakter pada data berukuran 2935 karakter

Dari Gambar 6 dapat terlihat bahwa apabila dibandingkan dengan persebaran ASCII pada *plaintext* yang didominasi oleh ASCII pada rentang alfabet dan spasi, *ciphertext* pada kelima jenis blok cipher memiliki distribusi karakter yang merata dan tidak jauh berbeda antara satu dengan yang lain.

Jenis	MAX	MEAN	STDEV
CBC	31	11.5	3.753168596
CFB	21	11.46484375	3.355816486
Counter	57	11.5	8.189424074
ECB	42	11.46484375	4.143354516
OFB	20	11.46484375	3.359320422

Tabel 6. Statistik data pada *ciphertext*

Apabila melihat statistik frekuensi data yang terdapat pada Tabel 7, terlihat bahwa tiap *block cipher* memiliki jumlah rata-rata serta standard deviasi frekuensi kemunculan yang mirip. Pengecualian terdapat pada jenis Counter, dimana jumlah kemunculan terbanyak dan standard deviasinya jauh melampaui *block cipher* yang lain.

D. *Analisis Waktu Enkripsi*

Eksperimen untuk mengukur durasi enkripsi dilakukan dengan menggunakan *plaintext* pada Lampiran B yang berukuran 2935 karakter. Hasil dari eksperimen disajikan pada tabel 7. Berdasarkan eksperimen, mode CBC, CFB, dan OFB memakan waktu yang cukup cepat.

Mode	Waktu (ms)
CBC	0.2822165489
CFB	4.450150967
Counter	0.2924702168
ECB	0.3060278893
OFB	4.480173349

Tabel 7. Lama Waktu Enkripsi berdasarkan Mode

V. KESIMPULAN

Algoritma 474 memiki struktur yang hanya melibatkan operasi bit dan matematika sederhana. Meskipun demikian, algoritma ini cukup kuat dalam menghadapi serangan kriptanalis seperti serangan *bruteforce* dan analisis frekuensi. Hal tersebut dapat dilihat dari hasil persebaran karakter *ciphertext*. Selain itu, algoritma 474 juga memiliki kinerja yang cukup cepat.

REFERENSI

[1] C. Chris. (2010, January 27). *Caesar Ciphers*. Retrieved from http://www.nku.edu/~christensen.

[2] N. Arora and Y. Gigras, "Block and Stream Cipher Based Cryptographic Algorithms: A Survey," International Journal of Information and Computation Technology. India, vol. 4-2, pp. 189-196, 2014.

[3] C. E. Shannon, “Communication Theory of Secrecy Systems,” Bell System Technical Journal. vol. 28-4, pp. 656-715, October 1949.

[4] NIST, “Data Encryption Standard,” FIPS PUB 46-3. October 1999.

[5] D. Joan and R. Vincent, “The Design of Rijndael, AES - The Advanced Encryption Standard,” Springer-Verlag, pp. 238, 2002.

PERNYATAAN

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

Bandung, 12 Maret 2019

Husnulzaki Wibisono Haryadi 13515005

Aya Aurora Rimbamorani 13515098

LAMPIRAN

A. Plaintext Eksperimen I

all work and no play makes jack a dull boy

B. Plaintext Eksperimen II

Have you ever noticed that most other wine regions compare themselves to the Napa Valley? For good reason, Napa Valley wines are still the gold standard. And while these other areas have excellent, even award winning wines, Napa is the glamor name through the years.

I much prefer Napa Valley as well, just because of familiarity, and comfort. I know where to go, and what to eat. I have cycled through the Valley since the 70s. I love the vibe, despite the ridiculous prices of both wine and tasting options. Go online, and it is not unusual to find wine tasting options at some Napa wineries over $100!!!!! But best travel buddy, Mike and I were just talking this morning about this. Why are they doing the money grab? Because they can. They get visitors from around the world, never been here before, never coming back. Make them feel special by handing over $100 for "exclusive" tastings. This is simply not the Napa Valley I once knew. And the others are following suit, including the Central Coast wineries.

What can you do about this? One option I use is to join my favorite winery's tasting club. In my case, I have been a member of Domaine Carneros Chateau Society. It obligates me to buy two bottles of sparkling wine every other month. But when I visit, my tasting is complimentary, and my food and other tastings are discounted. And on weekends, we get to taste in their Member's Club Room and terrace. The other option is to just patronize the smaller guys. Why should I pay for someone's outrageous cost structure? We were joined by friends at Domaine Carneros, and spent about three hours there. We brought home a dozen bottles, in addition to my club purchases. With the cold and rain outside, the indoor areas were in great demand. In other words, it pays to have the club membership.

Another big issue is having a designated driver. Sheri is my designated driver, I am fortunate she does not drink! The highways are crawling with the local law enforcement people. So, just be smart! After a nice stop for a snack at Hog Island Oyster at Oxbow for a Po' Boy sandwich, we headed over to Sonoma. We had a very enjoyable dinner at the Swiss Hotel in downtown Sonoma. We were joined by two more extremely interesting characters from Sonoma. Their identities shall remain confidential, so that I can visit them again. We had a fabulous time, heard many "insider" stories, and had a very nice meal and even better laughs.

Dining here is another big plus, as there are no shortages of great places to eat. Since I prefer casual, I usually end up at Mustard's Grill in Yountville. Good menu, decent prices, noisy, but always good food, and a nice wine list. And parking on site, another plus. But we chose Sonoma on this trip. And, with our new and old friends, we will explore many more places in the future.

See you there, despite everything I just said. Good, bad, outrageous, it is still the one and only, Napa Valley!!