DAMEN: Algoritma blok cipher dengan pembangkitan kunci dengan fungsi hash

Riady Sastra Kusuma/13512024
Teknik Informatika
Institut Teknologi Bandung
Bandung, Indonesia
riadysastrak@gmail.com

Marcelinus Henry M./13512082
Teknik Informatika
Institut Teknologi Bandung
Bandung, Indonesia
henrymenori@yahoo.com

***Abstrak*—Makalah ini mengandung dasar teori dari konsep-konsep yang dipakai algoritma DAMEN, Rancangan algoritma, DAMEN, hasil dan eksperimen serta analisisnya**

Keywords—block cipher; kriptografi; kriptografi modern

I. PENDAHULUAN

Pada zaman ini, hampir semua hal berhubungan dengan internet. Informasi banyak sekali menyebar di internet dan informasi itu dapat dengan mudah diakses oleh semua orang yang mengakses internet.

Informasi tersebut banyak juga yang merupakan rahasia dan yang mempunyai informasinya tidak ingin informasi itu dapat diakses oleh semua orang. Maka dari itu dibutuhkan keamanan dalam pengiriman pesan diinternet. Salah satu cara pengamanannya adalah dengan kriptografi.

Kriptografi sudah dipakai sejak berabad-abad yang lalu.. Bahkan sebelum masehipun kriptografi sudah dipakai untuk mengamankan pengiriman pesan. Sudah banyak sekali algoritma kriptografi, seperti caesar cipher, vigenere cipher, playfair cipher dll.. Sudah banyak orang yang dapat memecahkan algoritma tersebut, jadi dapat dibilang algoritma tersebut sudah tidak aman. Algoritma-algoritma tersebut dapat digunakan hanya untuk teks saja, karena pada zaman dulu pesan yang dikirmkan berupa teks. Sedangkan pada zaman ini, informasi yang beredar pada internet merupakan serangkaian bit. Maka dari itu, algoritma-algoritma kuno yang berbasis teks sudah jarang digunakan untuk mengamankan pesan yang tersebar di internet.

Pada zaman sekarang algoritma yang sering dipakai untuk mengamankan pesan di internet adalah kriptografi modern. Kriptografi modern adalah kriptografi yang berbasis bit. Kriptografi modern dibagi menjadi dua yaitu, block cipher dan stream cipher. Beberapa contoh block cipher adalah DES dan AES. Algoritma-algoritma tersebut sudah sangat umum digunakan dalam mengamankan pesan.

Di makalah ini akan dibahas tentang algoritma DAMEN. Hal-hal akan dibahas adalah tentang bagaimana metodanya, ekperimen tentang algoritma ini dan analisis keamanannya.

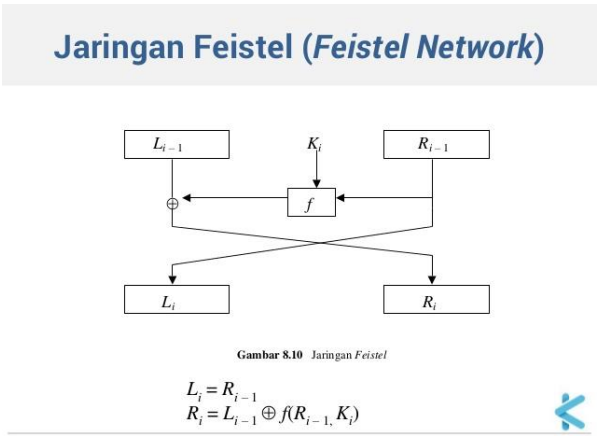
II. DASAR TEORI

Pada algoritma DAMEN memakai beberapa konsep yang sering digunakan dalam algoritma kriptografi modern. Berikut adalah konsep-konsep yang kami pakai dalam algoritma ini.

A. Jaringan Feistel

Jaringan Feistel ditemukan oleh Horst Feistel 1970. Model jaringan Feistel adalah sebagai berikut:

1. Bagi blok yang panjangnya n bit menjadi dua bagian, kiri (L) dan kanan (R), yang masing-masing panjangnya $n/2$ (hal ini mensyaratkan n harus genap).
2. Definisikan cipher blok berulang dimana hasil dari putaran ke-i ditentukan dari hasil putaran sebelumnya



Gambar 1. Skema jaringan feistel [1]

Plainteks adalah gabungan L dan R awal, atau secara formal dinyatakan dengan (L_0, R_0), sedangkan cipherteks didapatkan dari L dan R hasil dari putaran terakhir setelah terlebih dauhulu dipertukarkan, atau secara formal dinyatakan sebagai (R_r, L_r).

B. Cipher berulang

Fungsi transformasi sederhana yang mengubah plainteks menjadi cipherteks diulang sejumlah kali. Pada setiap putaran

digunakan upa-kunci (subkey) atau kunci putaran (round key) yang dikombinasikan dengan plainteks.

Secara formal, cipher berulang dinyatakan sebagai

$C_i = f(C_i - 1, K_i)$

yang dalam hal ini,

$i = 1, 2, \ldots, r$ (r adalah jumlah putaran).

K_i = upa-kunci (*subkey*) pada putaran ke-i

f = fungsi transformasi (di dalamnya terdapat fungsi substitusi, permutasi, dan/atau ekspansi, kompresi).

Plainteks dinyatakan dengan C_0 dan cipherteks dinyatakan dengan C_r.

C. Kotak-S

Kotak-S adalah matriks yang berisi substitusi sederhana yang memetakan satu atau lebih bit dengan satu atau lebih bit yang lain. Berikut adalah contoh kotak-S.

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8

Masukan untuk proses subtitutsi adalah 5 bit.

$b_1b_2b_3b_4b_5$

string bit b_1 menyatakan nomer baris tabel

string bit $b_2b_3b_4b_5$ menyatakan nomor kolom tabel

Perancangan kotak-S menjadi isu penting karena kotak-S harus dirancang sedemikian sehingga kekuatan kriptografinya bagus dan mudah diimplementasikan.

Ada 4 cara untuk menentukan isi dari kotak-S:

- Dipilh secara acak
- Dipilih secara acak lalu diuji
- Dibuat oleh orang
- Dihitung secara matematis

D. Mode operasi

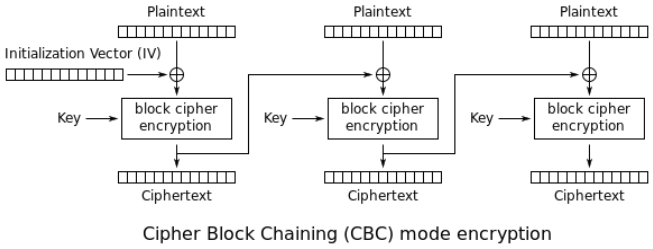
Terdapat beberapa mode operasi dalam krpitografi modern, yaitu:

1) ECB (Electronic code book)

Pada mode ini, setiap blok dienkripsi secara individual dan independen

2) CBC(Cipher block chaining)

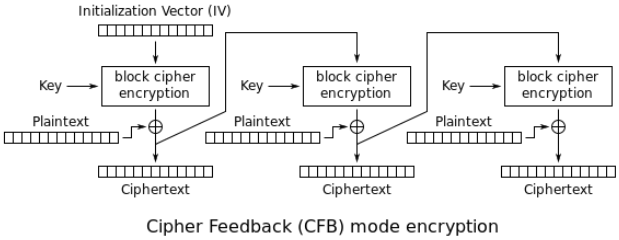
Setiap blok cipherteks tidak hanya bergantung pada blok plainteksnya, tetapi bergantung pula pada blok sebelumnya. Berikut adalah skema dari CBC:



Gambar 2. Skema CBC [2]

3) CFB (Cipher feedback)

Setiap blok plainteks dipakai untuk menjadi umpan pada blok berikutnya dengan cara dienkripsi terlebih dahulu. Berikut adalah skemanya:



Gambar 3. Skema CFB [2]

III. RANCANGAN BLOCK CIPHER

A. Algoritma pembangkit kunci

Untuk setiap kunci yang dimasukkan, akan dibangkitkan empat kunci berbeda yang akan digunakan untuk proses enkripsi dan dekripsi. Ada empat tahap untuk pembangkitan kunci yang akan menghasilkan satu kunci berbeda pada setiap tahapnya. Berikut tahapan pembangkitannya. Kunci *k* merupakan tipe data string dengan panjang 32 byte

.

1. Inisialisasi

Hitung nilai konstanta *m* dengan rumus hash berikut.

$$m \equiv k[0]\cdot 31^{n-1} + k[1]\cdot 31^{n-2} + \ldots + k[n-2]\cdot 31^{1} + k[n-1] \bmod 256$$

Kemudian nilai dari *m* diubah menjadi karakter. Lakukan *xor* dengan konstanta *m* untuk setiap karakter pada kunci *k*. Selanjutnya, ubah kunci *k* menjadi sebuah matriks berukuran 4x8, dengan isi dari tiap elemen matriks adalah karakter dari kunci *k*.

2. Tahap 1

Tukar baris pertama matriks dengan baris keempat matriks dan tukar baris kedua matriks dengan baris ketiga matriks. Ubah menjadi bentuk string untuk mendapatkan kunci k_1.

3. Tahap 2

Selanjutnya, tukar kolom pertama matriks dengan kolom kedelapan matriks, tukar kolom kedua matriks dengan kolom

keempat matriks, dan tukar kolom kelima matriks dengan kolom ketujuh matriks. Ubah menjadi bentuk string untuk mendapatkan kunci k_2.

4. Tahap 3

Kemudian, tukar baris pertama matriks dengan baris kedua matriks dan tukar baris ketiga matriks dengan baris keempat matriks. Ubah menjadi bentuk string untuk mendapatkan kunci k_3.

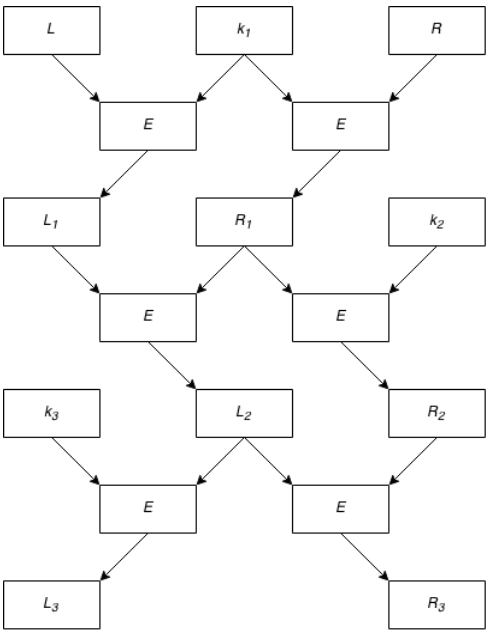
5. Tahap 4

Bentuk kunci k4 dengan menggeser karakter dari kunci k_3 sebanyak 3 ke kiri. Didapat kunci k_4.

B. Algoritma enkripsi

1. Fungsi enkripsi :

Fungsi enkripsi menerima masukan berupa string plainteks sebesar 64 byte, yang dipisah menjadi 2 bagian yaitu *L* dan *R* masing-masing sebesar 32 byte. Kemudian dilakukan enkripsi dengan diagram seperti berikut:



Gambar 4. Skema enkripsi

Kunci yang dipakai akan berulang terus mulai dari k_1, k_2, k_3, k_4, dan kembali lagi ke k_1. Fungsi *E* menunjukkan proses enkripsi yang lebih spesifik, dimana fungsi *E* menerima dua buah string sebesar 32 byte dan mengembalikan sebuah string berukuran 32 byte yang sudah dienkripsi. Berikut spesifikasi fungsi *E*.

a. Tahap Inisialisasi

Misalkan untuk tahapan pertama dengan masukan *L* dan k_1, dibuat 4 buah matriks berukuran 4x4 sehingga *L* menjadi 2 matriks dan k_1 menjadi 2 matriks. Kita namakan matriks dari *L* matriks *a* dan *b*, sedangkan matriks dari k_1 adalah matriks *c* dan *d*.

b. Tahap 1, pergeseran

Untuk setiap matriks, lakukan

- Baris pertama tetap
- Baris kedua digeser 1 ke kanan
- Baris ketiga digeser 2 ke kanan
- Baris keempat digeser 3 ke kanan
- Tukar baris pertama dan baris ketiga, tukar baris kedua dan baris keempat
- Tukar kolom pertama dan kolom kedua, tukar kolom ketiga dan keempat

c. Tahap 2, perkalian *xor*

Lakukan operasi *xor* antara *a* dengan *c* dan *b* dengan *d*. Operasi *xor* dilakukan mirip seperti perkalian matriks yaitu baris pertama matriks *a* dengan kolom pertama matriks *c*, baris kedua matriks *a* dengan kolom kedua matriks *c*, dan seterusnya. Ubah matriks *a* dan *c* menjadi string utuh berukuran 32 byte sehingga dihasilkan string L_1.

d. Tahap 3, subtitusi S-BOX

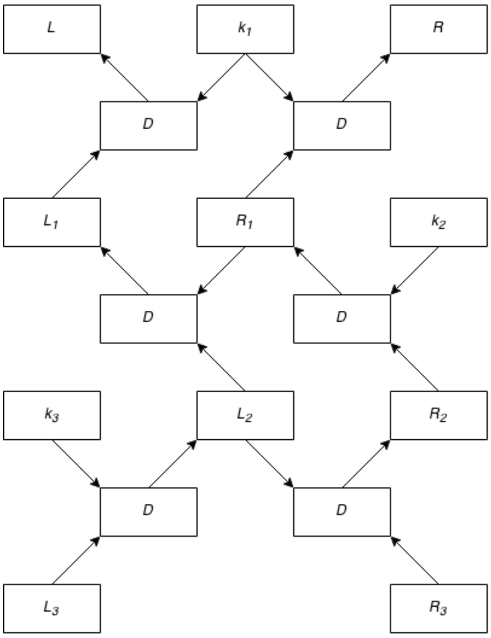
Lakukan subtitusi S-BOX dengan S-BOX Rijndael yaitu

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

Setelah semua proses selesai hingga menghasilkan L_3 dan R_3, proses diulang dari awal dengan L_3 dan R_3 sebagai masukan. Lakukan langkah tersebut hingga proses terjadi sebanyak 32 kali putaran.

2. Fungsi dekripsi

Fungsi dekripsi hanya membalikkan proses pada fungsi enkripsi, namun diagram yang digunakan sedikit berubah karena ada keterbatasan informasi. Dimulai dari cipherteks berukuran 64 byte yang dipisah menjadi dua buah string 32 byte yaitu L_3 dan R_3. Berikut diagramnya



Gambar 5. Skema dekripsi

Fungsi dekripsi menerima dua buah string 32 byte dan mengeluarkan sebuah string 32 byte yang sudah terdekripsi. Pada intinya, proses ini hanya kebalikan dari proses enkripsi yang telah disebutkan pada bagian sebelumnya.

IV. Eksperimen dan Pembahasan Hasil

Berikut hasil dari implementasi algoritma yang sudah dirancang.

Plainteks :

Cryptography prior to the modern age was effectively synonymous with encryption, the conversion of information from a readable state to apparent nonsense. The originator of an encrypted message shared the decoding technique needed to recover the original information only with intended recipients, thereby precluding unwanted persons to do the same. Since World War I and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread.

Kunci :

nice cryptography you got there!

Hasil dengan ECB :

QJUPW} KÏ ¹u ¤h¤U) 60 □æ·Ÿ¹À□0²¸□Õ×øŽ C'Z Ê{v Xóè;°ý ï ¤$ô-zi_¼, %S¼A cj¥ °^=•ÕÙ¿&½•Y¹×_²êÒ1 4`w¹I¥µ tÁ 326ºS&Ô47?ñ õakR¡q^ F»\Ÿ#Ó6ÉB7}Ñt-úeë}8□Ìîý¸8¥□#÷æÂ□ÛkÊ](éÕ*»□Ï□ÖöÐA”9b□ÂFO Â–□†ª˜àHZ]îyøÔ^—»Í ô÷-□ÕÜÐîœù)’□□z`CÑ□•Ê

™WŸT□‹P£□î□-°}±ÍvÕsl*áµB÷n_aûí□´ŠÃZI…é□*£8=bóÞ2/¦AN)Á$O†/åIki0Ô•Š

Hasil dengan CBC :

□缉솩瓢・绷□嘍□?貍箊□□P□□抹叓倅・諺覴?嗔掳갓腣□⚙㑷幫蓖?녇醞蹭 菰鑊□禮□蜜・渧□?鸠씉rad熄밍?□?ट□⩾・擴□홥삔莉瀿轝□捚・訷□긺닁덥俜✠□懲嬐〰동骇□□瘆□□꾺豙⁝ば뉴斺鬆궗Ű訂乭ᓗ芊彊젴갌켑떙鞭舵肟䜣㟶?ᐟ・豚킨ᑎ혈伉擈畽易□꽃□‿韡胸□粃篁恨㝮憮엥□・□쌕糧贵黇-
□π 砙苹䓛薅賫⍝鄘ᕃ□꾌铜□갬ᛗ □福监嵋粹븨 봀所宀蛤猳扨酈□ي□?↩픔을싉 헶氤 θ 議莣滀‿□秠秘뷰挈滬昭僑ᒐ锇□豥□□沩艿晟뵜 탕□庡仩昊?豋?즈□숋□浹强歙謔벙鐛□□□□뵒褫Ť觧□弧袮・饕鶚왐ㄝ킨嘭妟暖饐祏?Ꮍ□ꙮ☆ ҧ□□읦藞猽迄橘緆•蔚□혀罙☆ 쾡梓謏沍・郇粵懞珇✕唪秺?僳贗埢댐은□껬腘줭□徖蹇悬械・⌈飝□땅□ᓚ땐駺ꮚ됳・□択?벡ⷳ 筌蓼 ρ 额䅁厘狍偺▼≟?沶븐贖ῴ□걶眡7ᏂƐ□猗ᚭ烙□쥩炏页跲□粗فمي㶎ᒻ 旸□˝E䜷檻公□灩痺ꀜ 珧ᓚ労杪驃댁ậ彏ᛃ・ꙍ北禧萓鼜뾃줓잰藻嵫諰호□취♀ 庫쐥锖ᓘ□乀ㄱ 蚕?亱돠駤☐□맵ᓘ뷺ნ캐膝・絆?☻琑慝挽섰肜头□ꜝ ᵉ ᏰᏰ鰊诬ᛢ η蒸봐ᖾ왤튄齊仦鉲掽쾟韞瘺餇☐ᘁ 挘扯⋨뵞갹烽□鏞・俳・ی箬□□弖□□⁚ᏅᏅ?□솟뮽⅍鲅箭巨✭ 崅?鉖嗺□□ᅩ화□熈جےñ梹餂log 갔□ⓚ 枯ᐅ⊪ 媜몣维揕ㅐ 䂣ᕊ뱎欓疇秔24岾耜・탓・旃■変衉乤觚鯎镲뾕䋶蠌

Hasil dengan CFB :

熡鞸]行⋞□楞粶□莿捶繎졥쩩䛑ᢊ읃 ・丕豖õÙ뢷鑾□?□ɔ 쐒□□扊ヽ/Ȩ□穴살□滪熄젫?#祕□꾌䊮宰졋蜡隊蠆郓긍䶣 획㊀殨・ᵩ琈楜柾愢□罠䂹璘・纆□嘶□?赛窗□Ꮡю□□懜丞便・譎觊?唔揖갼臾□⊡樽乆蒹?뵘釿軩r荅醿□祷 蟦・燹□?鶻셋ᄈµg要밧?□?□⊨・撑□헵색峅澑祉□拂・託□걩닎돼캳❋□應学ᕳ델騴☐□廣□□꾌줆⁚ 〖뇻绰鮮근ȩ詥俸ᓗ乔莮쥟탄衰뗑音昆蔵냥뽛?Ꮳ・谨敆ᛃᓖ활佬旋瓴轢□꼰□□輼肝□流箪怉宣庴웰□・□쌀紂畾鷰ᑊ៣ᐂ 씀芗菠藍謗ᕍ郊⊠□꾌钭ᒕ겉ᡠ □☐皺贯过뒔뵘欃嶲蛄賣戳牗□غم□?⇒뢺뵍숭혈汉ᛎ謙莚渔>□襘柒뵌榃楙時傣ᓚ钀ᕍ逍□ᖶ沉彬昱뵌텃□뫃亀晾?貗?豨□훪□洧䕶欽諗벡戀罷□□□뵞褖Ḟ萩□弇蓺・霦鷚엥ᐯ쿸喓뒥眐□稤?⋷˝E힉ᕼ I□□笅虑谐辪橱粿✿蔺□쾥繝ㅣ켹楤謦沭・郇粕儋琚❋嗹䅟?仸雛塹뱐옰갈瓬옰□笅瘏膠猪・Ɣ飈□뗩□ᏋᏋ뎚聘□뤽・□拺?뼿Φ 欉蕖Π餰秞叽黠皲▜?浄뫽貓Ÿ□갸晐nFᕷ□獣ᛏ熙□쬉奂ᄈ줄□聰تمي淚峠ᛆ 斞□˝I豉㯬叟□ᄈ幰ᛢ 玶ᓗ漏桻駸뎁ᕕ臢ᏋᏋᛪ・Ꮳ□巵秄芽留뾀岙익蘺嵖稈챂□좋ᕼ 廉쐟镵□○く 끃?賽뎀駸☐ᕒ멌ᓗ븟ᓗ갸膦・耨?ᛢ빠恎擂셕能奚□ᄀ□♀鑢诌ᛆ ᓗ虢멩ᕕ경긎馯佡銼揹첨韛瘷餀□ᛢ 戋戆Ꮼ쩽굊贩□鏟・侗・ᓗ管

Dari hasil eksperimen didapatkan bahwa hasil dengan ECB terlihat sedikit dan tidak tampak huruf mandarin. Hal ini terjadi karena mode ECB yang tidak terkait antar blok yang satu

dengan blok lainnya, sehingga tidak memicu huruf untuk keluar dari range 256 *bit character*, selain itu hasil enkripsi yang lebih sedikit dari plainteks, menandakan bahwa ada karakter yang terenkripsi menjadi karakter *backspace*. Berbeda dengan hasil CBC dan CFB yang sebagian besar terdiri dari huruf mandarin, karena dipicu oleh fungsi yang memanfaatkan blok sebelumnya atau sesudahnya (*xor*), sehingga huruf bisa keluar dari range 256 *bit character*.

V. ANALISIS KEAMANAN

Keamanan algoritma ini terletak pada panjangnya kunci yang dibutuhkan yaitu sebesar 32 byte. Apabila dilakukan brute force terhadap suatu ciphertext, dengan asumsi setiap 1 juta proses enkripsi dapat dilakukan dalam waktu 1 ms, maka lamanya waktu yang dibutuhkan untuk menemukan kunci adalah 256^{32} ms, atau setara dengan

$$t = \frac{256^{32}\,ms}{1000000} \cdot \frac{1s}{1000ms} \cdot \frac{1h}{3600s} \cdot \frac{1d}{24h} \cdot \frac{1y}{356d} = 599.73 \times 256^{24}\,year$$

Dibutuhkan waktu berabad-abad untuk memecahkan kunci tersebut. Jadi, kemungkinan kunci dapat ditemukan dengan brute force sangatlah kecil.

Keamanan algoritma ini juga terletak pada penyebaran byte. Apabila kunci yang dimasukkan berbeda satu huruf saja dengan kunci yang sebenarnya, hasil dekripsi sama sekali tidak menunjukkan kemiripan dengan plainteks yang sebenarnya. Hal ini akan menyulitkan analisis, dimana kriptanalis tidak akan tahu seberapa dekat perkiraannya dengan kunci yang sebenarnya. Contoh :

Dekripsi CBC dengan kunci "*nice cryptography you got there!*"

Cryptography prior to the modern age was effectively synonymous with encryption, the conversion of information from a readable state to apparent nonsense. The originator of an encrypted message shared the decoding technique needed to recover the original information only with intended recipients, thereby precluding unwanted persons to do the same. Since World War I and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread.

Dekripsi CBC dengan kunci "*nice cryptography you get there!*"

->A âCql< □dœ% • ÇÀ;€ ë¨ÒÆq«®dá
„UÀ…Î_k"?3Nª J□Sª=-)Éo°ùÓÎ.□³"c*pÐ

Terlihat perbedaan yang sangat jauh terhadap hasil dekripsi, walaupun perbedaan kunci hanya satu karakter.

VI. KESIMPULAN DAN SARAN

Algoritma DAMEN dapat dibilang cukup aman, karena untuk mencari kuncinya dibutuhkan waktu yang lama. Selain itu jika kuncinya berbeda 1 karakter saja, hasil dekripsinya pun tidak dapat dibaca sama sekali.

Saran untuk algoritma kriptografi block cipher adalah algoritmanya harus mudah dimplementasikan dan harus aman

VII. DAFTAR PUSTAKA

[1] http://www.slideshare.net/KuliahKita/kriptografi-prinsip-perancangan-cipher-blok diakses pada 17 Maret 2015

[2] https://www.adayinthelifeof.nl/2010/12/08/encryption-operating-modes-ecb-vs-cbc/ diakses pada 17 Maret 2015

[3] http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2014-2015/kripto14-15.htm diakses pada 17 Maret 2015