Data Encryption Standard (DES)

Rachmat Arifin

Teknik Informatika
Sekolah Teknologi Elektro dan Informatika
Institut Teknologi Bandung
Jln Cijawura GIrang II / I No 8
If18070@students.if.itb.ac.id

ABSTRAK

Dalam dunia digital, pengamanan sangat dibutuhkan bagi informasi – informasi yang ada. Enkripsi merupakan salah satu metode yang banyak digunakan untuk sistem pengaman informasi digital. Enkripsi merupakan sistem pengamanan yang memanfaatkan algoritma tertentu untuk merubah *plaintext* menjadi sebuah *cipher* yang tidak akan dimengerti tanapa proses dekripsi terlebih dahulu.

Kata kunci: enkripsi, dekripsi, cipher, pengamanan, informasi .

1. PENDAHULUAN

Data Encryption Standard berawal dari awal tahun 1970an. Pada tahun 1972, NBS (*National Bureau of Standards)* yang merupakan badan pemberian standar Amerika serikat mengidentifikasikan sebuah kebutuhan untuk memberikan standar pemerintah tentang pengenkripsian informasi sensitive yang belum terklasifikasi. Pada 15 Mei 1973, setelah NBS berkonsultasi dengan NSA, NBS mengajukan proposal untuk sebuah *cipher* yang memenuhi kriteria yang diinginkan. Pada pengajuan proposal yang pertama ini, NBS tidak menemukan *cipher* yang memenuhi kriteria. Pada 27 Agustus 1974 pengajuan kedua diserahkan. Kali ini, IBM mencalonkan *cipher* Lucifer yang dikembangkan oleh Horst Feistel, Walter Tuchman, Don Coppersmith, Alan Konheim, Carl Meyer, Mike Matyas, Roy Adler, Edna Grossman, Bill Notz, Lynn Smith, dan Bryan Tuckerman. Penamaan Lucifer diambil berdasarkan nama Lucifer yang merupakan salah satu *Demon*. *Demon* pada hal ini berarti *Demonstration*.

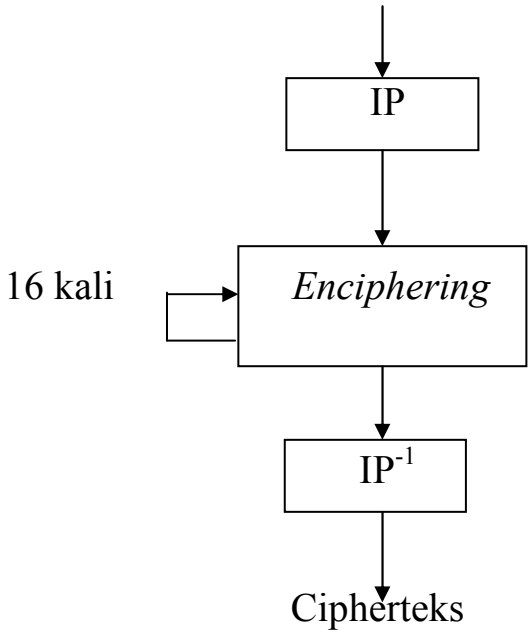
2. METODE

DES termasuk ke dalam sistem kriptografi simetri dan tergolong jenis *cipher* blok. DES dirancang untuk melakukan *enchiper* dan *dechiper* data yang berisi 56 bit dibawah kendali 56 bit kunci internal atau upa-kunci. Dalam melakukan *dechiper* harus dilakukan dengan menggunakan kunci yang sama dengan saat proses *enchiper* tetapi sat melakukan *dechiper* pemberian alaman berubah sehingga proses *dechiper* merupakan kebalikan dari proses *enchiper*. Sejumlah data yang akan di *enchiper* disebut sebagai permutasi awal atau *initial permutation (IP)*. Komputasi *key – dependent* didefinisikan sebagai fungsi ***f*** sebgai fungsi *chipper* dan function ***KS*** sebagai *key schedule*. Deskripsi dari komputasi diberikan pertama, bersama dengan detail bagaimana algoritma digunakan dalam proses *enchiper*. Selanjutnya, penggunaan algoritma untuk proses *dechiper* dideskripsikan. Pada akhirnya, sebuah definisi *chipper* fungsi ***f*** diberikan dalam bentuk fungsi primitive yang disebut fungsi seleksi ***Si*** dan fungsi permutasi ***P***.

2.1 Skema Global DES

Pada awalnya, blok plainteks dipermutasi dengan matriks permutasi awal (*initial permutation* atau IP). Hasil dari permutasi awal tersebutkemudian di *enchiper* sebanyak 16 kali atau 16 putaran. Setiap putarannya menggunakan kunci internal yang berbeda. Hasil dari proses *enchiper* kembali dipermutasi dengan matriks permutasi balikan (*invers initial permutation* atau IP^{-1}) menjadi blok cipherteks.

Plainteks



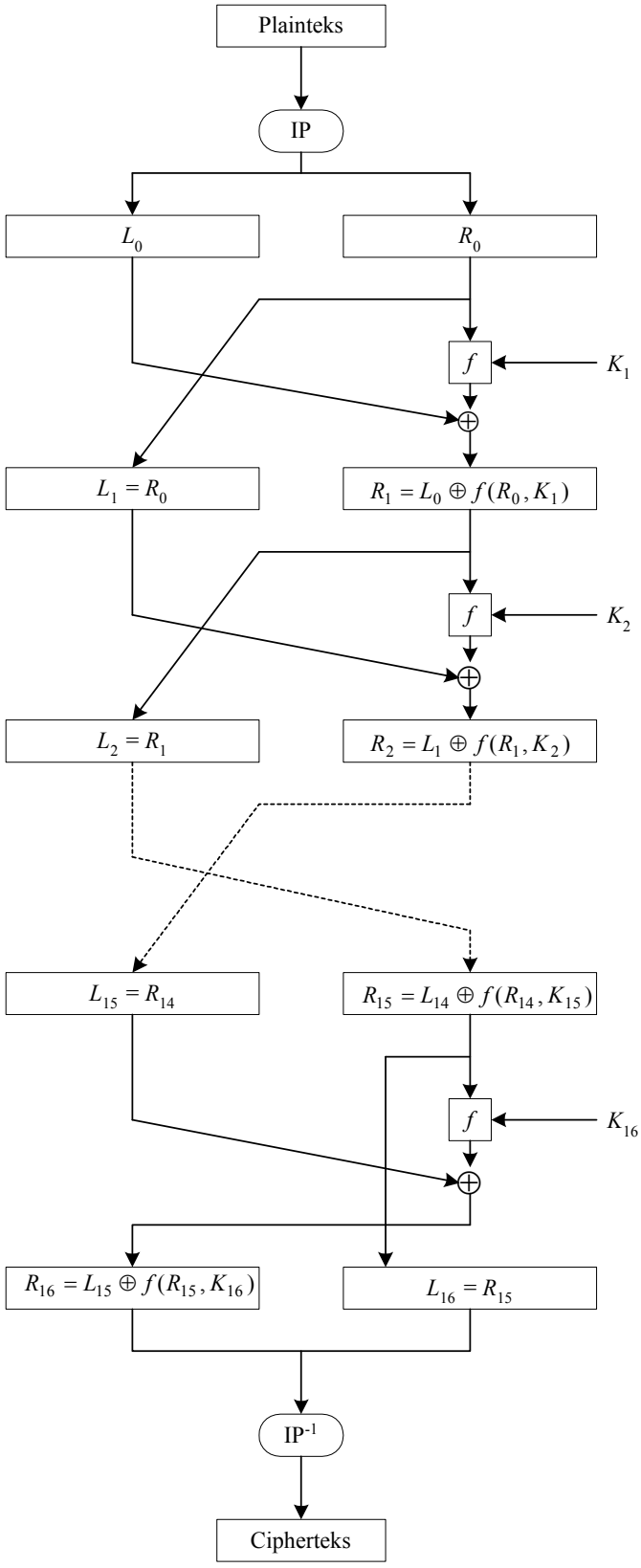
Gambar 1. Skema global DES

Dalam proses *enchiper*, blok plainteks terbagi menjadi dua bagian yaitu bagian kiri (L) dan bagian kanan (R), yang masing masing memiliki panjang 32 bit. Pada setiap putaran *i*, blok R merupakan masukan untuk fungsi transformasi fungsi ***f***. Pada fungsi ***f***, blok R dikombinasikan dengan kunci internal K_i. Keluaran dari fungsi ini di XOR kan dengan blok L yang langsung diambil dari blok R sebelumnya. Ini merupakan 1 putaran DES.

Secara matematis, satu putaran DES dinyatakan sebagai berikut :

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

Gambar 2 memperlihatkan skema algoritma DES yang lebih rinci dan jelas.



Gambar 2. Algoritma enkripsi dengan DES

2.2. Permutasi Awal

Sebelum putaran pertama, terhadap blok plainteks dilakukan permutasi awal (*initial permutation* atau IP). Tujuan permutasi awal adalah mengacak plainteks sehingga urutan bit-biit di dalamnya berubah. Pengacakan dilakukan dengan menggunakan matriks permutasi awal berikut ini:

29	21	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	55	30	24	16	8
57	49	41	33	25	17	9	32	59	51	43	35	27	19	11	3
61	53	45	37	58	50	13	5	63	1	47	39	31	23	15	7

Cara membaca tabel/matriks di atas: dua *entry* ujung kiri atas (58 dan 50) berarti:

"pindahkan bit ke-29 ke posisi bit 1"
"pindahkan bit ke-21 ke posisi bit 2"

2.3. Pembangkitan Kunci Internal

Karena ada 16 putaran, maka dibutuhkan kunci internal sebanyak 16 buah, yaitu K_1, K_2, …, K_{16}. Kunci-kunci internal ini dapat dibangkitkan sebelum proses enkripsi atau bersamaan dengan proses enkripsi. Kunci internal dibangkitkan dari kunci eksternal yang diberikan oleh pengguna. Kunci eksternal panjangnya 64 bit atau 8 karakter. Misalkan kunci eksternal yang tersusun dari 64 bit adalah *K*.

Kunci eksternal ini menjadi masukan untuk permutasi dengan menggunakan matriks permutasi kompresi PC-1 sebagai berikut:

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

Dalam permutasi ini, tiap bit kedelapan (*parity bit*) dari delapan *byte* kunci diabaikan. Hasil permutasinya adalah sepanjang 56 bit, sehingga dapat dikatakan panjang kunci DES adalah 56 bit. Selanjutnya, 56 bit ini dibagi menjadi 2 bagian, kiri dan kanan, yang masing-masing panjangnya 28 bit, yang masing-masing disimpan di dalam C_0 dan D_0:

C_0: berisi bit-bit dari *K* pada posisi
57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18
10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36

D_0: berisi bit-bit dari *K* pada posisi
63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22
14, 6, 61, 53, 45, 37, 29, 21, 13, 5, 28, 20, 12, 4

Selanjutnya, kedua bagian digeser ke kiri (*left shift*) sepanjang satu atau dua bit bergantung pada tiap putaran. Operasi pergeseran bersifat *wrapping* atau *round-shift*.

Misalkan (C_i, D_i) menyatakan penggabungan C_i dan D_i. (C_{i+1}, D_{i+1}) diperoleh dengan menggeser C_i dan D_i satu atau dua bit.

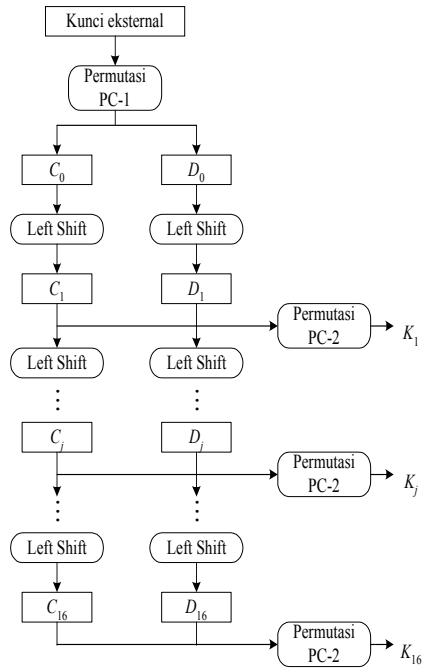
Setelah pergeseran bit, (C_i, D_i) mengalami permutasi kompresi dengan menggunakan matriks PC-2 berikut:

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

Dengan permutasi ini, kunci internal K_i diturunkan dari (C_i, D_i) yang dalam hal ini K_i merupakan penggabungan bit-bit C_i pada posisi:
14, 17, 11, 24, 1, 5, 3, 28, 15, 6, 21, 10
23, 19, 12, 4, 26, 8, 16, 7, 27, 20, 13, 2

dengan bit-bit D_i pada posisi:
41, 52, 31, 37, 47, 55, 30, 40, 51, 45, 33, 48
44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32

Jadi, setiap kunci internal K_i mempunyai panjang 48 bit.



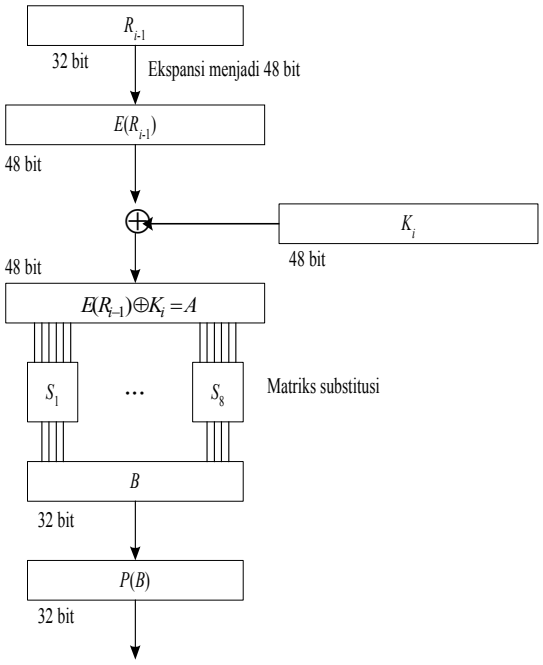
Gambar 3. Proses Pembangkitan Kunci – Kunci Internal DES

2.5. Enchipering

Proses *enciphering* terhadap blok plainteks dilakukan setelah permutasi awal (lihat Gambar 1). Setiap blok plainteks mengalami 16 kali putaran *enciphering* (lihat Gambar 2). Setiap putaran *enciphering* merupakan jaringan Feistel yang secara matematis dinyatakan sebagai

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

Diagram komputasi fungsi f diperlihatkan pada Gambar 5.



Gambar 4. Rincian Komputasi Fungsi *f*

E adalah fungsi ekspansi yang memperluas blok R_{i-1} yang panjangnya 32-bit menjadi blok 48 bit. Fungsi ekspansi direalisasikan dengan matriks permutasi ekspansi sbb:

32	1	2	3	4	5	4	5	6	7	8	9
8	9	10	11	12	13	12	13	14	15	16	17
16	17	18	19	20	21	20	21	22	23	24	25
24	25	26	27	28	29	28	29	30	31	32	1

Selanjutnya, hasil ekpansi, yaitu $E(R_{i-1})$, yang panjangnya 48 bit di-XOR-kan dengan K_i yang panjangnya 48 bit menghasilkan vektor A yang panjangnya 48-bit:

$$E(R_{i-1}) \oplus K_i = A$$

Vektor A dikelompokkan menjadi 8 kelompok, masing-masing 6 bit, dan menjadi masukan bagi proses substitusi. Proses substitusi dilakukan dengan menggunakan delapan buah kotak-S (*S-box*), S_1 sampai S_8. Setiap kotak-S menerima masukan 6 bit dan menghasilkan keluaran 4 bit. Kelompok 6-bit pertama menggunakan S_1, kelompok 6-bit kedua menggunakan S_2, dan seterusnya.
(*cara pensubstitusian dengan kotak-S sudah dijelaskan pada materi "Prinsip-prinsip Perancangan* Cipher *Blok"*)
Keluaran proses substitusi adalah vektor B yang panjangnya 48 bit. Vektor B menjadi masukan untuk proses permutasi. Tujuan permutasi adalah untuk mengacak hasil proses substitusi kotak-S. Permutasi dilakukan dengan menggunakan matriks permutasi P (*P-box*) sbb:

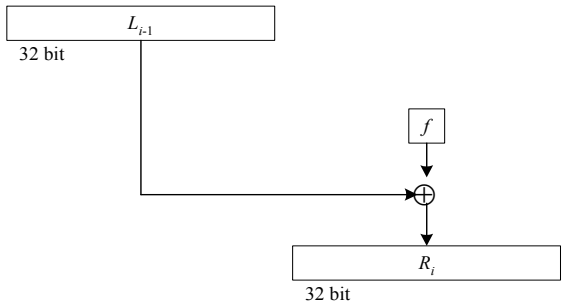
16	7	20	21	29	12	28	17	1	15	23	26	5	8	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

Bit-bit $P(B)$ merupakan keluaran dari fungsi f. Akhirnya, bit-bit $P(B)$ di-XOR-kan dengan L_{i-1} untuk mendapatkan R_i (lihat Gambar 6):

$$R_i = L_{i-1} \oplus P(B)$$

Jadi, keluaran dari putaran ke-i adalah

$$(L_i, R_i) = (R_{i-1}, L_{i-1} \oplus P(B))$$



Gambar 5. Skema Perolehan R_i

2.5. Permutasi Terakhir (*Inverse Initial Permutation)*

Permutasi terakhir dilakukan setelah 16 kali putaran terhadap gabungan blok kiri dan blok kanan. Proses permutasi menggunakan matriks permutasi awal balikan (*inverse initial permutation* atau IP^{-1}) sbb:

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

2.6. Dekripsi

Proses dekripsi terhadap cipherteks merupakan kebalikan dari proses enkripsi. DES menggunakan algoritma yang sama untuk proses enkripsi dan dekripsi. Jika pada proses enkripsi urutan kunci internal yang digunakan adalah K_1, K_2, …, K_{16}, maka pada proses dekripsi urutan kunci yang digunakan adalah K_{16}, K_{15}, …, K_1.

Untuk tiap putaran 16, 15, …, 1, keluaran pada setiap putaran *deciphering* adalah

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

yang dalam hal ini, (R_{16}, L_{16}) adalah blok masukan awal untuk deciphering. Blok (R_{16}, L_{16}) diperoleh dengan mempermutasikan cipherteks dengan matriks permutasi IP^{-1}. Pra-keluaran dari *deciphering* adalah adalah (L_0, R_0). Dengan permutasi awal IP akan didapatkan kembali blok plainteks semula.

Tinjau kembali proses pembangkitan kunci internal pada Gambar 4. Selama *deciphering*, K_{16} dihasilkan dari (C_{16}, D_{16}) dengan permutasi PC-2. Tentu saja (C_{16}, D_{16}) tidak dapat diperoleh langsung pada permulaan *deciphering*. Tetapi karena (C_{16}, D_{16}) = (C_0, D_0), maka K_{16} dapat dihasilkan dari (C_0, D_0) tanpa perlu lagi melakukan pergeseran bit. Catatlah bahwa (C_0, D_0) yang merupakan bit-bit dari kunci eksternal K yang diberikan pengguna pada waktu dekripsi.

Selanjutnya, K_{15} dihasilkan dari (C_{15}, D_{15}) yang mana (C_{15}, D_{15}) diperoleh dengan menggeser C_{16} (yang sama dengan C_0) dan D_{16} (yang sama dengan C_0) satu bit ke kanan. Sisanya, K_{14} sampai K_1 dihasilkan dari (C_{14}, D_{14}) sampai (C_1, D_1). Catatlah bahwa (C_{i-1}, D_{i-1}) diperoleh dengan menggeser C_i dan D_i dengan cara yang sama seperti pada Tabel 1, tetapi pergeseran kiri (*left shift*) diganti menjadi pergeseran kanan (*right shift*).

2.6. Keamanan DES

Isu-isu yang menjadi perdebatan kontroversial menyangkut keamanan DES:

1. Panjang kunci
2. Jumlah putaran
3. Kotak-S

2.6.1. Panjang Kunci

Panjang kunci eksternal DES hanya 64 bit atau 8 karakter, itupun yang dipakai hanya 56 bit. Pada rancangan awal, panjang kunci yang diusulkan IBM adalah 128 bit, tetapi atas permintaan NSA, panjang kunci diperkecil menjadi 56 bit. Alasan pengurangan tidak diumumkan.

Tetapi, dengan panjang kunci 56 bit akan terdapat 2^{56} atau 72.057.594.037.927.936 kemungkinan kunci. Jika diasumsikan serangan *exhaustive key search* dengan menggunakan prosesor paralel mencoba setengah dari jumlah kemungkinan kunci itu, maka dalam satu detik dapat dikerjakan satu juta serangan. Jadi seluruhnya diperlukan 1142 tahun untuk menemukan kunci yang benar.

Tahun 1998, *Electronic Frontier Foundation* (EFE) merancang dan membuat perangkat keras khusus untuk menemukan kunci DES secara *exhaustive search key* dengan biaya $250.000 dan diharapkan dapat menemukan kunci selama 5 hari. Tahun 1999, kombinasi perangkat keras EFE dengan kolaborasi internet yang melibatkan lebih dari 100.000 komputer dapat menemukan kunci DES kurang dari 1 hari.

2.6.2. Jumlah Putaran

Dari penelitian, DES dengan jumlah putaran yang kurang dari 16 ternyata dapat dipecahkan dengan *known-plaintext attack* lebih mangkus daripada dengan *brute force attack*.

2.6.3. Kotak - S

Pengisian kotak-S DES masih menjadi misteri tanpa ada alasan mengapa memilih konstanta-konstanta di dalam kotak itu.

3. KESIMPULAN

Pengamanan selalu dibutuhkan, salah satunya di dunia digital. Salah satu yang banyak digunakan adalah *Data Encryption Standard (DES)* yang merupakan salah satu dari *cipher* blok.

DES diaplikasikan dengan melakukan pengolahan – pengolahan angka, oleh karena itu konsep dasar yang digunakan DES adalah teori bilangan.

Pada awalnya DES digunakan sebagai salah pengaman yang paling aman, tetapi seiring berjalannya waktu banyak kekurangan sistem ini yang menjadi kontroversi karena menyangkut kemanannya.

4. REFERENSI

[1]http://en.wikipedia.org/wiki/Data_Encryption_Standard ; Tanggal Akses : 20/12/09

[2]*csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf* ; Tanggal Akses : 20/12/09

[3]*kur2003.if.itb.ac.id/file/DES.doc* ; Tanggal Akses : 20/12/09

[4]www.google.co.id/url?sa=t&source=web&ct=res&cd=12&ved=0CDAQFjAL&url=http%3A%2F%2Fwww.informatika.org%2F~rinaldi%2FKriptografi%2F2006-2007%2FData%2520Encryption%2520Standard%2520(DES).ppt&rct=j&q=data+encryption+standard&ei=Jl8tS-3_B46OkQXM1o3gCw&usg=AFQjCNGmFVaYSDRU0Tve8nLWqKkLyp31dw ;Tanggal Akses : 20/12/09

[5]*belhob.wordpress.com/2007/12/14/the-des-**algorithm* ; Tanggal Akses : 20/12/09

[6]*en.wikipedia.org/wiki/**Lucifer**_(cipher)* ; Tanggal Akses : 20/12/09

[7]http://www.total.or.id/info.php?kk=Data%20Encryption%20Standard ;Tanggal Akses : 20/12/09