Jurnal CyberTech
Vol.x. No.x, Januari 2020, pp. xx~xx
E-ISSN: 2615-5133 ❒1

Implementasi Kriptografi Untuk Pengamanan Data Produksi Harian Dengan Algoritma *Data Encryption Standard* (DES) Pada PT. Cogindo

Budiman Nasra Laia* ,Nurcahyo Budi Nugroho S.Kom, M.Kom , Jufri Halim, S.E., M.M.*****
* Program Studi Mahasiswa, STMIK Triguna Dharma
** Program Studi Dosen Pembimbing, STMIK Triguna Dharma

Article Info

Article history:
Received Jun 12th, 201x
Revised Aug 20th, 201x
Accepted Aug 26th, 201x

Keyword:
Algoritma *DES*
simetris
Block Cipher
Ciphertext
Data Produksi harian
Enrollment
Visual Basic

ABSTRACT

Algoritma *Data Encryption Standard* (DES) merupakan algoritma simetris yaitu algoritma yang memiliki dua kunci yang sama untuk proses enkripsi dan dekripsinya. Algoritma ini banyak digunakan dalam permasalahan keamanan data. Salah satu pilihan untuk penerapan algoritma *Data Encryption Standard* (DES) dapat digunakan dengan metode sistem *Block Cipher*. Untuk meneliti kemampuan algoritma *Data Encryption Standard* (DES) dalam melakukan pengamanan data produksi harian yang ada di PT.Cogindo. Pengamanan data produksi harian yang dilakukan terkait produksi pagi, siang malam yang diterapkan proses enkripsi sehingga menghasilkan data yang tidak dapat dibaca lagi dalam bentuk *chipertext.* Proses enkripsi ini dilakukan menggunakan kunci internal sebanyak 64 bit menjadi 56 bit kunci internal. Untuk melihat data produksi harian adalah dengan mendekripsikan data terssebut kembali dengan kunci yang sama. Hasil dari analisa pengamanan data yang dilakukan kemudian diimplementasikan kedalam bentuk pemrograman Visual basic 2012.

Corresponding Author: *First Author
Nama : Budiman Nasra Laia
Program Studi Sistem Informasi
STMIK Triguna Dharma
Email: didimannasralaia@gmail.com

1. PENDAHULUAN

Teknologi Informasi telah menyebabkan perubahan dan cara pandang hidup manusia maupun suatu organisasi. Perkembangan yang sedemikian cepatnya membawa dunia memasuki era baru yang lebih cepat dari yang di bayangkan sebelumnya. Seperti komputer yang tidak hanya di gunakan sebagai pengolahan data saja, namun telah menjadi senjata utama dalam berkompetisi. Hal ini dikarenakan dengan adanya komputer dapat mempermudah dan mempercepat suatu pekerjaan dalam mengakses informasi [1].

Dalam sebuah perusahaan memiliki data atau informasi yang sangat penting dan perlu di lakukan untuk menjaga kerahasiaan dan keakuratan data tersebut. Oleh sebab itu berbagai perusahaan termaksud PT. Cogindo harus melakukan pengamanan data produksi agar data tersebut aman dan terjaga keakuratan datanya. Karena Kerahasiaan sebuah data merupakan aspek yang sangat penting bagi perusahaan untuk melindungi data atau informasi tersebut supaya tidak mudah jatuh ke tangan pihak lain dan tidak merugikan pemilik informasi. Mengingat Revolusi 4.0 merupakan sebuah perubahan digital di dunia industri [2]. Sehingga PT. Congindo sudah menggunakan komputer sebagai alat penunjang untuk mempermudah dan mempercepat suatu pekerjaan yang bersifat pengolahan data. Dan oleh sebab itu perlu adanya dilakukan pengamanan data untuk menjaga kerahasiaan dan keakuratan datatersebut [3].

Untuk menghindari hal-hal yang tidak di inginkan terjadi maka di butuhkan sebuah metode penyandian, ilmu sekaligus seni guna menjaga file yang disebut juga dengan Kriptografi [4]. Kriptografi merupakan teknik suatu metode dengan suatu kunci tertentu menggunakan mengolah informasi awal (*plain text)* yang tidak dapat dibaca baru (*cipher text*) suatu informasi menghasilkan enkripsi tertentu sehingga menjadi informasi awal (*plain text*) melalui tersebut dapat dikembalikan *cipher text* secara langsung sehingga orang lain tidak dapat mengenali data tersebut. Adapun proses penamaannya disebut proses *Enkripsi*. Data atau pesan yang asli sering disebut sebagai *plaintext* dan data yang telah dienkripsi disebut yang lebih tepat *encipher* [3].

Data Encryption Standard (DES) adalah salah satu metode kriptografi cipher blok yang popular digunakan karena tingginya tingkat keamanan informasi dan dijadikan standard algoritma enkripsi kunci-simetri. DES adalah nama standard enkripsi simetri yang dahulu memiliki nama algoritma enkripsinya DEA (*Data Encryption Algorithm*), namun nama DES lebih populer dari pada DEA. Keamanan algoritma DES terletak pada banyaknya proses enkripsi dan dekripsi yang dilakukan sebanyak 16 kali putaran. Setiap putarannya akan menggunakan kunci internal yang berbeda. Hasil dari proses enchiper kembali dipermutasi dengan matriks permutasi balikan (*invers initial permutation*) menjadi blok cipherteks.

Dari beberapa referensi kriptografi metode *Data Encrytion Standard* telah diterapkan untuk mengamankan data-data yang bersifat rahasia seperti keamanan informasi berbasis tanda tangan digital yang mana bertujuan untuk melakukan verifikasi apakah pesan atau informasi tersebut diterima dalam keadaan asli dari pengiriman atau telah dimodifikasi sehingga pesan atau informasi tersebut tidaklah asli **[**5**]**. dan DES juga mampu mengenkripsi dan dekripsi pesan atau informasi yang sangat rahasia dari orang-orang yang tidak bertanggung jawab dan tidak berkepentingan. Dari beberapa refrensi diatas dapat disimpulkan metode *Data Encrytion Standard* bisa dijadikan sebagai solusi untuk mengamankan data produksi harian yang bersifat rahasia pada PT. Cogindo.

Harapannya sebuah sistem yang mengadopsi *Data Encryption Standard* dapat di implementasikan di PT. Cogindo. Sistem tersebut akan membantu pihak PT. Cogindo untuk mengamankan data produksi harian. Sehingga data tersebut terjaga kerahasiaan dan keakuratan data tersebut. Berdasarkan deskripsi di atas maka penelitian ini diangkatlah sebuah judul "Implementasi Kriptografi Untuk Pengamanan Data Produksi Harian Dengan Algoritma *Data Encryption Standard* (DES)*"* Pada PT. Cogindo.

2. METODE PENELITIAN

Dalam teknik pengumpulan data dilakukan dengan dua tahapan, diantaranya yaitu:

2.1 Pengumpulan Data (*Data Collecting*)

1. Observasi

Observasi merupakan salah satu teknik dalam pengumpulan data yang kompleks. Dalam penelitian ini observasi dilakukan untuk mendapatkan data di PT Cogindo. Hal ini bertujuan untuk memperoleh informasi tentang data yang akan digunakan dalam penelitian ini. Data yang didapatkan di PT.Cogindo adalah data produksi harian.

2.2 Studi Kepustakaan (*Study of Literature*)

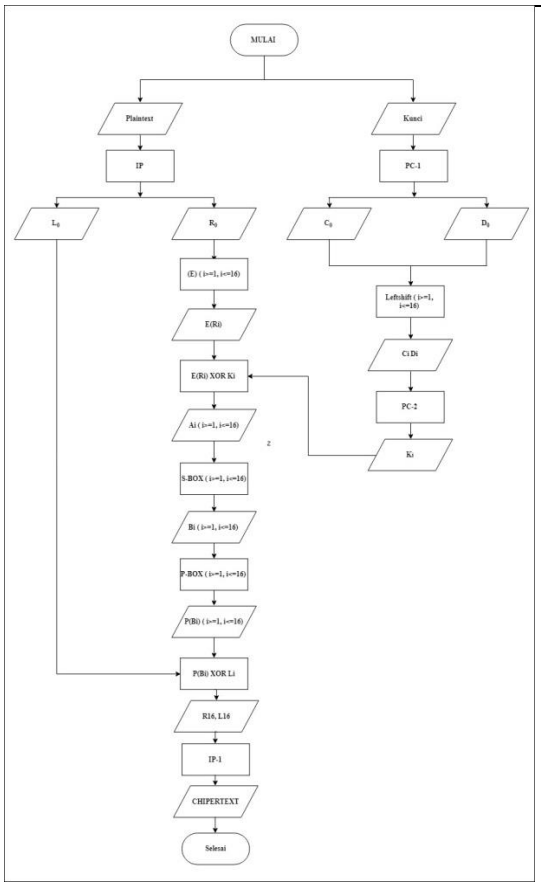
Dalam penelitian ini banyak menggunakan jurnal-jurnal baik jurnal nasional maupun buku sebagai sumber referensi. Dari komposisi yang ada jumlah literatur yang digunakan sebanyak 20 dengan rincian: 19 jurnal nasional, dan 1 buku nasional.

2.3 Algoritma Sistem

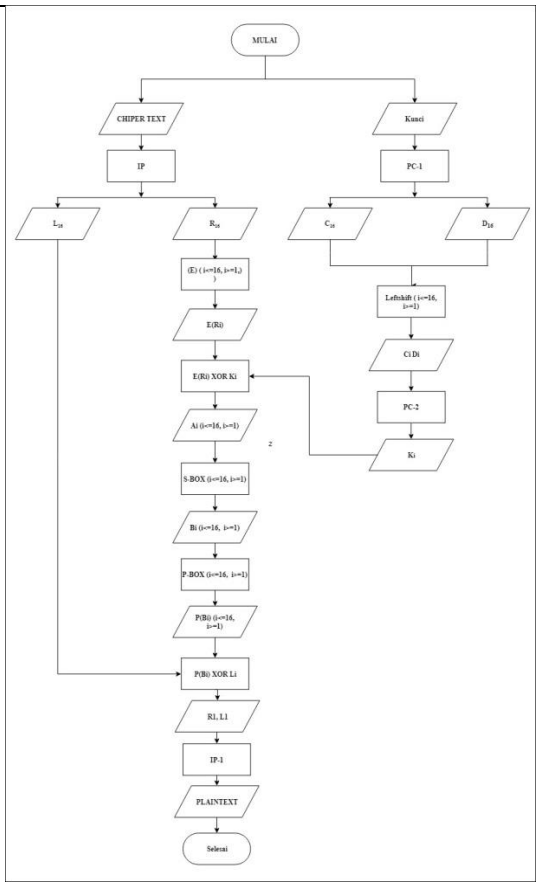
Algoritma sistem merupakan penjelasan langkah-langkah dalam penyelesaian masalah dalam perancangan sistem keamanan data produksi harian dengan menggunakan algoritma DES. Hal ini dilakukan untuk meningkatkan keamanan data produksi harian tersebut.

2.3.1 Flowchart dari Metode Penyelesaian

Berikut ini adalah *flowchart* dari proses enkripsi dan dekripsi dari algoritma DES yaitu sebagai berikut:



Gambar 3.1 *flowchart* proses enkripsi



Gambar 3.2 *flowchart* Proses Dekripsi

2.4 Dekripsi Data Dari Penelitian

Tabel 3.1 Data Produksi harian

PRODUKSI	PAGI	SIANG	MALAM	RATA-RATA
Total Produksi 1st RO (m3)	903	512	920	778
Total Produksi 2nd RO (m3)	391	188	342	307
Total Produksi Mixed Bed (m3)	345	118	476	313
Total Supply DM (m3)	213	192	248	218

2.5 Dekripsi Data Dari Penelitian

Sesuai dengan referensi yang telah dipaparkan pada bab sebelumnya, berikut ini adalah langkah- langkah penyelesaiannya yaitu:

2.5.1 Proses Enkripsi DES

Proses enkripsi algoritma DES, ada delapan tahapan berikut adalah tahapan-tahapan yang terdapat pada algoritma DES

Proses enkripsi adalah mengubah suatu data plaintext ke chippertext. Dalam proses enkripsi terdapat beberapa langkah-langkah berikut:

1. Mengubah plaintext dan key menjadi bilangan biner

Mengubah plaintext kedalam biner berdasarkan tabel ASCII.

Tabel 3.2 Konversi *plaintext* ke biner

PLAINTEXT			
P	DEC	HEXA	BINER
9	57	39	00111001
0	48	30	00110000
3	51	33	00110011
	48	30	00110000
	48	30	00110000
	48	30	00110000

Tabel 3.2 Konversi *plaintext* ke biner (Lanjutan)

	48	30	00110000
	48	30	00110000

Mengubah *key* ke dalam biner berdasarkan tabel ASCII

Tabel 3.3 Konversi *key* ke biner

KEY			
K	**DEC**	**HEXA**	**BINER**
C	67	43	01000011
O	79	4F	01001111
G	71	47	01000111
I	73	49	01001001
N	78	4E	01001110
D	68	44	01000100
O	79	4F	01001111
1	49	31	00110001

2. Initial Permutation (IP) plaintext

Lakukan *initial permutation* (IP) pada bit plaintext menggunakan tabel IP seperti berikut:

Tabel 3.4 *initial permutation*

PLAINTEXT (X)							
0	0	1	1	1	0	0	1
0	0	1	1	0	0	0	0
0	0	1	1	0	0	1	1
0	0	1	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	1	1	0	0	0	0

IP1							
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

Keterangan pada tabel *initial permutation* dan tabel IP(X):
Angka 0 dan 1 merupakan bilangan biner
Angka 1,2,3 dan seterusnya yang menggunakan penebalan adalah urutan posisi bit
Urutan bit ke-58 pada tabel *plaintext* (X), di letakan pada posisi 1 pada tabel IP,
Urutan bit ke-50 pada tabel *plaintext* (X), di letakan pada posisi 2 pada tabel IP,
Urutan bit ke-42 pada tabel *plaintext* (X), di letakan pada posisi 3 pada tabel IP,
Demikian seterusnya dan menghasilkan Tabel IP(X).

Tabel 3.5 Tabel IP(X)

TABEL IP (X)								
0	0	0	0	0	0	0	0	**L0**
1	1	1	1	1	1	1	1	
0	0	0	0	0	0	0	0	
0	0	0	0	0	1	0	1	
0	0	0	0	0	0	0	0	**R0**
1	1	1	1	1	1	1	1	
0	0	0	0	0	0	0	1	
0	0	0	0	0	1	0	0	

Selanjutnya bit pada IP(X) di pecah menjadi dua bagian yaitu L0 dan R0 sehingga hasilnya dapat di lihat pada tabel 3.5.

3. Melakukan Permutasi Key Kompresi PC-1

Kunci yang sudah diubah menjadi bilangan biner, lalu di permutasikan dengan menggunakan tabel permutasi kompresi PC-1, pada langkah ini terjadi kompresi 64 bit menjadi 56 bit.

Tabel 3.6 Tabel Permutasi Kompresi PC-1

KEY							
0	1	0	0	0	0	1	1
0	1	0	0	1	1	1	1
0	1	0	0	0	1	1	1

PC1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27

Tabel 3.6 Tabel Permutasi Kompresi PC-1(Lanjutan)

0	1	0	0	1	0	0	1
0	1	0	0	1	1	1	0
0	1	0	0	0	1	0	0
0	1	0	0	1	1	1	1
0	0	1	1	0	0	0	1

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

Keterangan pada tabel Permutasi Kompresi PC-1
Angka 0 dan 1 merupakan bilangan biner
Angka **1,2,3** dan seterusnya yang menggunakan penebalan adalah urutan posisi bit
Urutan bit ke-57 pada tabel key, diletakan pada posisi 1 pada Tabel PC-1,
Urutan bit ke-49 pada tabel key, diletakan pada posisi 2 pada Tabel PC-1 dst, dan hasil permutasi key dapat di lihat padatabel 3.7.

Tabel 3.7 Tabel Permutasi Kompresi PC-1

TABEL PC-1							
0	0	0	0	0	0	0	**C0**
0	0	1	1	1	1	1	
1	1	1	0	0	0	0	
0	0	0	1	0	0	0	
0	1	0	1	0	1	1	**D0**
1	0	1	1	1	0	1	
1	0	0	1	0	1	1	
0	1	0	0	0	0	0	

Selanjutnya bit pada Tabel hasil permutasi PC-1 di pecah menjadi dua bagian yaitu C0 dan D0 sehingga hasilnya sebagai berikut.
C0: 0000000 0011111 1110000 0001000
D0: 0101011 1011101 1001011 0100000

4. Melakukan Pergeseran Kiri (Left Shift Operation)

Lakukan pergeseran kiri (*left Shift Operation*) pada C0 dan D0 sebanyak satu atau dua kali berdasarkan putaran yang ada pada tabel putaran sebagai berikut:

Tabel 3.8 Tabel *Left shif*

Putaran ke – i	**Jumlah Pergeseran Bit (Left Shift)**
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Keterangan:
Untuk putaran ke-1, dilakukan pergeseran 1bit ke kiri,
Untuk putaran ke-2, dilakukan pergeseran 1 bit ke kiri,

Untuk putaran ke-3, dilakukan pergeseran 2 bit ke kiri, dan seterusnya hingga putaran yang ke-16.
Berikut hasil dari *left shift*:
Putaran ke-1, di geser 1 bit ke kiri.
C1: 0000000 0111111 1100000 0010000
D1: 1010111 0111011 0010110 1000000
Putaran ke-2, di geser 1 bit ke kiri.
C2: 0000000 1111111 1000000 0100000
D2: 0101110 1110110 0101101 0000001
Putaran ke-15, di geser 2 bit ke kiri.
C15: 0000000 0001111 1111000 0000100
D15: 0010101 1101110 1100101 1010000
Putaran ke-16, di geser 1 bit ke kiri.
C16: 0000000 0011111 1110000 0001000
D16: 0101011 1011101 1001011 0100000

Tabel 3.9 Tabel *Permutation Compression* 2 (PC-2)

PC-2					
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

Keterangan:
Urutan bit pada C_iD_i yang ke-14, diletakan di posisi 1 pada tabel PC-2,
Urutan bit pada C_iD_i yang ke-17, diletakan di posisi 2 pada tabel PC-2,
Urutan bit pada C_iD_i yang ke-11, diletakan di posisi 3 pada tabel PC-2, dan seterusnya.
Berikut hasil *output*nya:
C_1D_1: 0000000 0111111 1100000 0010000 1010111 0111011 0010110 1000000
K_1: 101100 001001 001000 100010 101110 000111 001010 011010
C_2D_2: 0000000 1111111 1000000 0100000 0101110 1110110 0101101 0000001
K_2: 101000 001001 101001 000010 100110 110010 110110 100101
C_3D_3: 0000011 1111110 0000001 0000000 0111011 1011001 0110100 0000101
K_3: 001000 000111 001001 010010 001010 100100 101110 010101
C_4D_4: 0001111 1111000 0000100 0000000 1101110 1100101 1010000 0010101
K_4: 001001 000101 010101 010000 010100 110110 000110 010111
C_5D_5: 0111111 1100000 0010000 0000000 0111011 0010110 1000000 1010111
K_5: 010001 100100 000101 010001 111001 110000 000110 001001
C_6D_6: 1111111 0000000 1000000 0000001 1101100 1011010 0000010 1011101
K_6: 000011 111100 000100 010001 110000 100011 001101 001111
C_7D_7: 1111100 0000010 0000000 0000111 0110010 1101000 0001010 1110111
K_7: 000011 110000 000110 001011 011101 101001 001110 101100
C_8D_8: 1110000 0001000 0000000 0011111 1001011 0100000 0101011 1011101
K_8: 001110 110000 000010 001001 010100 000001 110111 101011
C_9D_9: 1100000 0010000 0000000 0111111 0010110 1000000 1010111 0111011
K_9: 000110 010001 100010 001001 011011 001111 010111 000100
$C_{10}D_{10}$: 0000000 1000000 0000001 1111111 1011010 0000010 1011101 1101100
K_{10}: 000100 010010 100011 001000 101010 001100 010011 101011
$C_{11}D_{11}$: 0000010 0000000 0000111 1111100 1101000 0001010 1110111 0110010
K_{11}: 000100 000110 110010 000100 110011 101101 111000 000011
$C_{12}D_{12}$: 0001000 0000000 0011111 1110000 0100000 0101011 1011101 1001011
K_{12}: 010100 000010 110100 000100 100111 100100 011101 111000
$C_{13}D_{13}$: 0100000 0000000 1111111 1000000 0000001 0101110 1110110 0101101
K_{13}: 010000 001010 010000 100101 100110 011101 101101 000000
$C_{14}D_{14}$: 0000000 0000011 1111110 0000001 0000101 0111011 1011001 0110100

K_{14}: 110000 011000 010000 100110 110100 001110 011000 110000
$C_{15}D_{15}$: 0000000 0001111 1111000 0000100 0010101 1101110 1100101 1010000
K_{15}: 111000 001000 001010 100010 111110 010010 111000 001100
$C_{16}D_{16}$: 0000000 0011111 1110000 0001000 0101011 1011101 1001011 0100000
K_{16}: 111000 001001 001000 100010 000000 111001 011010 11010

5. Melakukan Ekspansi Data

Pada langkah ini, kita akan meng-ekspansi data R_{i-1} 32 bit menjadi R_i 48 bit sebanyak 16 kali putaran dengan nilai perputaran 1>= i <=16 menggunakan Tabel Ekspansi (E).

Tabel 3.10 Tabel Ekspansi

Tabel Ekspansi					
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

Hasil $E(R_{i-1})$ kemudian di XOR dengan K_i dan menghasilkan Vektor Matriks A_i. Berikut hasil *output*nya:

Iterasi 1

E((R1)-1) : 000000 000001 011111 111110 100000 000010 100000 001000
K1 : 101100 001001 001000 100010 101110 000111 001010 011010
----------XOR
A1 : 101100 001000 010111 011100 001110 000101 101010 010010

Pada iterasi satu (1) diatas didapat A1 dari hasil XOR E(R1-1) dan K1, setelah itu maka proses selanjutnya langsung ke langkah ke-6 terlebih dahulu, dimana $A_{i,}$ akan dimasukan ke dalam S-BOX dan menghasilkan PB1 yang kemudian di XOR kan dengan L0 dan menghasilkan nilai R_i. Nilai R_i ini digunakan untuk melanjutkan iterasi ke-2.

6. Memasukan Data Ke Dalam S-BOX

A1 : 101100 001000 010111 011100 001110 000101 101010 010010

Tabel 3.11 Tabel Subtitusi S1

	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
01	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
10	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
11	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Kemudian kita ambil sampel blok bit pertama yaitu 101100, pisahkan menjadi 2 blok yaitu:

1. Bit pertama dan terakhir yaitu 1 dan 0, digabungkan menjadi 10
2. Bit kedua hingga kelima yaitu 0110

Selanjutnya dibandingkan dengan memeeriksa perpotongan antara kedua di dapatkan nilai 2 (warna kuning) lalu dibinerkan menjadi 0010

Tabel 3.12 Tabel Subtitusi S2

	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
01	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
10	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
11	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

Kemudian kita ambil sampel blok bit kedua yaitu 001000, pisahkan menjadi 2 blok yaitu:

1. Bit pertama dan terakhir yaitu 0 dan 0, digabungkan menjadi 0
2. Bit kedua hingga kelima yaitu 0100

Selanjutnya dibandingkan dengan memeeriksa perpotongan antara kedua di dapatkan nilai 6 (warna kuning) lalu dibinerkan menjadi 0110. Dan seterusnya untuk blok ketiga hingga blok kedelapan dibandingkan dengan S3 dan S8. Berdasrkan cara diatas diperoleh hasil sebagai berikut:

B1 = 00100110 11100100 01100100 00111001

7. Memutasikan Bit Vektor B_i

Setelah didapatkan nilai vector B_i, langkah selanjutnya adalah memutasikan bit vektor B_i menggunakan tabel P-*BOX*, lalu dikelompokkan menjadi 4 blok dimana setiap blok memiliki 32 bit data

Tabel 3.13 Tabel Matrik Permutasi P (P-*box*)

Matrik Permutasi							
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

Sehingga hasil yang didapatkan sebagai berikut:

P(B1) : 01111000 00100011 00111111 00011100

Hasil $P(B_i)$ kemudian di XOR kan dengan L_i-1 untuk mendapatkan nilai R_i. Sedangkan nilai L_i sendiri diperoleh dari nilai R_i-1 untuk nilai 1<=i<=16.

L0 : 00000000 11111111 00000000 00000101
R0 : 00000000 11111111 00000001 00000100
P(B1) : 01001010 00000101 00011111 10011100
L0 : 00000000 11111111 00000000 00000101
----------XOR
R1 : 01001010 11111010 00011111 10011001

Untuk mencari R2 sampai R16, lakukan langkah yang sama dari langkah 5 sampai 7dan dituliskan dalam bentuk itrasi.

Iterasi 2

E((R2)-1) : 101001 010101 011111 110100 000011 111111 110011 110010
K2 : 101000 001001 101001 000010 100110 110010 110110 100101
----------XOR
A2 : 000001 011100 110110 110110 100101 001101 000101 010111
B2 : 00000101 11001110 11001001 10111011
P(B2) : 00011011 01000111 01111101 01010001
L(2)-1 : 00000000 11111111 00000001 00000100
----------XOR
R2 : 00011011 10111000 01111100 01010101

Iterasi 3

E((R3)-1) : 100011 110111 110111 110000 001111 111000 001010 101010
K3 : 001000 000111 001001 010010 001010 100100 101110 010101
----------XOR
A3 : 101011 110000 111110 100010 000101 011100 100100 111111
B3 : 10010101 01110110 00100101 10111011
P(B3) : 00001110 11000011 01111100 10011111
L(3)-1 : 01001010 11111010 00011111 10011001
----------XOR
R3 : 01000100 00111001 01100011 00000110

Iterasi 15

E (R14) : 001011 111100 000001 010001 011011 111001 010000 000100
K15 : 111000 001000 001010 100010 111110 010010 111000 001100
----------XOR
A15 : 110011 110100 001011 110011 100101 101011 101000 001000
B15 : 10111100 01000100 11000101 11000110
P(B15) : 00000001 10011111 00110010 00111011
L(15)-1 : 00000001 10011111 00110010 00111011
----------XOR
R15 : 0001011 100111001 01100010 10101111

Iterasi 16

E (R15) : 100010 101110 100111 110010 101100 000101 010101 011110
K16 : 111000 001001 001000 100010 000000 111001 011010 110101
----------XOR
A16 : 011010 100111 101111 010000 101100 111100 001111 101011
B16 : 10010001 01110001 01111011 10101010
P(B16) : 10111100 10100111 01100100 10000111
L(16)-1 : 01011110 00001000 11011100 10000010
----------XOR
R16 : 11100010 10101111 10111000 00000101
L(16)-1 : 01011110 00001000 11011100 10000010

8. Menggabungkan R16 dan L16

Langkah terakhir adalah menggabungkan R_{16} dengan L_{16} kemudian dipermutasikan dengan tabel *initial permutation* (IP^{-1}).

Tabel 3.14 Tabel Permutasi R16 dan L16 dengan Tabel IP^{-1}

R16 dan L16								
R16	1	1	1	0	0	0	1	0
	1	0	1	0	1	1	1	1

TABEL IP-1							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31

	1	0	1	1	1	0	0	0		38	6	46	14	54	22	62	30
	0	0	0	0	0	1	0	1		37	5	45	13	53	21	61	29
L16	0	0	0	1	0	1	1	1		36	4	44	12	52	20	60	28
	0	0	1	1	1	0	0	1		35	3	43	11	51	19	59	27
	0	1	1	0	0	0	1	0		34	2	42	10	50	18	58	26
	1	0	1	0	1	1	1	1		33	1	41	9	49	17	57	25

Tabel 3.15 Tabel *Chipertext*

CHIPERTEKS							
1	0	1	1	0	0	1	1
1	1	0	1	1	0	1	0
1	0	0	1	0	0	1	1
0	0	1	1	0	1	1	0
1	0	1	0	0	1	0	0
0	1	1	1	1	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	0	1	1	0

Menghasilkan *output*:
Chiper dalam biner : **10110011 11011010 10010011 00110110 10100100 01111110 01001000 01010110**
Atau dalam *chiper* hexa : **B3 DA 93 36 A4 7E 48 56**
Atau dalam bentuk *plaintext* : **³ Ú “ 6 ¤ ~ H V**

2.5.2 Proses Dekripsi DES

Untuk dapat mengetahui isi pesan sebenarnya, perlu dilakukan konversi *ciphertext* menjadi bentk biner untuk mendapatkan bit *chipertext*. Dekripsi dapat dilakukan sebagai berikut:

1. Melakukan permutasi terhadap Cipher

Cipher dalam biner**: 10110011 11011010 10010011 00110110 10100100 01111110 01001000 01010110**
Atau dalam hexa : **B3 DA 93 36 A4 7E 48 56**
Atau dalam bentuk *plaintext* : **³ Ú “ 6 ¤ ~ H V**

Tabel 3.16 Tabel *initial permutation chipper* (IP)

Ciphertext							
1	0	1	1	0	0	1	1
1	1	0	1	1	0	1	0
1	0	0	1	0	0	1	1
0	0	1	1	0	1	1	0
1	0	1	0	0	1	0	0
0	1	1	1	1	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	0	1	1	0

Tabel IP							
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

Tabel 3.17 Tabel hasil *initial permutation chipper* (IP)

IP(Cipher)								
1	1	1	0	0	0	1	0	**L0**
1	0	1	0	1	1	1	1	
1	0	1	1	1	0	0	0	
0	0	0	0	0	1	0	1	
0	0	0	1	0	1	1	1	R0
0	0	1	1	1	0	0	1	
0	1	1	0	0	0	1	0	
1	0	1	0	1	1	1	1	

Selanjutnya bit pada IP (*Chiper*) dipecah menjadi 2 bagian yaitu L0 dan R0,
Sehingga menghasilkan sebagai berikut:
L0 : 11100010 10101111 10111000 00000101
R0 : 00010111 00111001 01100010 10101111

Iterasi **16**

P(B16) : 10111100 10100111 01100100 10000111
L15 : 11100010 10101111 10111000 00000101
----------XOR
R16 : 01011110 00001000 11011100 10000010
Iterasi 15
P(B15) : 00000001 10011111 00110010 00111011
L14 : 00010111 00111001 01100010 10101111
----------**XOR**
R15 : 00010110 10100110 01010000 10010100
Iterasi 2
P(B2) : 00011011 01000111 01111101 01010001
L1 : 00011011 10111000 01111100 01010101
----------XOR
R2 : 00000000 11111111 00000001 00000100
Iterasi 1
P(B1) : 01001010 00000101 00011111 10011100
L0 : 01001010 11111010 00011111 10011001
----------XOR
R1 : 00000000 11111111 00000000 00000101
L1 : 00011011 10111000 01111100 01010101

2. Melakukan Permutasi R1 dan L1 Dengan Tabel IP-1

Kemudian R_1 dan L_1 di permutasikan kembali dengan tabel *inverse initial permutation* sehingga menghasilkan *output*:
Plaintext dalam biner : **00111001 00110000 00110011 00110000 00110000 00110000 00110000 00110000**
Atau dalam bentuk hexa: **57 48 51 48 48 48 48 48**
Dan dalam bentuk *plaintext:* **903**

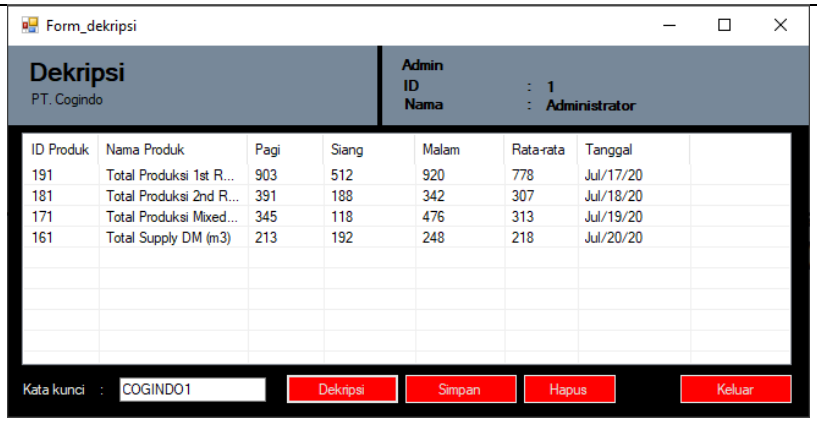
3. Analisa dan Hasil

3.1 Pengujian Sistem

Uji coba sistem bertujuan untuk membuktikan bahwa *input*, *proses*, *output* yang dihasilkan oleh sistem aplikasi *Visual Studio* 2012 telah benar dan sesuai dengan yang diinginkan. Pengujian sistem dengan cara memasukkan data ke dalam sistem dan memperhatikan *output* yang dihasilkan. Jika *input*, proses dan *output* telah sesuai, maka sistem telah benar. Berikut merupakan tahapan untuk pengujian sistem yaitu:

1. Melakukan *input* data produksi harian yang kemudian sistem akan menampilkan data produksi harian yang tersimpan di *database*.
2. Menggunakan bahasa pemrograman *Microsoft Visual Studio* 2012 dalam pengolahan data yang disimpan dalam *database Microsoft Office Access* 2010 . Penggunaan sistem pengamanan data produksi harian pada PT.Cogindo, agar dapat berjalan dengan baik *file* aplikasi *Visual Studio* 2012 harus ditempatkan pada satu *folder* dan dilengkapi dengan *input* data dari analisa sistem. Lokasi *folder* yang telah ditentukan adalah tempat untuk menyimpan *file-file* yang telah dikumpulkan, untuk menghindari kesalahan sebaiknya data tidak diletakkan kedalam *folder* yang berbeda. Selanjutnya untuk menerapkan metode dalam mengamankan data produksi harian, maka data tersebut akan *diinput* ke aplikasi lalu simpan data tersebut ke dalam *database Access*. Jalankan aplikasi *Visual Studio* 2012 yang telah terinstal dikomputer. Berikut ini merupakan hasil pengujian yang dilakukan pada sistem.

Gambar 5. Pengujian untuk data produksi harian enkripsi



Gambar 6 Pengujian untuk data produksi harian dekripsi

3.2 Kelemahan dan Kelebihan Sistem

Berikut ini diuraikan kelemahan dan kelebihan dari sistem:

1. Kelemahan Sistem
   Dalam sistem tentunya masih ada kekurangan dan kelemahan. Adapun kelemahan yang ada di dalam sistem adalah:
   a. Sistem yang dibangun tidak dapat diakses secara online, sehingga sistem hanya dapat digunakan secara lokal saja.
   b. Hasil ini hanya digunakan pada kasus di PT.Cogindo, tidak di perusahaan lain.
2. Kelebihan Sistem
   Hasil yang didapat dari pengujian sistem ini mempunyai kelebihan- kelebihan antara lain :
   a. Proses Pengamanan Data
      Bagi pengguna sistem khususnya pada PT.COGINDO yang ingin menggunakan sistem ini, cukup meng*input* data produksi harian yang akan dijadikan sebagai objek pengamanan data, kemudian melakukan proses enkripsi, maka hasil yang di dapat yaitu sebuah *cipherteks*, data produksi harian tersebut diamankan dengan menggunakan kombinasi kriptografi algoritma *data encryption standard* sehingga sulit untuk mengetahui dan membaca data produksi harian tersebut.
   b. Menjalankan Program
      Program yang dibangun bersbasis *desktop programming*, walaupun tidak terhubung jaringan ataupun internet sistem tetap dapat untuk dijalankan. Dapat membantu pihak karyawan pada PT.COGINDO dalam mengamankan data produksi harian.

4. Kesimpulan

Berdasarkan pembahasan dan evaluasi dari bab sebelumnya, maka dapat ditarik kesimpulan sebagai berikut :

1. Dalam menganalisa masalah yang terjadi terkait dengan pengamanan data produksi harian di PT. Cogindo menggunakan algotritma *Data Encryption Standard* (DES) maka dilakukan proses enkripsi untuk data produksi harian baik produksi pagi, siang, dan malam.
2. Perancang sistem kriptografi yang mengadopsi algoritma DES (*Data Encryption Standard)* dengan metode sistem *Block Cipher* di dalam menyelesaikan masalah terkait pengamanan data produksi harian di PT. Cogindo menggunakan pemrograman yang berbasis desktop yaitu *Visual Basic*.
3. Pengimplementasikan sistem kriptografi yang terintegrasi dengan sistem yang berbasis *Visual Basic* 2012 dan Microsoft Acsess 2010 dapat dilakukan dalam menyelesaikan masalah terkait pengamanan data produksi harian di PT. Cogindo.
4. Pengujian sistem ini dilakukan sebelum nantinya dapat dicoba untuk membantu instansi-instansi terkait di dalam pengamanan data produksi harian di PT. Cogindo.

Referensi

[1] David Pratama Pahrizal, *IMPLEMENTASI ALGORITMA RSA UNTUK PENGAMANAN DATA BERBENTUK TEKS*, p. 44, Feb. 2016.

[2] Murti Ningsih, *PENGARUHPERKEMBANGAN REVOLUSI INDUSTRI 4.0DALAM DUNIA TEKNOLOGI DI INDONESIA*, pp. 2-9.

[3] Fachruddin,Dina Fitra Murad,Hetty Rohayani AH,Pandapotan S Erick Fernando, "AnalisaDanImplementasiAlgoritmaEnkripsiSimetrisDataEncryptionStandard(DES)PadaRaspberryPi," vol. XI, pp. 55-56, mei 2019.

[4] Deny adhar, *IMPLEMENTASI ALGORITMA DES (DATA ENCRYPTIONSTANDARD) PADAENKRIPSIDANDESKRIPSISMSBERBASISANDROID*, vol. 3, pp. 53-57, Juli 2019.

[5] Oris Krianto Sulaiman, Mohamad Ihwani, and Salman Fajar Rizki, "MODEL KEAMANAN INFORMASI BERBASIS TANDA TANGAN DIGITAL DENGAN DATA ENCRYPTION STANDARD (DES) ALGORITHM,".

BIOGRAFI PENULIS

	Nama : Budiman Nasra laia Tempat Lahir : Medan Tanggal Lahir : 23 Agustus 1997 Jenis Kelamin : Laki-Laki Agama : Kristen Warga Negara : Indonesia Status : Lajang Alamat : Jl.Teratai Gg. Palam
Second author's photo(3x4cm)	Nama : Nurcahyo Budi Nugroho S.Kom, M.Kom
Thirth author's photo(3x4cm)	Nama : Jufri Halim, S.E.,M.M

NB : Untuk Second dan Thirth Author's dap at di kosongkan dan cukup isikan nama author