**Modul 10**

1. **Tujuan**

1. Mampu menjelaskan definisi dan konsep Data Encryption Standard (DES)

2. Mampu memahami konsep dasar Data Encryption Standard (DES)

1. **Dasar teori**
2. **Data Encryption Standard**

DES merupakan salah satu algoritma kriptografi cipher block dengan ukuran blok 64 bit dan ukuran kuncinya 56 bit. Algoritma DES dibuat di IBM, dan merupakan modifikasi daripada algoritma terdahulu yang bernama Lucifer. Lucifer merupakan algoritma cipher block yang beroperasi pada blok masukan 64 bit dan kuncinya berukuran 28 bit. Pengurangan jumlah bit kunci pada DES dilakukan dengan alasan agar mekanisme algoritma ini bisa diimplementasikan dalam satu chip.

1. **Algoritma DES**
2. **Contoh Enkripsi**

Plaintext(x) = COMPUTER

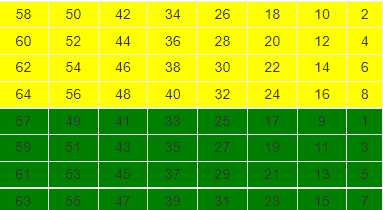
Key(k) = 13 34 57 79 9B BC DF F1

1. **Langkah Pertama (Konversi Teks dan Key ke biner)**

|  |  |
| --- | --- |
| Ubahlah plaintext kedalam bentuk biner  C : 01000011  O : 01001111  M : 01001101  P : 01010000  U : 01010101  T : 01010100  E : 01000101  R : 01010010 | Ubahlah key kedalam bentuk biner  13 : 00010011  34 : 00110100  57 : 01010111  79 : 01111001  9B : 10011011  BC : 10111100  DF : 11011111  F1 : 11110001 |

1. **Langkah Kedua (Initial Permutation)**

Tabel Initial Permutation



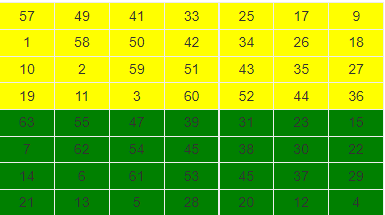
Urutan bit pada plaintext urutan ke 58 ditaruh diposisi 1,  
Urutan bit pada plaintext urutan ke 50 ditaruh di posisi 2,  
Urutan bit pada plaintext urutan ke 42 ditaruh di posisi 3, dst  
Sehingga hasil outputnya adalah  
IP(x) : 11111111 10111000 01110110 01010111 00000000 00000000 00000110 10000011

Pecah bit pada IP(x) menjadi 2 bagian yaitu:  
**L0 : 11111111 10111000 01110110 01010111 (tabel IP dengan warna kuning)  
R0 : 00000000 00000000 00000110 10000011 (tabel IP dengan warna hijau)**

1. **Langkah Ketiga (Generate Kunci)**

Generate kunci yang akan digunakan untuk mengenkripsi plaintext dengan menggunakan tabel permutasi kompresi PC-1, pada langkah ini terjadi kompresi dengan membuang 1 bit masing-masing blok kunci dari 64 bit menjadi 56 bit.

Tabel PC-1



Dapat kita lihat pada tabel diatas, tidak terdapat urutan bit 8,16,24,32,40,48,56,64 karena telah dikompress. Berikut hasil outpunya :

CD(k) : 1111000 0110011 0010101 0101111 0101010 1011001 1001111 0001111

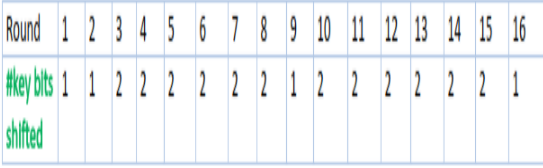
Pecah CD(k) menjadi dua bagian kiri dan kanan, sehingga menjadi

C0 : 1111000 0110011 0010101 0101111(tabel PC-1 warna kuning)

D0 : 0101010 1011001 1001111 0001111 (tabel PC-1 warna hijau)

1. **Langkah Keempat (Pergeseran bit biner pada kunci)**

Lakukan pergeseran kiri (Left Shift) pada C0 dan D0, sebanyak 1 atau 2 kali berdasarkan putaran yang ada pada tabel putaran sebagai berikut:



Tabel Pergeseran

Keterangan :

Untuk putaran ke 1, dilakukan pegeseran 1 bit ke kiri

Untuk putaran ke 2, dilakukan pergeseran 1 bit kekiri

Untuk putaran ke 3, dilakukan pergeseran 2 bit kekiri, dst

Maka hasilnya adalah

|  |  |  |  |
| --- | --- | --- | --- |
| Turn | C0 dan D0 | Turn | C0 dan D0 |
| 1 | C1 : 1110000 1100110 0101010 1011111 D1 : 1010101 0110011 0011110 0011110 | 9 | C9 : 0101010 1011111 1110000 1100110 D9 : 0011110 0011110 1010101 0110011 |
| 2 | C2 : 1100001 1001100 1010101 0111111 D2 : 0101010 1100110 0111100 0111101 | 10 | C10 : 0101010 1111111 1000011 0011001 D10 : 1111000 1111010 1010101 1001100 |
| 3 | C3 : 0000110 0110010 1010101 1111111 D3 : 0101011 0011001 1110001 1110101 | 11 | C11 : 0101011 1111110 0001100 1100101 D11 : 1100011 1101010 1010110 0110011 |
| 4 | C4 : 0011001 1001010 1010111 1111100 D4 : 0101100 1100111 1000111 1010101 | 12 | C12 : 0101111 1111000 0110011 0010101 D12 : 0001111 0101010 1011001 1001111 |
| 5 | C5 : 1100110 0101010 1011111 1110000 D5 : 0110011 0011110 0011110 1010101 | 13 | C13 : 0111111 1100001 1001100 1010101 D13 : 0111101 0101010 1100110 0111100 |
| 6 | C6 : 0011001 0101010 1111111 1000011 D6 : 1001100 1111000 1111010 1010101 | 14 | C14 : 1111111 0000110 0110010 1010101 D14 : 1110101 0101011 0011001 1110001 |
| 7 | C7 : 1100101 0101011 1111110 0001100 D7 : 0110011 1100011 1101010 1010110 | 15 | C15 : 1111100 0011001 1001010 1010111 D15 : 1010101 0101100 1100111 1000111 |
| 8 | C8 : 0010101 0101111 1111000 0110011 D8 : 1001111 0001111 0101010 1011001 | 16 | C16 : 1111000 0110011 0010101 0101111 D16 : 0101010 1011001 1001111 0001111 |

Setiap hasil tiap putaran kemudian digabungkan (Ci dan Di digabung) kembali. Kemudian di permutasi lagi menggunakan tabel kompresi permutasi PC-2. Kunci yang awalnya 56 bit akan di kompres menjadi 48 bit.

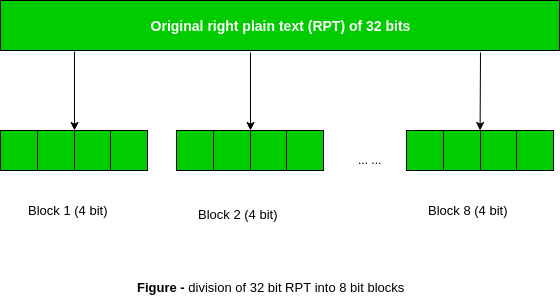


Maka Hasilnya adalah sebagai berikut :

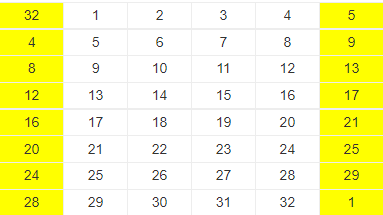
|  |  |  |  |
| --- | --- | --- | --- |
| K | Hasil | K | Hasil |
| 1 | 000110 110000 001011 101111 111111 000111 000001 110010 | 9 | 111000 001101 101111 101011 111011 011110 011110 000001 |
| 2 | 011110 011010 111011 011001 110110 111100 100111 100101 | 10 | 101100 011111 001101 000111 101110 100100 011001 001111 |
| 3 | 010101 011111 110010 001010 010000 101100 111110 011001 | 11 | 001000 010101 111111 010011 110111 101101 001110 000110 |
| 4 | 011100 101010 110111 010110 110110 110011 010100 011101 | 12 | 011101 010111 000111 110101 100101 000110 011111 101001 |
| 5 | 011111 001110 110000 000111 111010 110101 001110 101000 | 13 | 100101 111100 010111 010001 111110 101011 101001 000001 |
| 6 | 011000 111010 010100 111110 010100 000111 101100 101111 | 14 | 010111 110100 001110 110111 111100 101110 011100 111010 |
| 7 | 111011 001000 010010 110111 111101 100001 100010 111100 | 15 | 101111 111001 000110 001101 001111 010011 111100 001010 |
| 8 | 111101 111000 101000 111010 110000 010011 101111 111011 | 16 | 110010 110011 110110 001011 000011 100001 011111 110101 |

1. **Langkah Kelima (Ekspansi blok biner)**

Ingatlah bahwa setelah permutasi awal, kami memiliki dua area teks biasa 32-bit yang disebut Plain Teks Kiri (LPT) dan Plain Teks Kanan (RPT). Selama permutasi ekspansi, RPT diperluas dari 32 bit menjadi 48 bit. Bit juga di permutasi sehingga disebut permutasi ekspansi. Ini terjadi karena RPT 32 bit dibagi menjadi 8 blok, dengan masing-masing blok terdiri dari 4 bit. Kemudian, setiap blok 4 bit dari langkah sebelumnya kemudian diperluas ke blok 6 bit yang sesuai, yaitu, per blok 4 bit, 2 bit lagi ditambahkan.



Tabel Ekspansi



Hasil nya

E(R(1)-1) = 100000 000000 000000 000000 000000 001101 010000 000110

K1          = 000110 110000 001011 101111 111111 000111 000001 110010  
---------------------------------------------------------------------------------------- XOR  
A1          = 100110 110000 001011 101111 111111 001010 010001 110100

Bisa kita lihat pada iterasi1 diatas setelah kita dapatkan hasil XOR antara E(R(1)-1) dengan K1 dan menghasilkan A1, maka proses berikutnya langsung masuk ke langkah berikutnya, dimana A1 akan dimasukan ke dalam S-Box dan menghasilkan output B1. B1 kemudian akan dipermutasikan lagi dengan tabel P-Box dan menghasilkan nilai PB1 yang kemudian di XOR-kan dengan L0 dan menghasilkan nilai **R1**. Nilai R1 ini digunakan untuk melanjutkan iterasi ke-2.

Iterasi ke 2

E(R(2)-1) = 011010 101110 100001 010110 100110 100101 010000 001101  
K2        = 011110 011010 111011 011001 110110 111100 100111 100101  
---------------------------------------------------------------------------------------- XOR  
A2           = 000100 110100 011010 001111 010000 011001 110111 101000

Iterasi – 3

E(R(3)-1) = 010001 010111 111011 110011 110001 010101 010010 100001

K3 = 010101 011111 110010 001010 010000 101100 111110 011001

---------------------------------------------------------------------------------------- XOR

A3 = 000100 001000 001001 111001 100001 111001 101100 111000

Iterasi – 4

E(R(4)-1) = 010111 110001 010111 110011 110101 011100 001111 110001

K4 = 011100 101010 110111 010110 110110 110011 010100 011101

---------------------------------------------------------------------------------------- XOR

A4 = 001011 011011 100000 100101 000011 101111 011011 101100

Iterasi – 5

E(R(5)-1) = 110110 101001 011100 000101 011001 011010 100110 100011

K5 = 011111 001110 110000 000111 111010 110101 001110 101000

---------------------------------------------------------------------------------------- XOR

A5 = 101001 100111 101100 000010 100011 101111 101000 001011

Iterasi – 6

E(R(6)-1) = 100101 011011 110001 010110 101110 101100 000111 111010

K6 = 011000 111010 010100 111110 010100 000111 101100 101111

---------------------------------------------------------------------------------------- XOR

A6 = 111101 100001 100101 101000 111010 101011 101011 010101

Iterasi – 7

E(R(7)-1) = 110010 100001 011111 110010 100111 111101 011001 010011

K7 = 111011 001000 010010 110111 111101 100001 100010 111100

---------------------------------------------------------------------------------------- XOR

A7 = 001001 101001 001101 000101 011010 011100 111011 101111

Iterasi – 8

E(R(8)-1) = 111100 001010 101001 010101 010011 110000 001010 100011

K8 = 111101 111000 101000 111010 110000 010011 101111 111011

---------------------------------------------------------------------------------------- XOR

A8 = 000001 110010 000001 101111 100011 100011 100101 011000

Iterasi – 9

E(R(9)-1) = 010010 101111 111000 000000 000010 101111 110101 010001

K9 = 111000 001101 101111 101011 111011 011110 011110 000001

---------------------------------------------------------------------------------------- XOR

A9 = 101010 100010 010111 101011 111001 110001 101011 010000

Iterasi – 10

E(R(10)-1)= 100111 111000 001110 100010 100111 110111 111000 001010

K10 = 101100 011111 001101 000111 101110 100100 011001 001111

---------------------------------------------------------------------------------------- XOR

A10 = 001011 100111 000011 100101 001001 010011 100001 000101

Iterasi – 11

E(R(11)-1)= 010011 110111 111010 101010 101111 110011 110001 011001

K11 = 001000 010101 111111 010011 110111 101101 001110 000110

---------------------------------------------------------------------------------------- XOR

A11 = 011011 100010 000101 111001 011000 011110 111111 011111

Iterasi – 12

E(R(12)-1)= 001001 011010 101001 011111 110001 010111 110010 101100

K12 = 011101 010111 000111 110101 100101 000110 011111 101001

---------------------------------------------------------------------------------------- XOR

A12 = 010100 001101 101110 101010 010100 010001 101101 000101

Iterasi – 13

E(R(13)-1)= 100110 100111 110111 111011 111110 101110 101100 001010

K13 = 100101 111100 010111 010001 111110 101011 101001 000001

---------------------------------------------------------------------------------------- XOR

A13 = 000011 011011 100000 101010 000000 000101 000101 001011

Iterasi – 14

E(R(14)-1)= 111001 010111 110000 001000 001000 001000 001011 111011

K14 = 010111 110100 001110 110111 111100 101110 011100 111010

---------------------------------------------------------------------------------------- XOR

A14 = 101110 100011 111110 111111 110100 100110 010111 000001

Iterasi – 15

E(R(15)-1)= 000110 101100 001100 000001 011001 011010 100101 010100

K15 = 101111 111001 000110 001101 001111 010011 111100 001010

---------------------------------------------------------------------------------------- XOR

A15 = 101001 010101 001010 001100 010110 001001 011001 011110

Iterasi – 16

E(R(16)-1)= 101101 011101 010100 000101 010101 010001 010110 100010

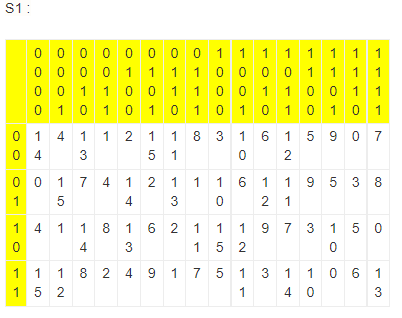
K16 = 110010 110011 110110 001011 000011 100001 011111 110101

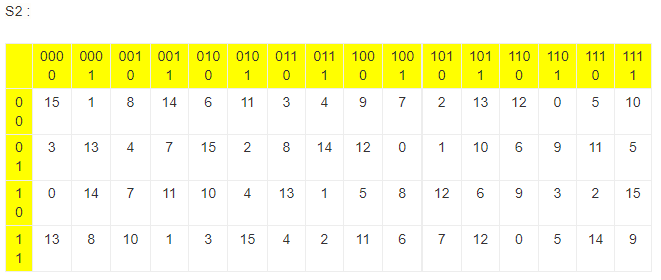
---------------------------------------------------------------------------------------- XOR

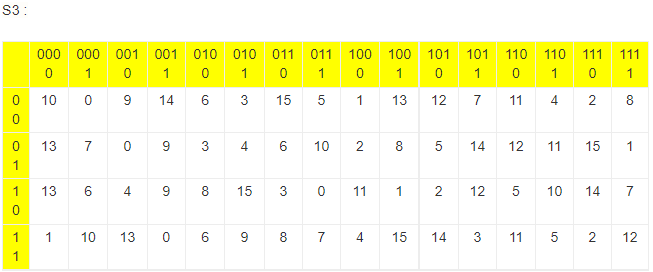
A16 = 011111 101110 100010 001110 010110 110000 001001 010111

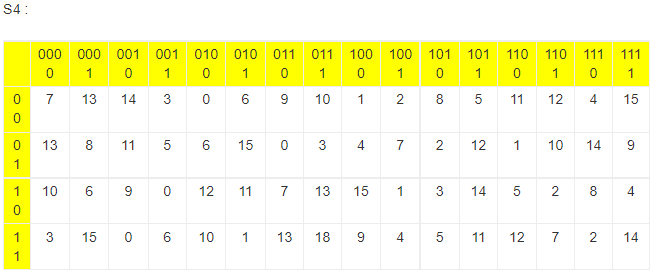
1. **Langkah Keenam Substitution BOX (S-BOX)**

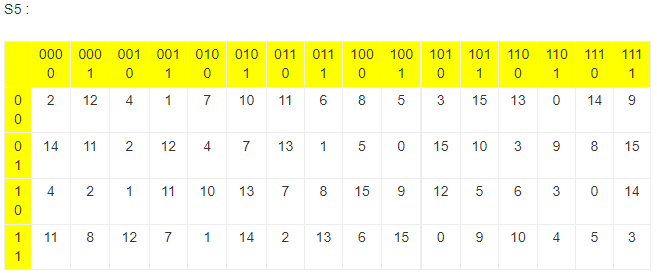
Tabel adalah tabel S-BOX

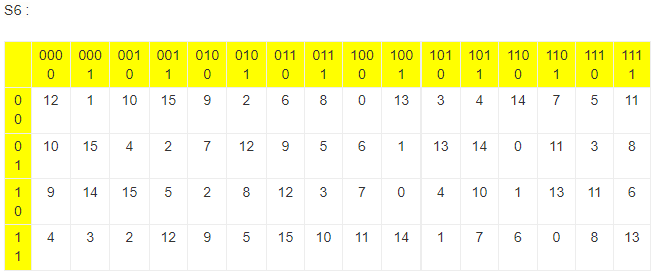


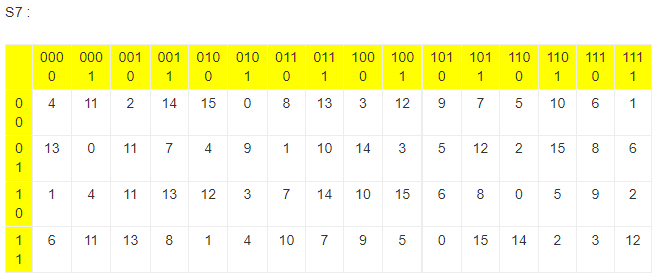


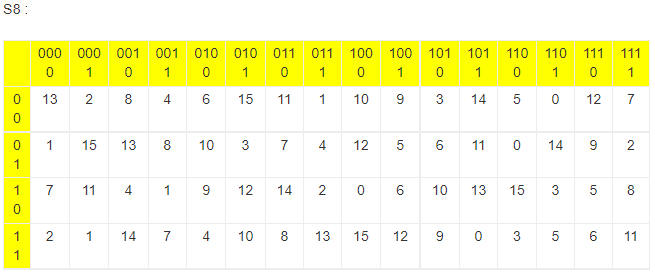




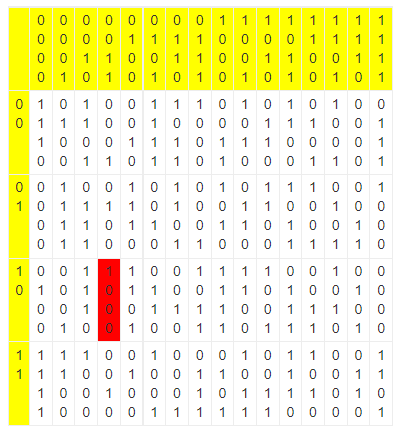








**Contoh Perhitungan S-BOX**



Kita ambil sampel blok bit pertama dari A1 yaitu 100110 kemudian Kita pisahkan blok menjadi 2 yaitu:

Bit pertama dan terakhir yaitu 1 dan 0 digabungkan menjadi 10

Bit kedua hingga ke lima 0011

Kemudian dibandingkan dengan memeriksa perpotongan antara keduanya didapatkan nilai 1000 (warna merah) dan seterusnya untuk blok kedua hingga blok kedelapan kita bandingkan dengan S2 hingga S8.

Maka hasilnya adalah :

|  |  |  |  |
| --- | --- | --- | --- |
| B1 | 1000 0101 0100 1000 0011 0010 1110 1010 | B9 | 0110 1110 1110 0001 1010 1011 0100 1010 |
| B2 | 1101 1100 0100 0011 1000 0000 1111 1001 | B10 | 0010 0001 0111 0000 0100 0001 0110 1101 |
| B3 | 1101 0110 0011 1100 1011 0110 0111 1111 | B11 | 0101 1110 0000 1100 1101 1011 1100 0010 |
| B4 | 0010 1001 1101 0000 1011 1010 1111 1110 | B12 | 0110 1000 0000 1011 0011 0110 1010 1101 |
| B5 | 0100 0001 0011 1101 1000 1010 1100 0011 | B13 | 1111 1001 1101 1011 0010 0100 1011 0011 |
| B6 | 0110 1101 1101 1100 0011 0101 0100 0110 | B14 | 1011 1000 0111 1110 1100 0101 1100 0001 |
| B7 | 1110 0011 0110 1011 0000 0101 0010 1101 | B15 | 0100 0001 0011 1001 1111 0111 0010 0111 |
| B8 | 0000 1000 1101 1000 1000 0011 1101 0101 | B16 | 1000 0001 0110 1010 1111 0111 0100 1011 |

1. **Langkah Ketujuh (Permutasi P-BOX dan XOR)**

Tabel P-BOX

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

Hasil Permutasi P-Box adalah

|  |  |  |  |
| --- | --- | --- | --- |
| P(Bi) | Hasil | P(Bi) | hasil |
| 1 | 00101000 10110011 01000100 11010001 | 9 | 11011001 00111011 10100011 10010100 |
| 2 | 10001011 11011001 10001100 00010011 | 10 | 00001100 00010101 01101110 00100100 |
| 3 | 01101111 10110010 10011100 11111110 | 11 | 01110001 00111110 10110000 01010011 |
| 4 | 00111111 00111011 01000111 10100001 | 12 | 10101000 01101000 10001110 11101001 |
| 5 | 10010101 00110010 11011000 01000101 | 13 | 10000110 11001011 11001111 11001011 |
| 6 | 00100100 00011011 11110011 11111000 | 14 | 00000101 11011101 00111010 01001111 |
| 7 | 11001000 11000001 11101110 01101100 | 15 | 10100101 00100110 11101100 11101100 |
| 8 | 00000111 00111001 00101001 01100001 | 16 | 00101001 11110111 01101000 11001100 |

Hasil P(Bi) kemudian di XOR kan dengan Li-1 untuk mendapatkan nilai Ri.

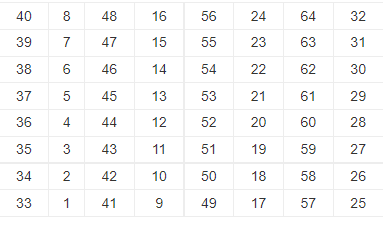
Sedangkan nilai Li sendiri diperoleh dari Nilai Ri-1 untuk nilai 1 <= i <= 16.

|  |  |
| --- | --- |
| P(B1) = 00101000 10110011 01000100 11010001  L(1)-1 = 11111111 10111000 01110110 01010111  ---------------------------------------------------------XOR  R1 = 11010111 00001011 00110010 10000110 | P(B9)      = 11011001 00111011 10100011 10010100 L(9)-1    = 11100101 01001010 10011000 01010001 -------------------------------------------------XOR R9    = 00111100 01110001 00111011 11000101 |
| P(B2) = 10001011 11011001 10001100 00010011  L(2)-1 = 00000000 00000000 00000110 10000011  ---------------------------------------------------XOR  R2 = 10001011 11011001 10001010 10010000 | P(B10)  = 00001100 00010101 01101110 00100100 L(10)-1    = 10010111 11000000 00010111 10101000 --------------------------------------------------XOR R10    = 10011011 11010101 01111001 10001100 |
| P(B3 = 01101111 10110010 10011100 11111110  L(3)-1 = 11010111 00001011 00110010 10000110  ---------------------------------------------------XOR  R3 = 10111000 10111001 10101110 01111000 | P(B11)  = 01110001 00111110 10110000 01010011 L(11)-1    = 00111100 01110001 00111011 11000101 -------------------------------------------------------XOR R11    = 01001101 01001111 10001011 10010110 |
| P(B4)      = 00111111 00111011 01000111 10100001 L(4)-1    = 10001011 11011001 10001010 10010000 ---------------------------------------------XOR R4    = 10110100 11100010 11001101 00110001 | P(B12)  = 10101000 01101000 10001110 11101001 L(12)-1    = 10011011 11010101 01111001 10001100 ----------------------------------XOR R12    = 00110011 10111101 11110111 01100101 |
| P(B5)      = 10010101 00110010 11011000 01000101 L(5)-1    = 10111000 10111001 10101110 01111000 ---------------------------------------XOR R5    = 00101101 10001011 01110110 00111101 | P(B13)  = 10000110 11001011 11001111 11001011 L(13)-1    = 01001101 01001111 10001011 10010110 ---------------------------------------------XOR R13    = 11001011 10000100 01000100 01011101 |
| P(B6)      = 00100100 00011011 11110011 11111000 L(6)-1    = 10110100 11100010 11001101 00110001 --------------------------------XOR R6    = 10010000 11111001 00111110 11001001 | P(B14)  = 00000101 11011101 00111010 01001111 L(14)-1    = 00110011 10111101 11110111 01100101 ------------------------------------------------XOR R14    = 00110110 01100000 11001101 00101010 |
| P(B7)      = 11001000 11000001 11101110 01101100 L(7)-1    = 00101101 10001011 01110110 00111101 --------------------------------------XOR R7    = 11100101 01001010 10011000 01010001 | P(B15)  = 10100101 00100110 11101100 11101100 L(15)-1    = 11001011 10000100 01000100 01011101 -----------------------------------------------XOR R15    = 01101110 10100010 10101000 10110001 |
| P(B8)      = 00000111 00111001 00101001 01100001 L(8)-1    = 10010000 11111001 00111110 11001001 --------------------------------------XOR R8    = 10010111 11000000 00010111 10101000 | P(B16)= 00101001 11110111 01101000 11001100  L(16)-1= 00110110 01100000 11001101 00101010  --------------------------------------------------XOR  R16 = 00011111 10010111 10100101 11100110 |

1. **Langkah Kedelapan**

Langkah terakhir adalah menggabungkan R16 dengan L16 kemudian dipermutasikan untuk terakhir kali dengan tabel Invers Initial Permutasi(IP-1).

Tabel IP-1



R16 L16 = 00011111 10010111 10100101 11100110 01101110 10100010 10101000 10110001

Menghasilkan Output:

Cipher(dalam biner) = 01010110 11110001 11010101 11001000 01010010 10101111 10000001 00111111

Maka hasil akhir dalam hexa adalah **56 f1 d5 c8 52 af 81 3f**

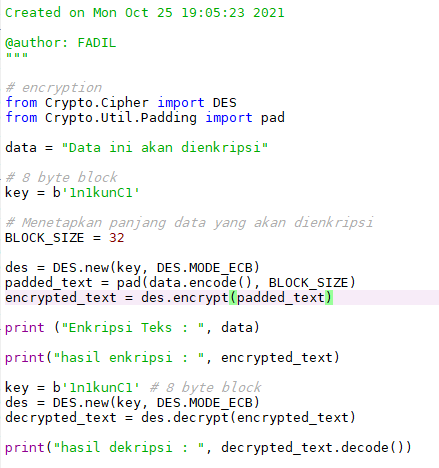
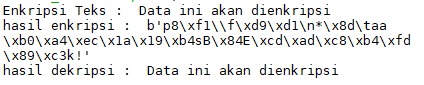
1. **Keamanan**

Algoritma DES mengundang banyak kontroversi dari para ahli di seluruh dunia. Salah satu kontroversi tersebut adalah S-Box yang digunakan pada DES. S-Box merupakan bagian vital dari DES karena merupakan bagian yang paling sulit dipecahkan. Hal ini disebabkan karena S-Box merupakan satu – satunya bagian dari DES yang komputasinya tidak linear. Sementara itu, rancangan dari S-Box sendiri tidak diberitahukan kepada publik. Karena itulah, banyak yang curiga bahwa S-Box dirancang sedemikian rupa sehingga memberikan trapdoor kepada NSA agar NSA bisa membongkar semua ciphertext yang dienkripsi dengan DES kapan saja.



Kontroversi yang kedua adalah jumlah bit pada kunci DES yang dianggap terlalu kecil, hanya 56 bit. Akibatnya DES rawan terhadap serangan brute force. Walaupun terdapat kerawanan tersebut, DES tetap digunakan pada banyak aplikasi seperti pada enkripsi PIN (Personal Identification Numbers) pada mesin ATM (Automatic Teller Machine) dan transaksi perbankan lewat internet. Bahkan, organisasi – organisasi pemerintahan di Amerika seperti Department of Energy, Justice Department, dan Federal Reserve System menggunakan DES untuk melindungi penyebaran data mereka.

**D. Contoh Program Python**

**E. TUGAS Kelompok**

1. Membuat program enkripsi dan dekripsi menggunakan algoritma DES. Boleh menggunakan library python/php atau membuat dari awal.

2. Tambahkan kondisi sehingga program dapat berjalan ketika kunci lebih dari 8 bit dan teks lebih dari atau kurang dari 8 bit.

3. Tugas dikerjakan secara kelompok, tetapi dikumpulkan per individu di akun github masing-masing

4. Yang dikumpukan :

Kode program dan Screenshot ­­­­

5. Pengumpulan Tugas Praktikum

* **Untuk kelas TI E** paling lambat tanggal 8 November 2021 Jam 23.59

**Untuk kelas TI D** paling lambat tanggal 9 November 2021 Jam 23.59

**Contoh** program bisa dilihat di https://github.com/fadilrahman46/DES