Faculty of Computer Science and Engineering

Ho Chi Minh City University of Technology



Nguyen Phuong Vuong - 1614186

# 

Cryptography and Network Security

Lab 2

# Exercise 1.

A transformation is singular if it is not invertible or undoable. This means if you apply that, you will lose information. A nonsingular transform means you can undo the transform.

# Exercise 2.

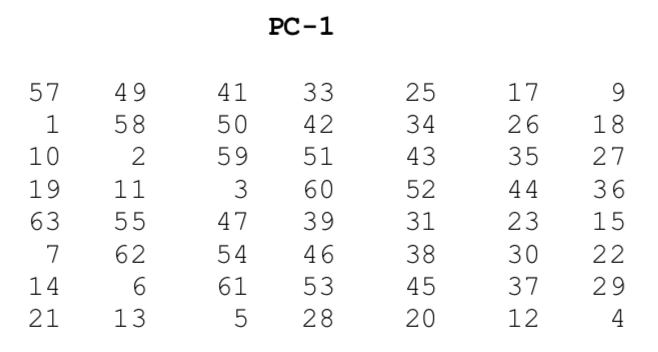
A block cipher is one in which a block of plain-text is treated as a whole and used to produce cipher-text block of equal length.

A stream cipher is one that encrypts a digital data stream one bit or one byte at a time.

# Exercise 4.

Part a. Derive K1, the first round key.

K = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111



Follow the table PC-1, we have:

**K+ = 1111 0000 1100 1100 1010 1010 0000 1010 1010 1100 1100 1111 0000 0000**

Next, split this key into left and right halves, C0 and D0, where each half has 28 bits.

**C0 = 1111 0000 1100 1100 1010 1010 0000**

**D0 = 1010 1010 1100 1100 1111 0000 0000**

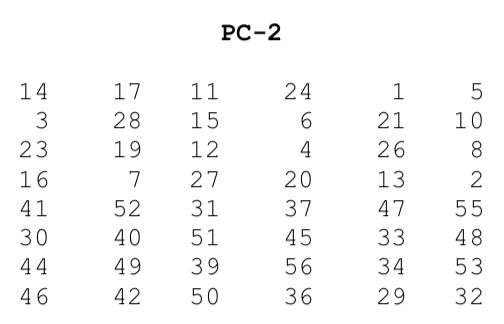
To derive K1, we will calculate C1, D1 by moving each bit one place to the left of C0 and D0.

**C1 = 1110 0001 1001 1001 0101 0100 0001**

**D1 = 0101 0101 1001 1001 1110 0000 0001**

We have **C1D1 = 1110 0001 1001 1001 0101 0100 0001 0101 0101 1001 1001 1110 0000 0001**

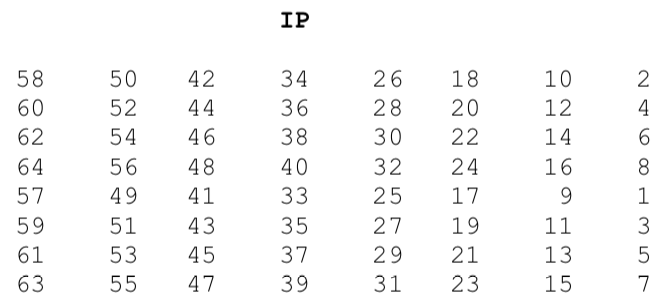
Follow the table PC-2, we have:



After we apply the permutation PC-2, becomes

**K1 = 0000 1011 0000 0010 0110 0111 1001 1011 0100 1001 1010 0101**

b,  
**M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111**

****

Follow the table IP, we have:

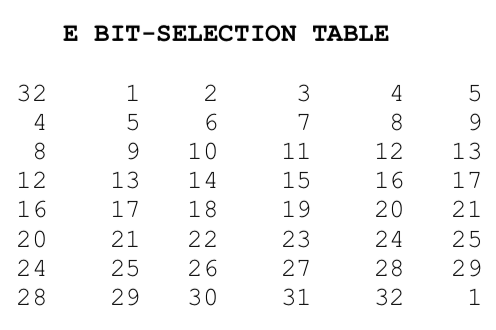
IP = 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000 1010 1010 1111 0000 1010 1010

Next divide the permuted block IP into left half L0 of 32 bits and a right half R0 of 32 bits.

L0 = 1100 1100 0000 0000 1100 1100 1111 1111

R0 = 1111 0000 1010 1010 1111 0000 1010 1010

c,



The E table expands R0 to 48 bits:

E(R0) = 01110 100001 010101 010101 011110 100001 010101 010101

d,  
A = 011100 010001 011100 110010 111000 010101 110011 110000

e,

Follow tables S1, S2, S3, S4, S5, S6, S7, S8, we have:  
B1 = 111100, S1[0,14]= 0 or 0000

B2 = 010001, S2[1,8]= 12 or 1100

B3 = 011100, S3[0,14]= 2 or 0010

B4 = 110010, S4[2,9]= 1 or 0001

B5 = 111000, S5[2,12]= 6 or 0110

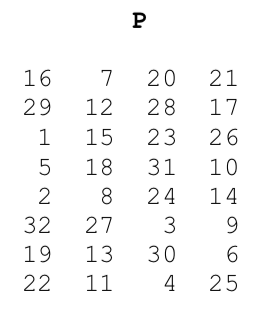
B6 = 010101, S6[1,10]= 13 or 1101

B7 = 110011, S7[3,9]= 5 or 0101

B8 = 110000, S8[2,8]= 0 or 0000

f, Follow the results of exercise e, we get B with 32 bits:  
B = 0000 1100 0010 0001 0110 1101 0101 0000

g,



Follow table P, we have:

**P(B) = 1001 0010 0001 1100 0010 0000 1001 1100**

h,

R1 = P (B) ⊕ L0

**R1 = 0101 1110 0001 1100 1110 1100 0110 0011**

i,

**R0 = 1111 0000 1010 1010 1111 0000 1010 1010**

**R1 = 0101 1110 0001 1100 1110 1100 0110 0011**

L1 = R0. The cipher-text which is the concatenation of L1 and R1, is

**F** [**0 A A**](https://www.facebook.com/messages/t/flukekado.fiction) **F** [**0 A A**](https://www.facebook.com/messages/t/flukekado.fiction) **5 E 1 C E C** [**6 3**](https://www.facebook.com/messages/t/flukekado.fiction)

# Exercise 5.

Decrypt the string (10100010) using the key (0111111101)

- First, we generate 2 keys:

We use table P10 to permutation key:

Table P10

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Output | 3 | 5 | 2 | 7 | 4 | 10 | 1 | 9 | 8 | 6 |

We derive K = 11111 10011

We will calculate K1 by moving each bit one place to the left of 2 section of 11111-10011 and using table P8:

LS-1 = 11111 00111

Table P8

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Output | 6 | 3 | 7 | 4 | 8 | 5 | 10 | 9 |  |  |

We use table P8 to deriving permutation key:

**K1 = 0101 1111**

We will calculate K2 by moving each bit two place to the left of 2 section of 11111-00111 and using table P8:

LS-2 = 11111 11100

We use table P8 to deriving permutation key:

**K2 = 1111 1100**

- Secondly, we decrypt the string S = 10100010

Table IP

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Output | 2 | 6 | 3 | 1 | 4 | 8 | 5 | 7 |

IP(S) = 0011 0001

Split IP(S) into 2 section such as the left section (0011) and the right section (0001)

Table E/P

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input | 1 | 2 | 3 | 4 |  |  |  |  |
| Output | 4 | 1 | 2 | 3 | 2 | 3 | 4 | 1 |

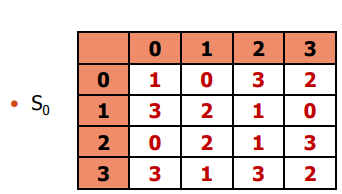
We use table E/P to deriving the permutated right section(0001)

E/P(0001) = 1000 0010

Xor(K2 and E/P(0001)) = 0111 1110

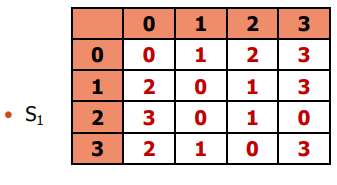
Split Xor(K2 and E/P(0001)) into 2 section such as the left section(0111) and the right section(1110)

We derive the left section(0111) to use for table S0:



We have row= 01 and column = 11 🡺 S0(left) = 00

We derive the right section(1110) to use for table S1:



We have row = 10 and column = 11 🡪 S1(right) = 00

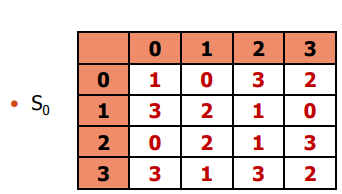
We use S1S2(0000) to use for table P4(2341) we derive P4(S1S2) = 0000

Xor(the left section of IP and P4(S1S2)) = 0011

We put together the right section of IP and the result of XOR above: 0001 0011 (1)

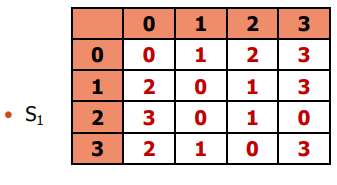
Split 0001 0011 into 2 section left (0001) and section right (0011)

We derive the left section(0011) to use for table S0:



We have row= 01 and column = 01 🡺 S0(left) = 10

We derive the right section(0001) to use for table S1:



We have row = 01 and column = 00 🡪 S1(right) = 10

We use S1S2(0100) to use for table P4(2341) we derive P4(S1S2) = 1010

Xor(the left section of (1) and P4(S1S2)) = 1011

We put together the right section of (1) and the result above: 1011 0011

We use it for table IP-1:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Output | 4 | 1 | 3 | 5 | 7 | 2 | 8 | 6 |

**We derive the plain-text: 1110 1010**