BÀI THỰC HÀNH SỐ 1

Môn: MẬT MÃ & AN NINH MẠNG

-o0o-

**Họ tên: Lương Hữu Phú Lợi**

**MSSV: 1911545**

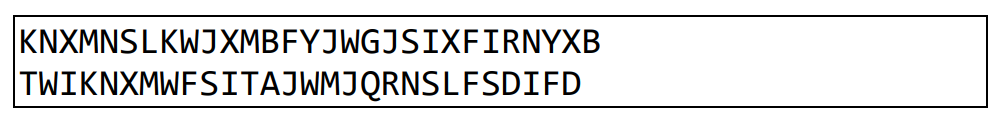
**Nhóm: L02**

**Phần 1. Các hệ mã đối xứng truyền thống**

**Câu 1:**

…………………………….…………………………….…………………………….

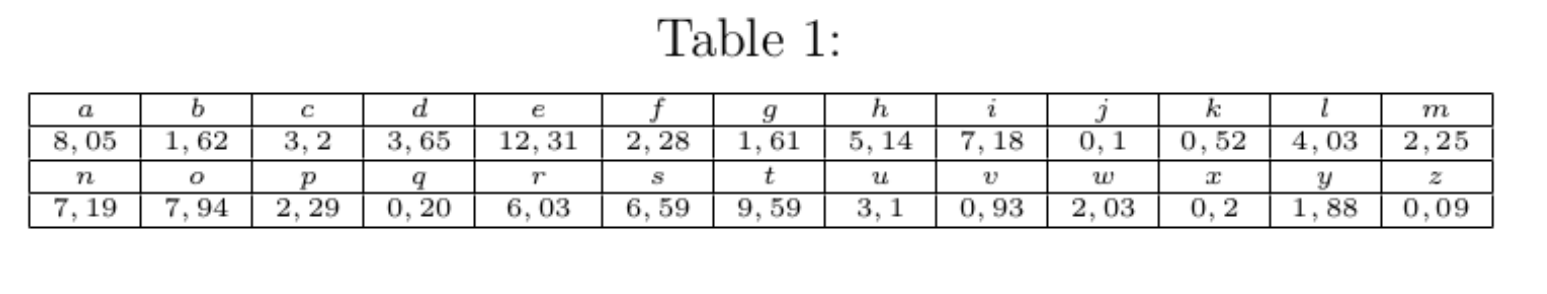
Đầu tiên, ta đếm số lần xuất hiện của các ký tự trong bảng



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Kí tự | F | N | X | W | J | I | S | M | K | B | Y | L | R | D | T | A | S | G |
| Tần suất | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 |

=> các ký tự xuất hiện nhiều nhất là: F,N, X, W, J, I ,S

Trong Table 1, tần suất xuất hiện của ký tự  **e** là nhiều nhất



Ta xét khoảng cách từ E đến các ký tự F,N, X, W, J, I ,S rồi từ ciphertext truy ngược lại thành plain text. Ta có các trường hợp sau:

- E -> F => k = 1 => JMWKMR…. => không có nghĩa

- E -> N => k = 9 => BEODE…. => không có nghĩa

- E -> X => k = 19 => RUETU…. => không có nghĩa

- E -> W => k = 18 => SVFU…. => không có nghĩa

- E -> J => k = 5 => FISHING…. => xuất hiện từ có nghĩa = > dịch cả câu => FISH FRESHWATER BENDS ADMIT SWORDFISH RANK OVERWHELMING => có nghĩa

- E -> I => k = 4 => GJTIJO…. => không có nghĩa

- E -> S => k = 14 => WZJY…. => không có nghĩa

Vậy kết quả plain text là FISH FRESHWATER BENDS ADMIT SWORDFISH RANK OVERWHELMING

**Điểm yếu:**

- Cách mã hóa đợn giản, chỉ bảo mật ở mức tối thiểu

- Tần suất các chữ cái là manh mối giải mã

**Câu 2:**

…………………………….…………………………….…………………………….

Ta sử dụng vòng lặp để thay thế M thành các ký tự trong bảng chữ cái.

Ta có psudocode như sau:

For k=0; k <= 26; k++

For each c in ciphertext

Append (c+26-k)mod26 to result

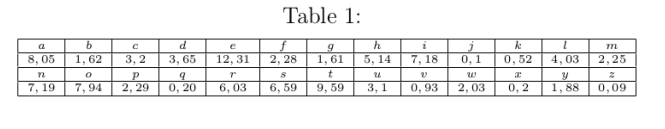
Print result + k

Sau khi thay thế các giá trị k thì ta có thể tìm được kết quả của plain text

Với đề bài thì giá trị k = 4 sẽ là kết quả. Ta được plain text WORLDCUP

**Câu 3:**

…………………………….…………………………….…………………………….



Dựa theo bảng trên thì tần xuất nhiều nhất và nhì là E và T

Mà trong ciphertext thì có B và U

Ta có :

E -> B ~ 4 -> 1

T -> U ~ 19 -> 20

(4a+b) mod 26 = 1

(19a+b) mod 26 = 20

=> 15a mod 26 = 19

=> a = 3, b= 15

Vậy công thức là C = (3a+15) mod 26

**Câu 4:**

…………………………….…………………………….…………………………….

**Vấn đề 1:**

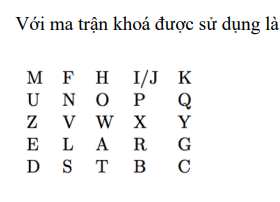
Nhược điểm chính của mã hóa với bảng điều khiển một lần là nó yêu cầu một vùng đệm có cùng độ dài với thư được mã hóa.Vì mỗi pad chỉ có thể được sử dụng một lần, điều này có nghĩa là cần phải chia sẻ pad có cùng độ dài với tin nhắn được chia sẻ. Key của OTP là ngẫu nhiên nên rất khó để thống nhất giữa người gửi và người nhận và rất dễ xảy ra sai sót vì phải dùng từ ngữ để diễn tả nơi chứa key.

**Vấn đề 2:**

Nếu mất đi bảng mã quy ước chung thì người nhận sẽ mãi không thể giải mã được

**Câu 5:**

…………………………….…………………………….…………………………….



Nội dung: Must see you over Cadogan West. Coming at once.

Mã hoá: UZTB DLG ZPN NWLG TGTUERO VLDB DUHFPE RH WQSRZ

**Câu 6:**

…………………………….…………………………….…………………………….

Họ và tên : Lương Hữu Phú Lợi = > key1 = LUONG, key2 = LOI

Plaintext: spyarrivesonthursday

Lần 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **L** | **U** | **O** | **N** | **G** |
| **11** | **20** | **14** | **13** | **6** |
| S | P | Y | A | R |
| R | I | V | E | S |
| O | N | T | H | U |
| R | S | D | A | y |

**=> RSUYSRORAEHAYVTDPINS**

Lần 2

|  |  |  |
| --- | --- | --- |
| **L** | **O** | **I** |
| **11** | **14** | **8** |
| R | S | U |
| Y | S | R |
| O | R | A |
| E | H | A |
| Y | V | T |
| D | P | I |
| N | S |  |

**=> URAATIRYOEYDNSSRHVPS**

**Câu 7:**

…………………………….…………………………….…………………………….

**Phần 2. Chuẩn mã hoá dữ liệu DES**

**Câu 1:**

Sự khác nhau giữa mã hóa khối và mã hóa dòng

|  |  |
| --- | --- |
| **Mã hóa khối** | **Mã hóa dòng** |
| Mã hóa các khối dữ liệu có kích thước lớn cố định, chiều dài khối lớn hơn hoặc bằng 64bits. | Mã hóa từng bit/từng byte |
| Được dùng rộng rãi | Được dùng trong các ứng dụng thời gian thực |
| Sử dụng chế độ ECB, CBC | Sử dụng chế độ CFB, OFB, Counter |
| Dùng cả kĩ thuật khuếch tán và nhầm lẫn | Chỉ dùng kĩ thuật nhầm lẫn |

**Câu 2:**

**a.** Tính khoá con K1 được sử dụng cho vòng mã hoá đầu tiên

Thông điệp ở dạng HEX:

*0 1 2 3 4 5 6 7 8 9 A B C D E F*

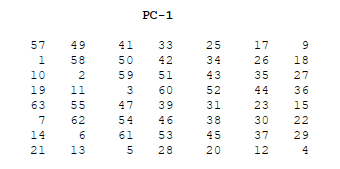
Khóa dưới dạng Hex

*0 1 2 3 4 5 6 7 8 9 A B 2 7 5 0*

Trước nhất, ta phân tích **key** từ mã hex sang binary như sau:

Keyb = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 0010 0111 0101 0000

Sử dụng bảng PC\_1, ta tìm K+ (56 bit)



K+ = 0011 0000 1000 1100 0110 1010 1000 0110 1010 0100 1100 0011 0000 0000

Tách K+ ra thành 2 nửa C0, D0 ta được:

C0 = 0011000010001100011010101000

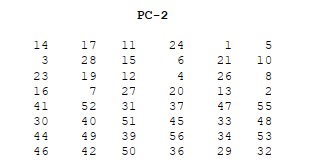
D0 = 0110101001001100001100000000

Để tạo Key1, ta shift left (dịch trái) C0, D0 1 bit, được C1, D1:

C1 = 0110000100011000110101010000

D1 = 1101010010011000011000000000

Sau đó, ta kết hợp C1, D1 lại và đưa qua bảng PC\_2 để tạo ra key1 (48 bit):



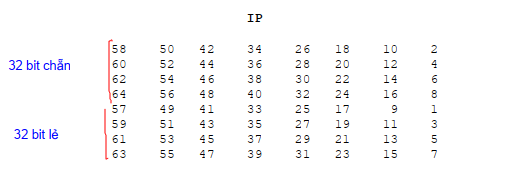
Key1 = 0101 0010 0000 0010 0100 0111 1001 1011 0000 0000 1010 0011

**b.** Tính L0, R0

Trước nhất, từ message, ta chuyển sang chuỗi nhị phân như sau:

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

Từ Messageb, ta thực hiện initial permutation được kết quả:



IP(messageb) = 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000 1010 1010 1111 0000 1010 1010

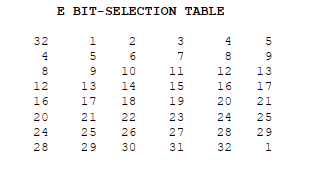
Tách message ra 2 nửa, ta được L0 và R0:

L0 = 1100 1100 0000 0000 1100 1100 1111 1111

R0 = 1111 0000 1010 1010 1111 0000 1010 1010

**c.** Tính kết quả mở rộng R0: E[R0], với E là hàm mở rộng

Áp dụng hàm mở rộng E cho R0, ta được kết quả (48 bits)



E[R0] = 0111 1010 0001 0101 0101 0101 0111 1010 0001 0101 0101 0101

**d.** Tính giá trị A = E[R0] ⊕ K1

Lấy E[R0] xor với Key 1, ta được kết quả (48 bits):

E[R0] = 0111 1010 0001 0101 0101 0101 0111 1010 0001 0101 0101 0101

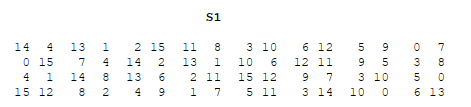
Key1 = 0101 0010 0000 0010 0100 0111 1001 1011 0000 0000 1010 0011

A = 0010 1000 0001 0111 0001 0010 1110 0001 0001 0101 1111 0110

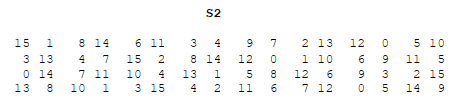
**e.** Chia 48-bit kết quả ở câu d và chia thành các nhóm 6 bit, thực hiện tính toán trên từng nhóm 6 bit thông qua S-box, ghi lại kết quả.

Ta chia kết quả câu A thành các nhóm 6 bit như sau:

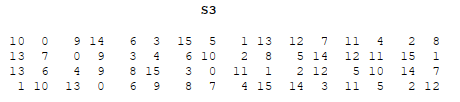
A = 001010 000001 011100 010010 111000 010001 010111 110110  
S1(B1) = Sbox1[0][5] = 15



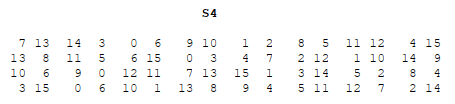
S2(B2) = Sbox2[1][0] = 3



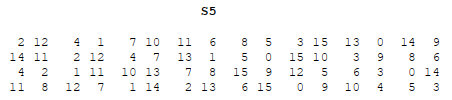
S3(B3) = Sbox3[0][14] = 2



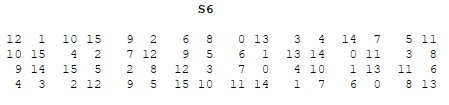
S4(B4) = Sbox4[0][9] = 2



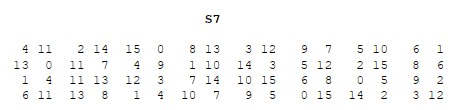
S5(B5) = Sbox5[2][12] = 6



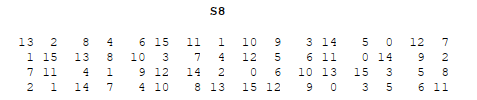
S6(B6) = Sbox6[1][8] = 6



S7(B7) = Sbox7[1][11] = 12



S8(B8) = Sbox8[2][11] = 13



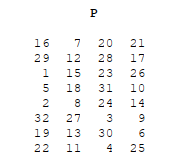
**f.** Nối các kết quả tính được ở câu e thành chuỗi kết quả 32-bit, ghi lại kết quả dưới dạng binary (B).

Áp dụng các S box lên từng nhóm 6 bit, ta được chuỗi kết quả (32 bits):

S-box = 1111 0011 0010 0010 0110 0110 1100 1101

**g.** Tính giá trị P(B), với P là hàm hoán vị

Áp dụng hàm hoán vị P lên chuỗi 32bit vừa tính được qua các Sbox, ta được kết quả:



P(B) = 0100 1000 1111 0100 1100 1010 1010 1111

**h.** Tính giá trị R1 = P (B) ⊕ L0

P(B) = 0100 1000 1111 0100 1100 1010 1010 1111  
L0 = 1100 1100 0000 0000 1100 1100 1111 1111

R1 = 1000 0100 1111 0100 0000 0110 0101 0000

**i.** Ghi lại kết quả ciphertext cho vòng thứ nhất

Ciphertext cho vòng thứ nhất được tạo ra bằng cách ghép L1 = R0 và R1 vừa tính được

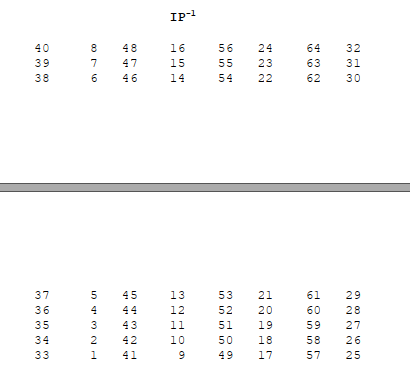
L1 = 1111 0000 1010 1010 1111 0000 1010 1010

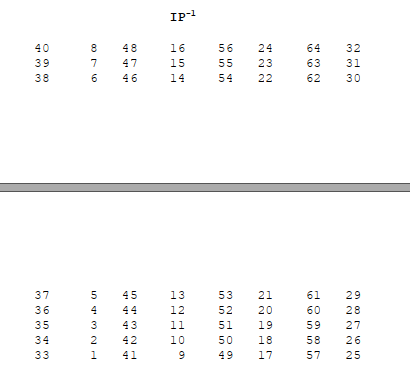
R1 = 1000 0100 1111 0100 0000 0110 0101 0000

Ta đảo ngược R1 và L1 được:

R1L1 = 1000 0100 1111 0100 0000 0110 0101 0000 1111 0000 1010 1010 1111 0000 1010 1010

Sau đó, áp dụng bảng final\_permutation, được kết quả sau cùng như sau:





Code cho việc mã hóa DES có thể được tìm thấy ở đây:

https://github.com/PhuLoi-1911545/Lab-encryption-and-network-security