BÀI THỰC HÀNH SỐ 1

Môn: MẬT MÃ & AN NINH MẠNG

-o0o-

**Họ tên: Phạm Duy Quang**

**MSSV: 2011899**

**Nhóm: L01**

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

Câu 1. Cho một bản mã (ciphertext) được tạo ra bằng cách dùng hệ mã Caesar để mã hoá một văn bản viết bằng tiếng Anh, hãy giải mã ciphertext trên mà không cần biết thông tin khoá và giải thích cách làm, từ đó cho biết điểm yếu của giải thuật Caesar là gì?

KNXMNSLKWJXMBFYJWGJSIXFIRNYXB

TWIKNXMWFSITAJWMJQRNSLFSDIFD

Trả lời: Tần suất xuất hiện của các ký tự trong bản mã trên:

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

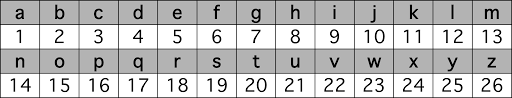
Ta nhận thấy các ký tự xuất hiện nhiều nhất trong dãy trên là: 'J', 'X', 'S', 'I', 'F', 'W', 'N'.

Tần suất xuất hiện của các chữ cái trong các văn bản viết bằng tiếng Anh:

Table

Description automatically generated

Ta sẽ so sánh tần suất xuất hiện của bản mã nói trên với tần suất xuất hiện trung bình của các chữ cái trong tiếng Anh để xác định các ký tự có thể là các ký tự thường xuất hiện trong ngôn ngữ. Với thông tin tần suất xuất hiện của các ký tự khác, ta có thể đoán rằng bản mã này có thể được mã hóa bằng phương pháp Caesar với khoá là một số xác định sao cho khi dịch các ký tự trong bản mã sang trái hoặc phải một số lượng ký tự tương ứng, chúng sẽ trở thành các ký tự phổ biến hơn trong tiếng Anh, chẳng hạn như kí tự “E” thường xuất hiện nhiều nhất, sau đó là kí tự “T”.



Với cách mã hóa Caesar, số lượng ký tự dịch sẽ là khoá của mã hóa. Vì vậy, ta có thể thử các giá trị khác nhau của khoá để giải mã bản mã. Trường hợp kí tự “E” được mã hóa thành các kí tự còn lại trong bảng tần suất từ cao tới thấp và ở trường hợp kí tự “J” với khóa là 5, ta thu được kết quả giải mã là:

**FISHINGFRESHWATERBENDSADMITSWORDFISHRANDOVERHELMINGANYDAY**

Câu 2: Cho ciphertext: asvphgyt đã được mã hoá bằng hệ mã thay thế theo công thức:

|  |
| --- |
| C = (M + K) mod 26 |

Trong đó C là bản mã (ciphertext), M là thông điệp (plaintext), K là khoá. Hãy tìm lại khoá K và giải mã thông điệp?

**Trả lời:** Chương trình brute-force:

import string

def decrypt(cipher\_text, key):

plain\_text = ""

for c in cipher\_text:

if c in string.ascii\_letters:

temp = ord(c) + key

if temp > ord('z'):

temp = temp - 26

plain\_text = plain\_text + chr(temp)

else:

plain\_text = plain\_text

return plain\_text

CIPHER\_TEXT = "asvphgyt"

for key in range(1, 26):

print(f"Key = {key} ====> ", decrypt(CIPHER\_TEXT, key))

Kết quả:

Key = 1 ====> btwqihzu

Key = 2 ====> cuxrjiav

Key = 3 ====> dvyskjbw

Key = 4 ====> ewztlkcx

Key = 5 ====> fxaumldy

Key = 6 ====> gybvnmez

Key = 7 ====> hzcwonfa

Key = 8 ====> iadxpogb

Key = 9 ====> jbeyqphc

Key = 10 ====> kcfzrqid

Key = 11 ====> ldgasrje

Key = 12 ====> mehbtskf

Key = 13 ====> nficutlg

Key = 14 ====> ogjdvumh

Key = 15 ====> phkewvni

Key = 16 ====> qilfxwoj

Key = 17 ====> rjmgyxpk

Key = 18 ====> sknhzyql

Key = 19 ====> tloiazrm

Key = 20 ====> umpjbasn

Key = 21 ====> vnqkcbto

**Key = 22 ====> worldcup**

Key = 23 ====> xpsmedvq

Key = 24 ====> yqtnfewr

Key = 25 ====> zruogfxs

Thử tất cả 26 trường hợp với khóa k chạy từ 1 đến 26, ta thu được bản giải mã ứng với k = 4 là **worldcup** khi ta dịch trái mỗi kí tự trong ciphertext sang trái 4 kí tự.

Câu 3: Cho một bản mã (ciphertext) được tạo ra bởi mật mã Affine với công thức như sau (bản rõ là một văn bản tiếng Anh):

|  |
| --- |
| C = E([a, b], p) = (ap+b) mod 26 |

Người ta nhận thấy rằng trong bản mã này, ký tự B xuất hiện nhiều nhất, ký tự U xuất hiện nhiều thứ hai, hãy tìm ra công thức đúng của hệ mã Affine nói trên (tìm giá trị a,b)?

Trả lời: Công thức mã hóa: C = (ap + b) mod 26

Không mất tính tổng quát, giả sử a, b < 26.

Kí tự B (2) nhiều nhất Thay E (5) B(2)

Kí tự U (20) nhiều thứ hai Thay T(20) U(21)

Từ (1) và (2)

hay

Ta có:

Với: k = 1 a = 3,

k = 16 a = 29.

Chọn a = 3.

Ta có:

Với: ,

.

Chọn b = 13.

Công thức hệ mã: .

Câu 4: Hãy nêu ra hai vấn đề đối với Mật mã One-time Pad?

**Trả lời:** Hai vấn đề đối với Mật mã One-time Pad:

**a) Pad phải thực sự ngẫu nhiên và có độ dài bằng plaintext**

Việc tạo ra dữ liệu một cách thực sự ngẫu nhiên cần lượng dữ liệu lớn. Phần lớn các trường hợp máy tính sử dụng các giải thuật để tạo dữ liệu có vể là ngẫu nhiên, nhưng thực ra các dữ liệu đó phụ thuộc vào dữ liệu mồi (seed). Dữ liệu mồi có thể được người dùng thiết lập trước hoặc lấy từ những nguồn khó đoán như thời gian hệ thống hiện đại. Một số giải thuật sinh dữ liệu giả ngẫu nhiên như CryptGenRandom, Yarrow, Fortuna,… được gọi là cryptographically-secure, tức là dữ liệu sinh ra hầu như không khác gì dữ liệu thực sự ngẫu nhiên và có thể dùng cho mục đích bảo mật. Nhưng chúng ta không thể dùng giải thuật có tính chất cryptographically-secure để sinh pad cho OTP vì mã hóa này chỉ đủ mức độ bảo mật để sử dụng trong thực tế chứ chưa đạt đến mức độ hoàn hảo. Phương pháp sinh dữ liệu thực sự ngẫu nhiên bằng cách đo một nguồn dữ liệu ngẫu nhiên như áp suất khí quyển, nhiệt độ,… rồi chuyển về dạng số tuy nhiên phương pháp này chậm và phức tạp. Mỗi khi cần sinh dữ liệu ta cần đợi để thu thập đủ lượng dữ liệu và phải có phương pháp loại trừ những sai số mang tính hệ thống trong quá trình đo đạc.

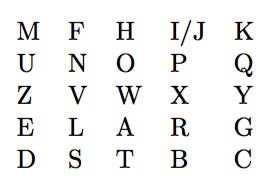
**b) Phía gửi và nhận đều phải biết nội dung của pad và giữ kín không để lộ cho bên thứ ba**

Nếu như ta có cách để chuyển 1 pad dài bằng plaintext cho người nhận mà không sợ bị lộ thì sao ta không dùng cách đó để chuyển luôn plaintext mà phải dùng OTP? Đây là nhược điểm lớn nhất và làm hạn chế đáng kể tính ứng dụng của OTP. Vậy nếu áp dụng các giải thuật trao đổi khóa (key exchange) vào OTP được không? Câu trả lời là được, những giải thuật mã hóa dùng public key như RSA đã xử lí vấn đề này tương đối ổn bằng các giải thuật như Diffie-Hellman hay ECDH. Tuy nhiên khi áp dụng vào OTP sẽ có 2 hạn chế là giải thuật trao đổi khóa được thiết kế cho các lượng dữ liệu nhỏ, nó không hiệu quả khi áp dụng với dữ liệu lớn như pad và thứ hai là không giải thuật trao đổi khóa nào là hoàn hảo, nếu dùng chúng thì OTP cũng không còn hoàn hảo nữa.

Câu 5: Hãy sử dụng mật mã Playfair để mã hoá thông điệp bên dưới:

|  |
| --- |
| Must see you over Cadogan West. Coming at once. |

Với ma trận khoá được sử dụng là:



**Trả lời:** Ma trận khóa:

**Text

Description automatically generated**

Plaintext: Must see you over Cadogan West. Coming at once

Viết lại thành cặp 2 kí tự: Mu St Se Ey Ou Ov Er Ca Do Ga nW es tC om in ga to nc ez (Do số lượng từ là lẻ nên thêm chữ “Z” vào từ cuối cùng)

Mã hóa:

Mu -> UZ

St -> TB

Se -> DL

Ey -> GZ

Ou -> PN

Ov -> NW

Er -> LG

Ca -> TG

Do -> TU

Ga -> ER

nW -> OV

es -> LD

tC -> BD

om -> UH

in -> PF

ga -> ER

to -> HW

nc -> QS

ez -> DE

Ciphertext:

**UZ TB DL GZ PN NW LG TG TU ER OV LD BD UH PF ER HW Qs DE**

Câu 6: Sử dụng mật mã Double Transposition (Tham khảo: <https://www.pbs.org/wgbh/nova/decoding/doubtrans.html>), để mã hoá thông điệp bên dưới, mô tả từng bước thực hiện?

|  |
| --- |
| spyarrivesonthursday |

*Yêu cầu:* sử dụng tên đệm để làm key1, và tên làm key2 (Ví dụ: sinh viên có tên Nguyễn Văn An thì key1=VAN, key2=AN).

Trả lời: Phương pháp mã hóa Double Transposition sử dụng hoán vị cột hai lần đối với bản plaintext ban đầu với một khóa cho cả hai bước hoặc 2 khóa khác nhau, ở đây chúng ta sử dụng 2 khóa lần lượt là DUY và QUANG.

Key1 = DUY

Key2 = QUANG

Hoán vị cột hoạt động như sau: Đầu tiên lấy khóa thứ nhất là DUY và viết bản plaintext dưới khóa bằng các hàng có độ dài bằng độ dài khóa như ở dưới. Sau đó ta mã hóa bằng cách viết nội dung các cột thành bản mã theo thứ tự xuất hiện trong alphabet của kí tự trong khóa trên cột đó.

Plaintext: spyarrivesonthursday

Key 1: DUY

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| **D** | **U** | **Y** |
| s | p | y |
| a | r | r |
| i | v | e |
| s | o | n |
| t | h | u |
| r | s | d |
| a | y |  |

saistra prvohsy yrenud

Ta thu được sau khi mã hóa lần thứ nhất là saistraprvohsyyrenud.

Key 2: QUANG

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | 5 | 1 | 3 | 2 |
| **Q** | **U** | **A** | **N** | **G** |
| s | a | i | s | t |
| r | a | p | r | v |
| o | h | s | y | y |
| r | e | n | u | d |

ipsn tvyd sryu sror aahe

Mã hóa lần 2 tương tự, ta thu được cipher text là **ipsntvydsryusroraahe**.

Câu 7: SOLVE A CIPHER game. Truy cập vào trang web American Cryptogram Association bên dưới:

<http://www.cryptogram.org/resource-area/solve-a-cipher/>

Chọn “New Puzzle” và nhấn nút Go!, hãy sử dụng gợi ý và các công cụ trên trang web để tìm lại bản rõ (plaintext) từ bản mã (ciphertext) được tạo ra, ghi lại kết quả và giải thích cách làm, chụp ảnh màn hình kết quả?

**Trả lời:** Cipher text:

|  |
| --- |
| OWOIQGDW WDS GQVX: JAIOBQIX IXHI DZ XCWJGQCVX. XKXGN NXQG HOUXGOQC TJHYOXH GJC DKXG DGOSOCQA SDAW GJHT IGQOA ZGDB QCVTDGQSX ID CDBX, QAQHYQ. |

Graphical user interface, text, application

Description automatically generated

Thứ tự về độ phổ biến tần suất của các từ (giảm dần): E, T, A, O, N, I …

Phân tích tần suất xuất hiện, ta thấy G, Q, X xuất hiện nhiều nhất.

Thử lần lượt, kiểm tra thấy X tương ứng với E.

Graphical user interface, text, application

Description automatically generated

Theo tần suất xuất hiện, tiếp tục thử thay G bằng T, A, O, R thì thấy G tương ứng với R.Graphical user interface, text, application

Description automatically generated

Tiếp tục thử thay Q bằng T, A, O thì thấy Q tương ứng với A.

Graphical user interface, text, application

Description automatically generated

Thử với D, ta thấy D tương ứng với O.

Graphical user interface, text, application

Description automatically generated

Tiếp tục thử với O, ta thấy O tương ứng với I.

Graphical user interface, text, application

Description automatically generated

Ta tiếp tục đoán trên một số cụm từ như “o-er” “over”.

Thử K tương ứng V ta có kết quả chính xác.

Graphical user interface, text, application

Description automatically generated

Ta đoán cụm “ever- -ear” “every year”.

Thử N tương ứng Y ta có kết quả chính xác.

A picture containing text

Description automatically generated

Ta đoán cụm “ra-e” “race”.

Thử V tương ứng C ta có kết quả chính xác.

Graphical user interface, text, application

Description automatically generated

Tiếp tục thử tần suất với C, ta thấy C tương ứng với N.

Text, application

Description automatically generated

Tiếp tục thử và sai, giải tương tự, ta giải được đáp án:

Text

Description automatically generated

Plaintext: iditarod dog race: ultimate test of endurance. every year siberian huskies run over original gold rush trail from anchorage to nome, alaska.

**Phần 2. Chuẩn mã hoá dữ liệu DES**

**Câu 1:** Mô tả sự khác nhau giữa mã hoá khối và mã hoá dòng?

**Trả lời:** Sự khác nhau giữa mã hoá khối và mã hoá dòng:

|  |  |
| --- | --- |
| **Mã hóa khối (Block Cipher)** | **Mã hóa dòng (Stream Cipher)** |
| Chuyển từ plaintext thành ciphertext theo từng block của plaintext | Chuyển từ plaintext thành ciphertext theo từng byte của plaintext |
| Mã hoá khối sử dụng 64 bit hoặc nhiều hơn 64 bit | Mã hoá dòng sử dụng 8 bit |
| Đơn giản hơn | Phức tạp hơn |
| Trong mã hoá khối, ta khó mã hoá ngược văn bản | Trong mã hoá dòng, văn bản được mã hóa ngược rất dễ dàng |
| Sử dụng cả confusion + diffusion | Chỉ sử dụng confusion |
| Các thuật toán được sử dụng trong mã hoá khối là ECB và CBC | Các thuật toán được sử dụng trong mã hoá dòng là CFB và OFB |
| Mã hoá khối hoạt động dựa trên các kỹ thuật chuyển vị như kỹ thuật hàng rào đường sắt, kỹ thuật chuyển vị cột,… | Mã hoá dòng hoạt động trên các kỹ thuật thay thế như giải thuật Caesar, mật mã thay thế polygram,… |
| Chậm hơn | Nhanh hơn |

Câu 2: Sử dụng giải thuật mã hoá DES để mã hoá thông điệp theo thông tin và trả lời các câu hỏi như bên dưới (chỉ thực hiện mã hoá 1 vòng):

Thông điệp được cho dưới dạng Hex:

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

Khoá được cho dưới dạng Hex (với 4 ký tự cuối (X) là 4 số cuối của mã số sinh viên):

|  |
| --- |
| 0 1 2 3 4 5 6 7 8 9 A B X X X X |

Hãy trả lời các câu hỏi sau:

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

b. Tính L0, R0

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

d. Tính giá trị A = E[R0] ⊕ K1

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.

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

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

h. Tính giá trị R1 = P (B) ⊕ L0

i. Ghi lại kết quả ciphertext cho vòng thứ nhất

**Trả lời: \* Message (hex): 0 1 2 3 4 5 6 7 8 9 A B C D E F**

M (64 bit) = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111

**\* Key (hex): 0 1 2 3 4 5 6 7 8 9 A B 1 8 9 9** (Mã số sinh viên: 2011899)

K (64 bit) = 00000001 00100011 01000101 01100111 10001001 10101011 00011000 10011001

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

K = 00000001 00100011 01000101 01100111 10001001 10101011 00011000 10011001

Thực hiện 56-bit permutation với PC-1:

Table

Description automatically generated

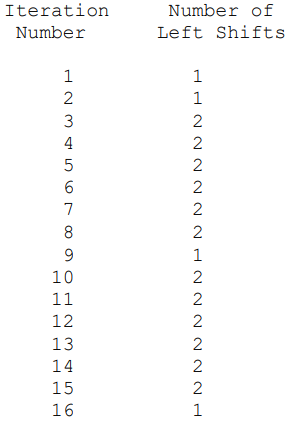
K+ = 1011000 0000011 0000101 0101100 0010101 0000011 0011110 0000000

Chia K+ thành nửa trái (28-bit) và nửa phải (28-bit):

C0 = 1011000 0000011 0000101 0101100

D0 = 0010101 0000011 0011110 0000000

Sau đó, ta dịch C0 sang trái 1 ký tự để thu được C1, tương tự với D1:

****

C1 = 0110000 0000110 0001010 1011001

D1 = 0101010 0000110 0111100 0000000

Kết hợp lại:

C1D1 = 0110000 0000110 0001010 1011001 0101010 0000110 0111100 0000000

Thực hiện 56-bit permutation với PC-2:

Table

Description automatically generated

**K1 = 000100 110000 001000 000111 100010 110100 100010 100001**

b. Tính L0, R0

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

Thực hiện initial permutation với M:

Table

Description automatically generated

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

Chia IP thành nửa trái L0 (32 bits) và nửa phải R0 (32 bits)

**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

L1 = R0**=** 1111 0000 1010 1010 1111 0000 1010 1010

R1 = L0 + f (Ro, K1)

Đầu tiên chúng ta phải mở rộng R0 từ 32 bits lên 48 bits bằng cách áp dụng hàm mở rộng E và thu được kết quả:

Table

Description automatically generated with medium confidence

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

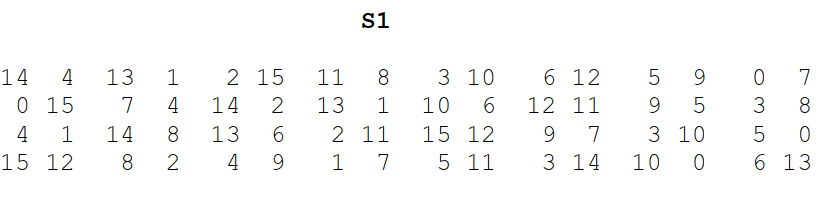
d. Tính giá trị A = E[]

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.

A =

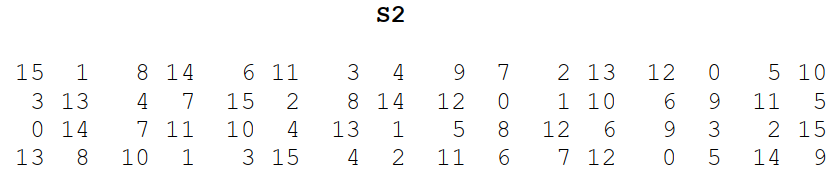
= B1 B2 B3 B4 B5 B6  B7 B8

Tính toán S1(B1), S2(B2), S3(B3), S4(B4), S5(B5), S6(B6), S7(B7), S8(B8), với Si(Bi) là output thứ i của S-box.



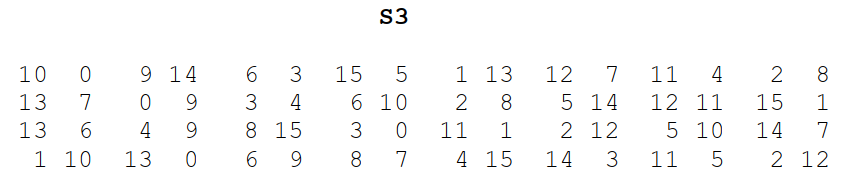
B1 = (00 hàng 0, 1101 cột 13)

S1() = 9dec = 1001bin



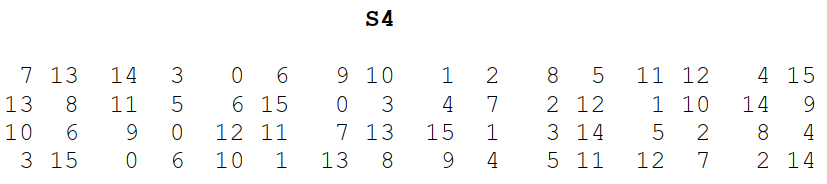
B2 = (01 hàng 1, 1000 cột 8)

S2() = 12dec = 1100bin



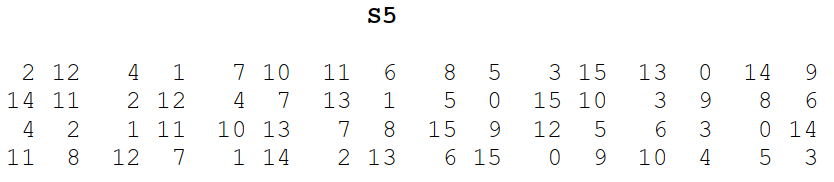
B3 = (01 hàng 1, 1110 cột 14)

S3() = 15dec = 1111bin



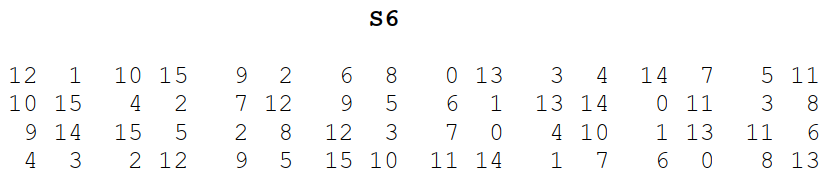
B4 = (00 hàng 0, 1001 cột 9)

S4() = 2dec = 0010bin



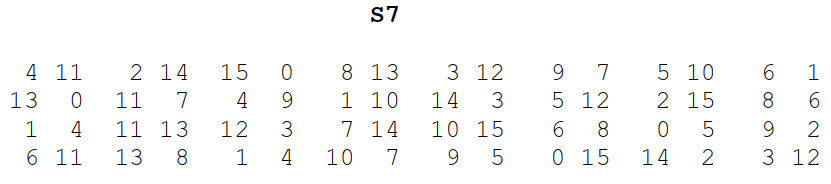
B5 = (10 hàng 2, 1110 cột 14)

S5() = 0dec = 0000bin



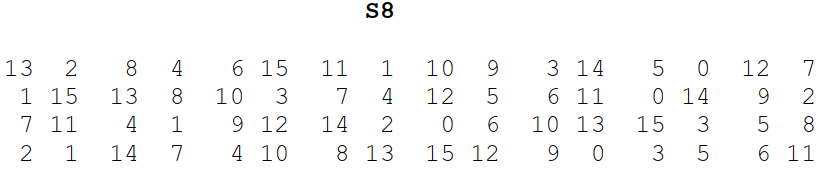
B6 = (01 hàng 1, 1010 cột 10)

S6() = 13dec = 1101bin



B7 = (11 hàng 3, 1011 cột 11)

S7() = 15dec = 1111bin



B8 = (10 hàng 2, 1010 cột 10)

S8() = 10dec = 1010bin

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

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

B = 1001(4) 1100(8) 1111(12) 0010(16) 0000(20) 1101(24) 1111(28) 1010(32)

A picture containing text

Description automatically generated

**f = P(B) = 0001 1110 1101 1011 0010 0101 0001 1111**

h. Tính giá trị

= 0001 1110 1101 1011 0010 0101 0001 1111

= 1100 1100 0000 0000 1100 1100 1111 1111

**= 1101 0010 1101 1011 1110 1001 1110 0000**

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

L1 = R0 = 1111 0000 1010 1010 1111 0000 1010 1010

R1 = 1101 0010 1101 1011 1110 1001 1110 0000

Kết quả ciphertext cho vòng thứ nhất:

**C1 = L1R1**

**= 1111 0000 1010 1010 1111 0000 1010 1010 1101 0010 1101 1011 1110 1001 1110 0000 (bin)**

**= F0AA F0AA D2DB E9E0 (hex)**