TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN ĐIỆN TỬ - VIỄN THÔNG

BÀI TẬP LỚN LÝ THUYẾT MẬT MÃ

Đề tài: Tìm hiểu mật mã Ceasar và mật mã nhân

Giảng viên hướng dẫn**:** TS. Hồ Mạnh Linh

Nhóm đề tài:

1. Hoàng Ngọc Anh - 20160075	6. Nguyễn Thị Minh Châu - 20167074
2. Nguyễn Ngọc Anh - 20160141	7. Nguyễn Văn Chính - 20160464
3. Nguyễn Thế Anh – 20160154 (Nhóm trưởng)	8. Tô Quang Chính – 20160467
4. Trần Đăng Bách - 20160313	9. Nguyễn Hữu Chúc - 20160453
5. Lê Ngọc Bảo - 20160326	10. Lê Văn Chung - 20160443

Hà Nội, 2019

Mục lục

1. Mật mã học cổ điển

1.1. Lịch sử

Mật mã học là nghành có lịch sử từ hàng nghìn năm nay. Trong phần lớn thời gian phát triển của mình (ngoại trừ vài thập kỷ trở lại đây), lịch sử của mật mã học chính là lịch sử của các phương pháp mật mã học cổ điển – các phương pháp mật mã hóa với bút và giấy, đôi khi có sự hỗ trợ từ những dụng cụ cơ khí đơn giản. Nhưng cho đến nay đã trở nên lạc hậu do các phương thức mã hóa này quá đơn giản và những kẻ tấn công có thể dễ dàng bẻ khóa thông qua nhiều phương thức như: tấn công vét cạn (sử dụng máy tính để thử hết mọi trường hợp), tấn công thông kê (dựa trên tần suất xuất hiện của các chữ cái).

1.2. Mật mã học là gì?

Theo Wikipedia, mật mã học là một lĩnh vực liên quan đến các kỹ thuật ngôn ngữ và toán học để đảm bảo an toàn thông tin, cụ thể là trong thông tin liên lạc.

1.3. Các thành phần của hệ thống mật mã

- Bản rõ (Plaintext)
- Bản mã (Ciphertext)
- Khóa mã hóa (Key)
- Mã hóa (Encrypt)
- Giải mã (Decrypt)

2. Mật mã Ceasar

2.1. Khái niệm

Mật mã Ceasar hay còn gọi là mật mã dịch chuyển là một trong những mật mã đơn giản được biết đến nhiều nhất. Mật mã thuộc hệ mật thay thế đơn ký tự (mono alphabetic), trong đó mỗi ký tự trong văn bản được thay thế bằng ký tự cách nó một đoạn trong bảng chữ cái để tạo thành bản mã.

Hệ mật Ceasar được lấy tên theo vị hoàng đế Ceasar, người đã sử dụng nó thường xuyên trong công việc để đảm bảo thông tin.

2.2. Ví dụ

Phép dịch chuyển có thể biểu diễn bằng hai bảng chữ cái (tiếng anh và tiếng việt).

Đối với bảng chữ cái tiếng anh, nếu độ dịch là 5, A sẽ được thay bằng F, B sẽ được thay bằng G…, W sẽ được thay bằng B.

Plain text	**AB**CDEFGHIJKLMNOPQRSTUV**W**XYZ
Cipher text	**FG**HIJKLMNOPQRSTUVWXYZA**B**CDE

Đối với bảng chữ cái tiếng việt, nếu độ dịch là 5, A sẽ được thay bằng D, B thay bằng Ê..., Ư thay bằng Ă.

Plain text	**A**ĂÂ**B**CDĐEÊGHIKLMNOÔƠPQRSTU**Ư**VXY
Cipher text	**D**ĐE**Ê**GHIKLMNOÔƠPQRSTUƯVXYA**Ă**ÂBC

2.3. Mã hóa

Bản rõ P mã hóa ta thu được bản mã C theo công thức:

C = (P + *k*) mod N, với $\begin{cases} C \text{ là bản mã}, P \text{ là bản rõ} \\ k \text{ là số thứ tự dịch} \\ N \text{ là số ký tự trong bảng chữ cái } (26) \end{cases}$

Xét ví dụ: Cho bản rõ: P = "**LYTHUYETMATMA**", khóa *k* = 5 với bảng chữ cái tiếng anh. Tìm bản mã?

Bản rõ	L	Y	T	H	U	Y	E	T	M	A	T	M	A
i	11	24	19	7	20	24	4	19	12	0	19	12	0
(i + *k*) mod 26	16	3	24	12	25	3	9	24	17	5	24	17	5
Bản mã	Q	D	Y	M	Z	D	J	Y	R	F	Y	R	F

Kết quả thu được sau mã hóa: C = "**QDYMZDJYRFYRF**".

2.4. Giải mã

Bản mã C tiến hành giải mã ta thu được bản rõ P theo công thức:

P = (C - *k*) mod N, với $\begin{cases} C \text{ là bản mã}, P \text{ là bản rõ} \\ k \text{ là số thứ tự dịch} \\ N \text{ là số ký tự trong bảng chữ cái } (26) \end{cases}$

Xét ví dụ: Cho bản mã: C = "**QDYMZDJYRFYRF**", khóa *k* = 5 với bảng chữ cái tiếng anh. Tìm bản rõ?

Bản rõ	Q	D	Y	M	Z	D	J	Y	R	F	Y	R	F
i	16	3	24	12	25	3	9	24	17	5	24	17	5

(i + *k*) mod 26	11	24	19	7	20	24	4	19	12	0	19	12	0
Bản mã	L	Y	T	H	U	Y	E	T	M	A	T	M	A

Kết quả thu được sau giải mã: P = "**LYTHUYETMATMA**"

2.5. Thám mã

Cũng như các hệ mật thay thế đơn ký tự khác mật mã Ceasar dễ dàng bị phá vỡ và không đáp ứng được yêu cầu an toàn thông tin trong truyền thông. Trong thực tế người ta thường kết hợp với một mật mã phức tạp hơn (mật mã Vignere) và được ứng dụng trong ROT13 (Rotate by 13 places).

Dấu hiệu để nhận thấy một bản mã sử dụng mật mã Ceasar thường là nó gồm những ký tự rất lộn xộn trông rất khó nhìn. Có thể thấy trong tiếng việt chữ 'E' thường xuất hiện rất nhiều, dựa vào yếu tố này ta có thể đếm số ký tự xuất hiện nhiều nhất trong bản mã để suy ra khóa *k* của hệ mật.

Cách phát hiện trên chỉ là cái phát hiện đơn giản để ta giải mã. Trên thực tế để giải mã hệ mật này ta sẽ giải mã với từng khóa *k* (vì mật mã Ceasar có không gian mã hóa rất hẹp, *k* thường từ 0 đến 26).

2.6. Bài tập vận dụng

Bài 1. Viết chương trình mã hóa chuỗi ký tự phân biệt hoa thường (N = 26) trong bảng mã **ASCII** sử dụng mật mã Ceasar (không mã hóa khoảng trống).

Source Code

```
def encrypt(plainText, k):
    cipherText = ""
    for i in range(len(plainText)):
        charIndex = ord(plainText[i])
        if charIndex == 32:
            cipherText += " "
        elif charIndex >= 65 and charIndex <= 90:
            cipherText += chr((charIndex - 65 + k) % 26 + 65)
        else:
            cipherText += chr((charIndex - 97 + k) % 26 + 97)
    return cipherText

def decrypt(cipherText, k):
    plainText = ""
    for i in range(len(cipherText)):
        charIndex = ord(cipherText[i])
        if charIndex == 32:
            plainText += " "
        elif charIndex >= 65 and charIndex <= 90:
            plainText += chr((charIndex - 65 - k) % 26 + 65)
```

```
        else:
            plainText += chr((charIndex - 97 - k) % 26 + 97)
    return plainText

def Ceasar_encode():
    print('Nhập bản rõ: ')
    plainText = str(input())
    print('Nhập khóa k = ')
    k = int(input())
    while k < 0:
        print('Nhập lại khóa k = ')
        k = int(input())
    cipherText = encrypt(plainText, k)
    print('Bản mã: %s' % cipherText)

def Ceasar_decode():
    print('Nhập bản mã: ')
    cipherText = str(input())
    print('Nhập khóa k = ')
    k = int(input())
    while k < 0:
        print('Nhập lại khóa k = ')
        k = int(input())
    plainText = decrypt(cipherText, k)
    print('Bản rõ: %s' % plainText)


Ceasar_encode()
Ceasar_decode()
```

Bài 2. Viết chương trình mã hóa chuỗi ký tự bất kỳ trong bảng mã **ASCII** sử dụng mật mã Ceasar (không mã hóa khoảng trống).

Source Code

```
def encrypt(plainText, k):
    cipherText = ""
    for i in range(len(plainText)):
        if plainText[i] == " ":
            cipherText += plainText[i]
        else:
            cipherText += chr((ord(plainText[i]) - 33 + k) % 94 + 33)
    return cipherText

def decrypt(cipherText, k):
    plainText = ""
    for i in range(len(cipherText)):
        if cipherText[i] == " ":
            plainText += cipherText[i]
        else:
            plainText += chr((ord(cipherText[i]) - 33 - k) % 94 + 33)
    return plainText
```

```
def Ceasar_encode():
    print('Nhập bản rõ: ')
    plainText = str(input())
    print('Nhập khóa k = ')
    k = int(input())
    while k < 0:
        print('Nhập lại khóa k = ')
        k = int(input())
    cipherText = encrypt(plainText, k)
    print('Bản mã: %s' % cipherText)

def Ceasar_decode():
    print('Nhập bản mã: ')
    cipherText = str(input())
    print('Nhập khóa k = ')
    k = int(input())
    while k < 0:
        print('Nhập lại khóa k = ')
        k = int(input())
    plainText = decrypt(cipherText, k)
    print('Bản rõ: %s' % plainText)


Ceasar_encode()
Ceasar_decode()
```

3. Mật mã nhân (Multicative Ciphers)

3.1. Khái niệm

Mật mã nhân cũng tương tự với mật mã Ceasar và thuộc hệ mật thay thế đơn ký tự (mono alphabetic). Trong đó mỗi ký tự trong văn bản được thay thế bằng ký tự cách nó một khoảng theo cấp số nhân để tạo thành bản mã.

3.2. Ví dụ

Phép dịch chuyển có thể biểu diễn bằng hai bảng chữ cái (tiếng anh và tiếng việt).

Đối với bảng chữ cái tiếng anh, nếu độ dịch là 5, A sẽ được thay bằng A, B sẽ được thay bằng F…, W sẽ được thay bằng G.

Plain text	**AB**CDEFGHIJKLMNOPQRSTUV**W**XYZ
Cipher text	**AF**KPUZEJOTYDINSXCHMRWB**G**LQV

Đối với bảng chữ cái tiếng việt, nếu độ dịch là 5, A sẽ được thay bằng A, B thay bằng N..., Ư thay bằng G.

Plain text	**A**ĂÂ**B**CDĐEÊGHIKLMNOÔƠPQRSTU**Ư**VXY

Cipher text	ADHNQƯĂĐIORVÂEKÔSXBÊLƠTYCGMPU

3.3. Mã hóa

Bản rõ P mã hóa ta thu được bản mã C theo công thức:

$$\boldsymbol{C = (P * k) \bmod N}, \textit{ với } \begin{cases} C \text{ là bản mã}, P \text{ là bản rõ} \\ k \text{ là hệ số mã khóa} \\ N \text{ là số ký tự trong bảng chữ cái } (26) \end{cases}$$

Điều kiện để tồn tại k^{-1} (để có thể giải mã được): Gcd (k, N) = 1.

Xét ví dụ: Cho bản rõ: P = "**LYTHUYETMATMA**", khóa k = 5 với bảng chữ cái tiếng anh. Tìm bản mã?

Bản rõ	L	Y	T	H	U	Y	E	T	M	A	T	M	A
i	11	24	19	7	20	24	4	19	12	0	19	12	0
(i * *k*) mod 26	3	16	17	9	22	16	20	17	8	0	17	8	0
Bản mã	D	Q	R	J	W	Q	U	R	I	A	R	I	A

Kết quả thu được sau mã hóa: C = "**DQRJWQURIARIA**".

3.4. Giải mã

Bản rõ P mã hóa ta thu được bản mã C theo công thức:

$$\boldsymbol{C = (P * k^{-1}) \bmod N}, \textit{ với } \begin{cases} C \text{ là bản mã}, P \text{ là bản rõ} \\ k^{-1} \text{ là nghịch đảo module } N \text{ của } k \\ N \text{ là số ký tự trong bảng chữ cái } (26) \end{cases}$$

Không phải lúc nào ta cũng có có thể tìm được nghịch đảo module N của k, nhưng theo định lý Euclid: "Tồn tại nghịch đảo module N của k nếu: gcd (k, N) =1".

Cơ sở toán học:

Định lý Euclid mở rộng: ax + by = c Trong đó a, b, c là các hệ số nguyên, x, y là các ẩn nhận giá trị nguyên. Điều kiện cần và đủ để phương trình này có nghiệm (nguyên) là UCLN (a, b) là ước của c.	Gọi k^{-1} là nghịch đảo module N của k thì: k * $k^{-1} \equiv 1$ (mod N) Đặt a = N, b = k, x = s, y = t, ta có: sN + tk = UCLN (N, k) Lấy mod cả hai vế với N: ⇨ t * $k \equiv 1$ mod N ⇨ tồn tại k^{-1} là nghịch đảo module N của k

Theo đó với N = 26: k = {1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25} thì luôn tồn tại nghịch đảo mudule 26 của k.

Xét ví dụ: Cho bản mã: C = "**DQRJWQURIARIA**", khóa k = 5 với bảng chữ cái tiếng anh. Tìm bản rõ?

Với k = 5 ta tiến hành tìm nghịch đảo module N = 26 của k áp dụng định lý Euclid mở rộng như trên:

q	**N**	**k**	**r**	**t1**	**t2**	**t**
5	26	5	1	0	1	-5
5	5	1	0	1	-5	26
	1	0		**-5**	26	

⇨ k^{-1} = -5 mod 26 = 21

Tiến hành giải mã:

Bản rõ	D	Q	R	J	W	Q	U	R	I	A	R	I	A
i	3	16	17	9	22	16	20	17	8	0	17	8	0
(i * k^{-1}) mod 26	11	24	19	7	20	24	4	19	12	0	19	12	0
Bản mã	L	Y	T	H	U	Y	E	T	M	A	T	M	A

Kết quả thu được sau giải mã: P = "**LYTHUYETMATMA**".

Như vậy kết quả giải mã được trùng khớp với kết quả ở phần mã hóa.

3.5. Thám mã

Tương tự mật mã Ceasar thì để thám mã mật mã nhân ta tiến hành giải ngược bản mã với từng giá trị của k, như đã trình bày thì không phải lúc nào cũng có thể giải được mật mã nhân nó phụ thuộc vào ý đồ của người mã hóa.

Ứng dụng phổ biến của mật mã nhân không phải là dùng để giải mã mà nó được dùng chỉ để mã hóa, nghe thật phi lý, sự thật nó được dùng để mã hóa password tài khoản của người dùng trên cơ sở dữ liệu. Khi Hacker tấn công được vào cơ sở dữ liệu của một trang web họ không thể giải mã được tài khoản người dùng tránh đánh cắp thông tin.

3.6. Bài tập vận dụng

Bài 1. Viết chương trình mã hóa và giải mã chuỗi ký tự phân biệt hoa, thường (N = 26) trong bảng mã **ASCII** sử dụng mật mã nhân (lưu ý: không mã hóa khoảng trống).

Source Code

```
def module_inverse(k, N = 26):
    t1, t2 = 0, 1
    while k > 0:
        q = N // k
        r = N - k*q
        N, k = k, r
        t = t1 - t2*q
        t1, t2 = t2, t
    if N == 1:
        return t1
    return 0

def encrypt(plainText, k):
    cipherText = ""
    for i in range(len(plainText)):
        charIndex = ord(plainText[i])
        if charIndex == 32:
            cipherText += " "
        elif charIndex >= 65 and charIndex <= 90:
            cipherText += chr(((charIndex - 65) * k) % 26 + 65)
        else:
            cipherText += chr(((charIndex - 97) * k) % 26 + 97)
    return cipherText

def decrypt(cipherText, k):
    k = module_inverse(k)
    if k != 0:
        plainText = ""
        for i in range(len(cipherText)):
            charIndex = ord(cipherText[i])
            if charIndex == 32:
```

```
                plainText += " "
            elif charIndex >= 65 and charIndex <= 90:
                plainText += chr(((charIndex - 65) * k) % 26 + 65)
            else:
                plainText += chr(((charIndex - 97) * k) % 26 + 97)
        return plainText
    return 0

def multiplicativeCipher_encode():
    print('Nhập bản rõ: ')
    plainText = str(input())
    print("Nhập k: ")
    k = int(input())
    while k < 0:
        print("Nhập lại k: ")
        k = int(input())
    cipherText = encrypt(plainText, k)
    print('Bản mã: %s' % cipherText)

def multiplicativeCipher_decode():
    print('Nhập bản mã: ')
    cipherText = str(input())
    print("Nhập k: ")
    k = int(input())
    while k < 0:
        print("Nhập lại k: ")
        k = int(input())
    plainText = decrypt(cipherText, k)
    if plainText == 0:
        print('Không thể giải mã !')
    else:
        print('Bản mã: %s' % plainText)

# Main
multiplicativeCipher_encode()
multiplicativeCipher_decode()
```

Bài 2. Viết chương trình mã hóa và giải mã chuỗi ký tự bất kỳ trong bảng mã **ASCII** sử dụng mật mã nhân (lưu ý: không mã hóa khoảng trống).

Source Code

```
# Hàm tìm ước chung lớn nhất
def gcd(a, b):
    while b > 0:
        q = a // b
        r = a - b*q
        a, b = b, r
    return a

# Hàm tìm nghịch đảo của k (k')
def module_inverse(k, n = 94):
```

```
    t1, t2 = 0, 1
    while k > 0:
        q = n // k
        r = n - k*q
        n, k = k, r
        t = t1 - t2*q
        t1, t2 = t2, t
    if n == 1:
        return t1
    return 0

# Hàm mã hóa mật mã nhân
def encrypt(plainText, k):
    cipherText = ""
    for i in range(len(plainText)):
        if plainText[i] == " ":
            cipherText += plainText[i]
        else:
            cipherText += chr((((ord(plainText[i]) - 33) * k ) % 94) + 33 )
    return cipherText

# Hàm giải mã mật mã nhân
def decrypt(cipherText, k):
    k = module_inverse(k, 94)
    if k != 0:
        plainText = ""
        for i in range(len(cipherText)):
            if cipherText[i] == " ":
                plainText += cipherText[i] # += " "
            else:
                plainText += chr((((ord(cipherText[i]) - 33) * k ) % 94) + 33 )
        return plainText
    return 0

def multiplicativeCipher_encode():
    print('Nhập bản rõ: ')
    plainText = str(input())
    print("Nhập k: ")
    k = int(input())
    while k < 0:
        print("Nhập lại k: ")
        k = int(input())
    cipherText = encrypt(plainText, k)
    print('Bản mã: %s' % cipherText)

def multiplicativeCipher_decode():
    print('Nhập bản mã: ')
    cipherText = str(input())
    print("Nhập k: ")
    k = int(input())
    while k < 0:
        print("Nhập lại k: ")
        k = int(input())
    plainText = decrypt(cipherText, k)
    if plainText == 0:
```

```
        print('Không thể giải mã !')
    else:
        print('Bản mã: %s' % plainText)

# Main
multiplicativeCipher_encode()
multiplicativeCipher_decode()
```

4. Bài tập thực hành

Tạo ứng dụng hoàn chỉnh với mật mã Ceasar và mật mã nhân: *Ứng dụng*.

Chú Thích

1. GCD – Greatest Common Divisor

2. ƯCLN – Ước chung lớn nhất

Tài Liệu Tham Khảo

[1]. Wikipedia, https://en.wikipedia.org/wiki/History_of_cryptography, truy nhập cuối cùng ngày 10/3/2019.

[2]. https://www.tutorialspoint.com/cryptography, truy nhập cuối cùng ngày 10/3/2019.