# Bài tập câu 1.

**Bài tập 1(2202).** E(K, M) || E(PR\_A, H(M))

A diagram of a circuit

Description automatically generated

**Bên gửi.** A gửi cho B đoạn mã gồm 2 phần ghép với nhau

* Phần 1. Mã hóa đoạn thông điệp M với khóa K bởi hàm mã hóa E thu được E(K, M)
* Phần 2. Mã háo đoạn thông điệp M với hàm hash thu được H(M). Tiếp đến đem mã hóa đoạn mã H(M) với khóa riêng người gửi PR\_A bởi hàm E, thu được E(PR\_A, H(M))

**Bên nhận.** B nhận được đoạn mã từ A:

* Đưa hàm mã hóa E(K, M) đưa qua hàm giải mã D với khóa K được thông điệp M.
* Mã hóa thông điệp M với hàm hash thu được H(M) (\*)
* Đưa hàm mã hóa E(PRa, H(M)) qua hàm giải mã D với khóa công khai PU\_A thu được H(M) (\*\*)
* So sánh hàm hash (\*) thu được và hàm hash (\*\*) trong đoạn mã để xác thực nội dung thông điệp có bị thay đổi trên đường truyền hay ko.

**Tác dụng.** Xác thực nội dung thông điệp có chính xác không, khóa E(K) có tác dụng bảo mật thông điệp, Khóa PRa: Xác thực nguồn gốc thông điệp.

**Bài tập 2 (2205).** E(PU\_B, [M || E(K\_1, H(M))])

A diagram of a chemical formula

Description automatically generated

**Bài giải.**

**Bên gửi.** A gửi B đoạn mã

* Phần 1: Mã hóa thông điệp M với hàm hash thu được H(M), Mã hóa H(M) với khóa K1 bởi hàm mã hóa E thu được E(K1,H(M)), lấy E(K1,H(M)) ghép nối với thông điệp M được M|| E(K1,H(M)). Đưa tất cả mã hóa với khóa công khai PUb bởi hàm mã hóa E thu được E(PUb, [M|| E(K1,H(M))])

**Bên nhận.** B nhận được đoạn mã từ A:

* Đưa hàm mã hóa E(PUb, [M|| E(K1,H(M))]) đưa qua hàm giải mã D với khóa PRb được thông điệp M|| E(K1,H(M))
* Mã hóa thông điệp M với hàm hash thu được H(M) (\*)
* Đưa hàm mã hóa E(K1,H(M)) qua hàm giải mã D với khóa K1 thu được H(M) (\*\*)
* So sánh hàm Hash (\*) thu được và hàm Hash (\*\*) trong đoạn mã để xác thực nội dung thông điệp có bị thay đổi trên đường truyền hay ko.

**Tác dụng.** Xác thực nội dung thông điệp có chính xác không, PUb: Bảo mật thông điệp trên đường truyền, E(K): Có tác dụng bảo mật

**Bài tập 3.** E(K1, M) || C(K2, E(K1, M)) || E(PU\_B, [K1 || K2])

**Bên gửi.** A gửi B đoạn mã gồm 3 phần ghép với nhau:

* Phần 1: Thông điệp M được mã hóa bởi hàm E với khóa K1
* Phần 2: Thông điệp M được mã hóa bởi hàm E với khóa K1 sau đó đưa qua hàm MAC với khóa K2 thu được C(K2,E(K1, M))
* Phần 3: Hàm mã hóa E gồm khóa K1 ghép với khóa K2 và mã hóa bởi khóa công khai PUB

**Bên nhận.** B nhận được đoạn mã từ A. Tách ra làm 3 phần:

* Giải mã hàm E(PUB, [K1||K2]) bằng khóa riêng PRB thu được 2 khóa K1 và K2
* Giải mã hàm E(K1, M) biết khóa K1 thu được thông điệp M
* Đưa hàm E(K1, M) qua hàm MAC với khóa K2 thu được hàm MAC C(K2, E(K1, M))(1)
* So sánh 2 hàm MAC (1)&(2) để xác thực nội dung thông điệp có bị thay đổi trên đường truyền hay ko

**Tác dụng:** Xác thực nội dung thông điệp có chính xác không – Có tác dụng giữ bí mật

**Bài tập 4.** E(K, M) || H(M) || E(PU\_B, K)

**Bên gửi.** A gửi B đoạn mã gồm 3 phần ghép với nhau:

* Phần 1: Mã hóa thông điệp M với khóa K bởi hàm mã hóa E thu được E(K,M)
* Phần 2: Đưa thông điệp M qua hàm Hash thu được H(M)
* Phần 3: Mã hóa khóa K với khóa công khai PUB thu được E(PUB, K)

**Bên nhận,** B nhận được đoạn mã từ A :

* Đầu tiên giải mã E(PUB, K) với hàm giải mã với khóa riêng PRB thu được khóa K, dùng khóa K giải mã E(K, M) thu được thông điệp M. Sau đo đưa M qua hàm Hash, kết quả nhận được so sánh với hàm hash trong đoạn mã để xác thực nội dung thông điệp có bị thay đổi trên đường truyền hay ko.

**Tác dụng**

* Xác thực nội dung thông điệp có chính xác hay ko
* Có tác dụng giữ bí mật thông điệp

**Bài tập 5.** E(K2, [M || C(K1, M)]) || E(PU\_B, [K1 || K2])

**Bên gửi.** A gửi B đoạn mã gồm 2 phần ghép với nhau:

* Phần 1: Thông điệp M được ghép mã MAC bởi hàm C với mật khóa K1 thu được C(K1, M) sau đó được ghép với thông điệp M và toàn bộ đoạn mã được mã hóa bởi hàm E với khóa K2 thu được E(K2, [M||C(K1, M)])
* Phần 2: Hàm mã hóa E gồm khóa K1 ghép với khóa K2 và mã hóa bởi khóa công khai PUB

**Bên nhận.** B nhận được đoạn mã từ A. Tách ra làm 2 phần:

* Giải mã hàm E(PUB, [K1||K2]) bằng khóa riêng PRB thu được 2 khóa K1 và K2
* Giải mã hàm E(K2, [M||C(K1, M)]) biết khóa K2 thu được thông điệp M và hàm C(K1, M) (1)
* Mã hóa hàm MAC với khóa K1 và thông điệp M thu được hàm C(K1, M) (2)
* So sánh 2 hàm MAC (1) & (2) để xác thực nội dung thông điệp có bị thay đổi trên đường truyền hay ko

**Tác dụng:** Xác thực nội dung thông điệp có chính xác không – Có tác dụng giữ bí mật

**Bài tập.** E(K1, M) || C(K2, E(K1, M)) || E(PUB, [K1 || K2])

Bên gửi. A gửi B đoạn mã có:

* Thông điệp M được mã hóa bởi hàm E với khóa K1 sau đó đoạn mã ghép với Thông điệp M được mã hóa bởi hàm E với mật khóa K1 sau đó hàm E đưa qua hàm MAC với mật khóa K2 thu được C(K2,E(K1, M)). Sau đó toàn bộ đoạn mã ghép với hàm E gồm Khóa K2 ghép với khóa K1 được mã hóa bởi khóa công khai PUB

Bên nhận. B nhận được đoạn mã từ A. Tách ra làm 3 phần

* Giải mã hàm E bằng khóa riêng PRB thu được 2 khóa K1 và K2
* Giải mã hàm E biết khóa K1 thu được thông điệp M
* Đưa hàm E(K1, M) qua hàm MAC với khóa K2 thu được hàm MAC(C(K2, E(K1, M)(1)
* So sánh 2 hàm MAC (1)&(2) để xác thực nội dung thông điệp có bị thay đổi trên đường truyền hay không

Tác dụng

* Xác thực nội dung thông điệp có chính xác không: Hàm MAC
* Có tác dụng giữ bí mật: : K1 ,Pu

**Bài tập.** E(K1, M) || C(K2, E(K1, M)

Bên A

* Thông điệp M đc mã hóa bằng khóa K1 nối tiếp
* Thông điệp M đc mã hóa bằng khóa K1 sau đó đem mã hóa qua hàm MAC với khóa K2 rồi đem gửi cho B

Bên B. Nhận chuỗi và giải mã

* Dùng khóa K1 giải mã ra M
* Đem thống điệp M mã hóa với khóa K1 sau đó lại đem mã hóa qua hàm Mác với khóa K2
* Đem 2 mã Mác so sánh với nhau nếu giống nhau thông điệp k bị thay đổi

**Tác dụng**

* Chứng thực: có
* Bảo mật khóa K1

**Bài tập.** E(PUB,[H(M) || M)

Bên A

* Thông điệp M đc đưa qua hàm hash nối với thông điệp M
* Sau đó được mã hóa bằng khóa công khai của B rồi đem gửi cho B

Bên B

* Nhận chuỗi và giải mã
* Dùng khóa riêng tư của B để giải mã thu được hàm hash của thông điệp M và thông điệp M
* Đưa thông điệp M vừa tìm được qua hàm hash
* So sánh 2 hàm 2 với nhau, nếu giống nhau thông điệp ko bị thay đổi

**Tác dụng:**

* Bảo mật: PUb(khóa công khai của b)
* Chứng thực: hàm hash

**Bài tập.** E(K, M) || H(M) || E(PUB, K)

Bên A:

* Thông điệp M đc mã hóa bởi hàm E với khóa K, ghép cùng mã hash của thông điệp M, sau đó ghép tiếp với khóa K được mã hóa bởi hàm E với khóa công khai của B

Bên B:

* B dùng khóa riêng PRB để giải mã thu được khóa K
* B sẽ dùng khóa K để giải mã ra thông điệp M, sau đó tính mã hash của thông điệp M và so sánh với mã hash ban đầu. Nếu đúng thì thông điệp không bị thay đổi

**Tác dụng của khóa công khai**

* Bảo mật: bảo mật chỗ PU
* Bảo mật thông điệp M bằng khóa K
* Bảo mật khóa K bằng khóa công khai của B

**Bài tập.** E(PUB, M) || E(PRA, H(M))

Bên A

* Thông điệp M được mã hóa bởi khóa công khai của B nối với mã hash của thông điệp M được mã hóa bởi khóa riêng tư của A đem gửi cho B

Bên B

* Dùng khóa công khai của A để giải mã ra mã hash của M
* Dùng khóa riêng tư của B để giải mã ra thông điệp M
* Đưa thông điệp M vừa giải đc qua hàm hash
* So sánh mã hash vừa tìm được với mã hash ban đầu. Nếu đúng thì thông điệp ko thay đổi.

Tác dụng

* Bảo mật : thông qua Pub
* Chứng thực : thông qua hàm h(), PRa

**Bài tập.** E(PUB, [M || E(PRA, C(K,M)])

Bên A

* Thông điệp M được mã hóa bằng hàm mác qua khóa K và được mã hóa 1 lần nữa bằng khóa riêng tư của A. Sau đó được mã hóa bằng khóa công khai của B. Đem gửi cho B

Bên B

* Nhận chuỗi giải mã
* Dùng khóa riêng tư của B giải mã thu được Thông điệp M và Thông điệp M được mã hóa bằng hàm MAC qua khóa K và được mã hóa 1 lần nữa bằng khóa riêng tư của A
* Dùng khóa công khai của A giải mã thu được Thông điệp M được mã hóa bằng hàm mác qua khóa K
* Mã hóa thông điệp M qua hàm MAC bằng khóa K sau đó so sánh với hàm mác ban đầu nếu đúng thì thông điệp ko bị thay đổi

**Tác dụng**

* Bảo mật: thông qua PUb
* Chứng thực: nguồn gốc thông qua Pra, nội dung C()

**Bài tập.** E(K1, [M || E(PRA, C(K2, M))])

Bên A

* Thông điệp M nối tiếp thông điệp M được mã hóa qua hàm MAC bằng khóa K2 sau đó đem mã hóa tiếp với khóa riêng tư của A.
* Đem tất cả mã hóa 1 lần nữa bằng khóa K1 rồi đem gửi cho B

Bên B

* Nhận chuỗi và giải mã
* Dùng khóa K1 giải mã được thông điệp M và thông điệp M đc mã hóa qua hàm Mác bằng khóa K2 sau đó đem mã hóa tiếp với khóa riêng tư của A
* Dùng mã công khai của A giải mã thu đc mã Mác của thông điệp M đc mã hóa bằng khóa K2
* Mã hóa thông thông điệp M bằng mã Mác với khóa K2 Đem 2 hàm MAC so sánh với nhau nếu giống nhau thông tin k bị thay đổi

Tác dụng

* Bảo mật K1
* Chứng thực : PRa, C()

**Bài tập.** E(K1,M) || E(PRA, C(K2,M))

Bên A

* Thông điệp M được mã hóa bởi khóa K1nối với thông điệp M được mã hóa qua hàm mạc bằng khóa K2 sau đó lại đc mã hóa một lần nữa bằng khóa riêng tư của A rồi đem gửi cho B

Bên B

* Nhận chuỗi và giải mã
* Dùng khóa công khai của A giải mã ra thông điệp M được mã hóa qua hàm mạc bằng khóa K2
* Dùng khóa K1 giải mã ra thông điệp M
* Dùng khóa K2 mã hóa thông điệp M qua hàm mac đem so sánh mã MAC vừa nhận đc với mã MAC đã giải. Nếu giống nhau thông điệp ko bị thay đổi

Tác dụng

* Bảo mật: thông qua K1
* Chứng thực: hàm MAC, PRa

**Bài tập.** E(K,[M || H(M)]) || E(PUB, K)

Bên A

* Thông điệp M nối với mã hash của M và được mã hóa bằng khóa K
* Nối với khóa K được mã hóa bằng khóa công khai của B rồi đem gửi cho B

Bên B.

* Nhận chuỗi và giải mã
* Dùng khóa riêng tư của B để giải mã và thu được khóa K
* Dùng khóa K giải mã thu được thông điệp M và mã hash của thông điệp M
* Tính mã hash của thông điệp M
* Đem so sánh hàm hash vừa mã hóa với hàm hash ban đầu

Tác dụng

* Bảo mật: PUb, K.
* Chứng thực thông qua hàm hash H

**Bài tập.** E(K1, M) || C(K2, E(K1, M) || E(PUB, [K1 || K2])

Bên A:

* Thông điệp M đc mã hóa bằng khóa K1 nối tiếp thông điệp M đc mã hóa bằng khóa K1 sau đó mã hóa bằng khóa K2 qua hàm MAC
* Nối tiếp khóa K1 nối tiếp khóa K2 đc mã hóa bằng khóa công khai của B rồi gửi đến B

Bên B

* Nhận chuỗi và giải mã
* Dùng mã riêng tư của B để giải mã ra khóa K1 và K2
* Dùng khóa K1 để giải mã thu được thông điệp M
* Đưa Thông điệp M đc mã hóa bằng khóa K1 sau đó mã hóa 1 lần nữa bằng khóa K2 qua hàm MAC
* So sánh 2 mã Mác vs nhau nếu giống thông điệp k bị thay đổi

**Bài tập.** E(K, M) || H(E(K, M))

Bên A

* Thông điệp M đc mã hóa bằng khóa K
* Nối tiếp thông điệp M đc mã hóa bằng khóa K đưa qua hàm hash rồi đem gửi cho

Bên B

* Nhận chuỗi vài giải mã. Dùng khóa K giải mã thu được thông điệp M. Sau đó đưa thông điệp M đc mã hóa bằng khóa K qua hàm hash tính mã hash rồi đem so sánh với mã hash ban đầu. Nếu đúng thì thông điệp không bị thay đổi

Tác dụng

* Bảo mật: qua khóa K
* Chứng thực nội dung: qua hàm Hash

**Bài tập 6.** E(K, [M || E(PR\_A, H(M))]) || E(PU\_B, K)

A diagram of a chemistry experiment

Description automatically generated

**Bên gửi.** A gửi B đoạn mã gồm 2 phần ghép với nhau:

* Phần 1: Mã hóa thông điệp M với hàm hash thu được H(M), Mã hóa H(M) với khóa PRa bởi hàm mã hóa E thu được E(PRa,H(M)), lấy mã thu được ghép nối với M ta được M || E(PRa,H(M)). Mã hóa M || E(PRa,H(M)) với khóa K bởi hàm mã hóa E thu được E(K, M || E(PRa, H(M)))
* Phần 2: Khóa K đưa qua khóa PUb của hàm E thu được E(PUb, K))

**Bên nhận.** B nhận được đoạn mã từ A

* Giải mã hàm E(K, M|| E(PRa, H(M))) qua hàm giải mã D với khóa K ta thu được M|| E(PRa,H(M))
* Mã hóa thông điệp M với hàm hash thu được H(M) (\*)
* Giải mã hàm E(PRa,H(M)) qua hàm giải mã D với khóa công khai PUb thu được H(M) (\*\*)
* So sánh hàm hash (\*) thu được và hàm hash (\*\*) trong đoạn mã để xác thực nội dung thông điệp có bị thay đổi trên đường truyền hay ko.

**Tác dụng:** Xác thực nội dung thông điệp có chính xác không

Khóa PRa: Xác thực nguồn gốc thông điệp, khóa PUb: Bảo mật thông điệp trên đường truyền, E(K): Có tác dụng bảo mật, khóa PRb: xác thực nguồn gốc thông điệp.

**Bài tập.** E(K,M) || H(E(K, M))

A diagram of a circuit

Description automatically generated

**Bên gửi.** A gửi B đoạn mã gồm 2 phần ghép với nhau

* Phần 1: Mã hóa thông điệp M với khóa K bởi hàm mã hóa E thu được E(K,M)
* Phần 2: Mã hóa thông điệp M với khóa K bởi hàm mã hóa E thu được E(K,M) rồi đưa toàn bộ E(K,M) qua hàm Hash thu được H(E(K,M))

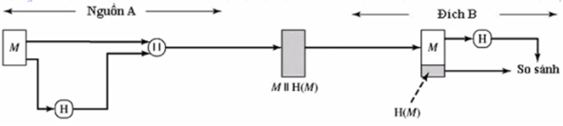
**Bên nhận.** B nhận được đoạn mã từ A

* Đưa hàm mã hóa E(K, M) đưa qua hàm hash thu được mã hask H(E(K, M)).
* So sánh hàm Hash thu được và hàm Hash trong đoạn mã để xác thực nội dung thông điệp có bị thay đổi trên đường truyền hay ko.

**Tác dụng:** Xác thực nội dung thông điệp có chính xác không.

Khóa E(K): Có tác dụng bảo mật.

#### **Bài tập 7.**



**Mục đích:** Biểu đồ giúp kiểm tra thông điệp có bị lỗi trên đường truyền

**Bên gửi**

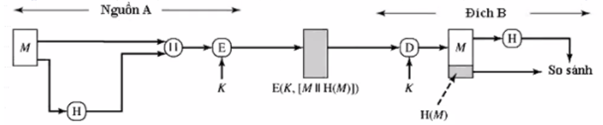
* Thông điệp M được đưa qua hàm Hash H thu được mã hash của M là H(M)
* Thông điệp M ghép với mã hash H(M) thu được M || H(M)

**Bên nhận**

* Đoạn mã nhận được thu được (M || H(M))
  + Phần 1: Thông điệp M đưa qua hàm hash H thu được mã H(M) (1)
  + Phần 2: Mã H(M)
  + Bên nhận so sánh 2 mã hash (1) & (2) xem thông điệp nhận được có chính xác hay không?

**Tác dụng của hệ thống.** Xác thực nội dung thông điệp (phát hiện thay đổi) có chính xác hay không?(So sánh 2 mã Hash)

**Bài tập 8.**



Figure

**Mục đích.** Đảm bảo rằng thông điệp M đến từ A và không bị thay đổi trên đường truyền (do lỗi đường truyền hoặc bị tấn công)

**Bên gửi**

* Thông điệp M được đưa qua hàm Hash H thu được mã hash của M (H(M))
* Thông điệp M ghép với mã hash H(M) thu được M || H(M)
* Toàn bộ đoạn mã M || H(M) được mã hóa bởi hàm E với khóa K

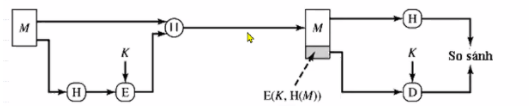
**Bên nhận**

* Đoạn mã nhận được đưa qua hàm giải mã với khóa K thu được (M || H(M))
* Phần 1: Thông điệp M đưa qua hàm hash H thu được mã H(M)(1)
* Phần 2: Mã H(M)
* Bên nhận so sánh 2 mã Hash(1) & (2) xem thông điệp nhận được có chính xác hay không?

**Tác dụng của hệ thống:**

* Xác thực nội dung thông điệp có chính xác hay không? (So sánh 2 mã Hash)
* Có khả năng giữ bí mật nội dung thông điệp

**Bài tập 9.**



Figure

**Mục đích.** Đảm bảo rằng thông điệp M đến từ A và không bị thay đổi trên đường truyền (do lỗi đường truyền hoặc bị tấn công)

Bên gửi

* Thông điệp M được đưa qua hàm Hash H thu được mã hash của M (H(M))
* Toàn bộ mã H(M) được mã hóa bởi hàm E với khóa K
* Thông điệp M được ghép với hàm E(K, H(M))

Bên nhận

* Đưa thông điệp M qua hàm Hash thu được mã H(M)(1)
* Hàm mã hóa E(K, H(M)) được đưa qua hàm giải mã với khóa K thu được mã H(M) (2)
* So sánh 2 mã Hash(1) & (2) để biết thông điệp M có chính xác hay không?

Tác dụng của hệ thống

* Xác thực nội dung thông điệp có chính xác hay không? (So sánh 2 mã Hash)
* Có khả năng giữ bí mật nội dung thông điệp

Nhận xét

* Giảm được khối lượng tính toán so với hình 1
* Không có khả năng giữ bí mật nội dung thông điệp

**Bài tập 10.**

A diagram of a diagram

Description automatically generated

Figure

**Mục đích.** Đảm bảo rằng thông điệp M đến từ A và không bị thay đổi trên đường truyền

**Bên gửi.**

* Thông điệp M được đưa qua hàm Hash H thu được mã hash của M (H(M))
* Mã H(M) được mã hóa bằng hàm E với khóa riêng Pra
* Thông điệp M được ghép với hàm E(PRa,H(M))

**Bên nhận**

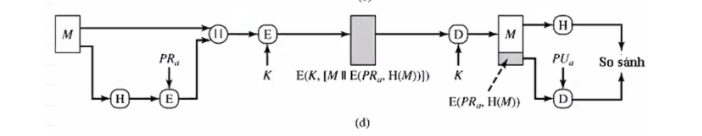
* Đưa thông điệp M qua hàm Hash thu được mã H(M) (1)
* Hàm mã hóa E(PRa,H(M)) được đưa qua hàm giải mã với khóa công khai PUa thu được mã H(M) (2)
* So sánh 2 mã Hash (1) và (2) để biết thông điệp M có chính xác hay không?

**Tác dụng của hệ thống:**

* Xác thực nội dung M có chính xác hay không (So sánh 2 mã hash (1), (2))
* Xác thực nguồn gốc thông điệp có phải do A gửi đến hay không (PRa) (chỉ có nguồn gốc bất đối xứng)
* Tuy nhiên ko có khả năng giữ bí mật nội dung thông điệp
* Đây chính là cơ sở của chữ ký số.

**Nhận xét:** Dùng mật mã bất đối xứng: a dùng mã hóa riêng của mình tức là đã tạo tiền đề chữ ký số (a muốn phủ nhận cũng ko đc)

**Bài tập 11.**



**Bên gửi**

* Thông điệp M được đưa qua hàm Hash thu được mã H(M), mã H(M) được mã hóa bằng hàm E với khóa riêng PRa thu được E(PRa, H(M))
* Thông điệp M ghép với hàm mã hóa E(PRa, H(M)) thu được (M||E(PRa,H(M)))
* Đoạn mã (M || E(PRa, H(M))) được mã hóa bằng hàm E với khóa K.

**Bên nhận.**

* Đoạn mã  nhận được đưa qua hàm giải mã với khóa K thu được (M || E(PRa, H(M)))
* Phần 1: Thông điệp M đưa qua hàm hash thu được mã H(M) (1)
* Phần 2: Hàm mã hóa E(PRa, H(M)) được đưa qua hàm giải mã với khóa công khai PUa thu được H(M) (2)
* Bên nhận so sánh 2 mã Hash (1) & (2)  xem thông điệp nhận được có chính xác hay không?

**Tác dụng của hệ thống:**

* Xác thực nội dung thông điệp có chính xác hay không ? (so sánh 2 mã Hash)
* Xác thực nguồn gốc thông điệp có phải do A gửi không (PRa)
* Bảo mật nội dung thông điệp(Nếu có ăn cắp thì cũng ko đọc được vì nó đã được mã hóa)

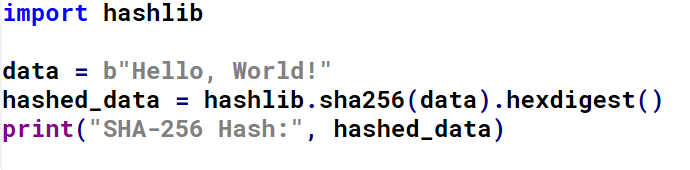
# Bài tập câu 2.

**Bài tập 1 (ChatGPT).** Loại hàm toán học nào thường được sử dụng trong mật mã khóa công khai? Đặc điểm cơ bản của chúng là gì? Hãy lấy ví dụ một hàm như vậy và giải thích.

Bài giải.

Trong mật mã khóa công khai, một số loại hàm toán học được sử dụng, và hai loại chính là Hàm Băm (Hash Function) và Hàm Toán Học Đối xứng (Asymmetric Cryptographic Function), thường là hàm toán học dựa trên vấn đề về số nguyên tố. Dưới đây là mô tả chi tiết về mỗi loại:

1. Hàm Băm (Hash Function):
   1. Đặc điểm Cơ Bản: Hàm băm được sử dụng để chuyển đổi dữ liệu vào một chuỗi cố định độ dài, thường là một mã băm có độ dài cố định. Nó có tính chất không đảo ngược, nghĩa là từ một giá trị băm, rất khó khăn để xác định dữ liệu ban đầu.
   2. Ví dụ: Một trong những hàm băm phổ biến là SHA-256 (Secure Hash Algorithm 256-bit). Nó tạo ra một mã băm có độ dài 256 bit từ dữ liệu vào.
2. Hàm Toán Học Đối xứng (Asymmetric Cryptographic Function):
   1. Đặc điểm Cơ Bản: Hàm toán học đối xứng thường dựa trên vấn đề về số nguyên tố và sử dụng một cặp khóa: khóa công khai (public key) và khóa riêng tư (private key). Dữ liệu được mã hóa bằng khóa công khai và chỉ có thể được giải mã bằng khóa riêng tư, và ngược lại.
   2. Ví dụ: Một trong những hệ thống mật mã khóa công khai phổ biến là RSA (Rivest–Shamir–Adleman). Trong hệ thống này, khóa công khai và khóa riêng tư là một cặp khóa, và một khóa được sử dụng để mã hóa và khóa kia được sử dụng để giải mã.



Lấy ví dụ là hàm sha256()

1. Chuẩn Băm SHA-256:
   1. SHA-256 là một thuật toán băm (hash) thuộc họ thuật toán SHA-2 (Secure Hash Algorithm 2) và tạo ra một mã băm có độ dài 256 bit (32 byte).
   2. Thuật toán này được thiết kế để tạo ra một mã băm duy nhất và khó khăn để tìm ra hai dữ liệu khác nhau có cùng mã băm.
2. Đầu Vào (data):
   1. data là dữ liệu đầu vào mà bạn muốn băm, có thể là một chuỗi ký tự (string) hoặc dữ liệu nhị phân (bytes).
   2. Ví dụ: b"Hello, World!" là một dữ liệu đầu vào được chuyển đổi từ chuỗi "Hello, World!" sang dạng bytes.
3. Sử Dụng hashlib Module:
   1. hashlib là một module trong Python cung cấp các hàm băm phổ biến, bao gồm SHA-256.
   2. Để sử dụng SHA-256, bạn sử dụng hashlib.sha256() để tạo một đối tượng hash cho thuật toán SHA-256.
4. Băm Dữ Liệu:
   1. Sau khi tạo đối tượng hash, bạn sử dụng phương thức update() để đưa dữ liệu vào và sau đó sử dụng hexdigest() để nhận giá trị mã băm dưới dạng chuỗi hex (hexadecimal).

**Bài tập 2 (2202).** Quy trình chuyển giao khóa phiên giữa client và server trong TLS diễn ra như thế nào? Quy trình đó có đảm bảo an toàn cho khóa phiên không, tại sao?

**Bài giải.**

Trình bày như SSL. Quy trình chuyển giao khóa phiên này được thiết kế để đảm bảo tính an toàn và bảo mật của khóa phiên. Mã hóa pre-master secret và sử dụng khóa riêng tư của server để giải mã giúp tránh được tình trạng tấn công MITM (Man-In-The-Middle). Đồng thời, quy trình này hỗ trợ Perfect Forward Secrecy (PFS), nơi mỗi phiên kết nối sử dụng một khóa phiên duy nhất. Điều này làm tăng tính an toàn, ngay cả khi một khóa phiên bị lộ.

**Bài tập.** Hàm HASH, SHA512, MAC là gì?

**Bài giải.**

(\*) Hàm hash:

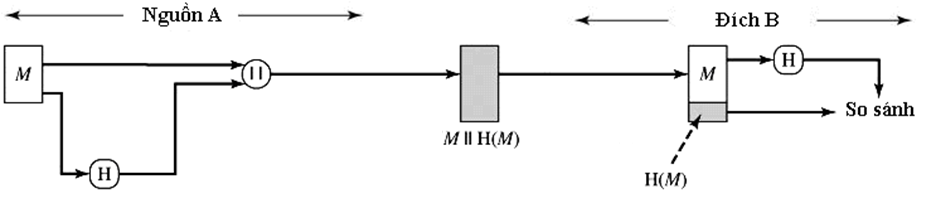
* Hàm Hash là một công cụ quan trọng trong chứng thực.
* Nhiệm vụ của hàm hash H là tác động lên thông điệp đầu vào M (có kích thước bất kì) để thu được một khối dữ liệu nhỏ H(M) có kích thước cố định ở đầu ra. H(M) được gọi là mã hash của M.
* H(M) đặc trưng cho M, nhưng thường có kích thước nhỏ hơn M, sẽ được sử dụng làm giá trị chứng thực.

Đặc điểm hàm hash

* H(M) là một hàm của tất cả các bít trong thông điệp M, chỉ cần thay đổi 1 bít của M cũng khiến H(M) bị thay đổi theo.
* Việc tính toán để tìm một giá trị N ≠ M thoả mãn điều kiện H(N) = H(M) là rất khó khăn (hoặc không thể).
* Không thể tính ngược để thu được M từ H(M).

**Ứng dụng**

* Mã hash thường được đính kèm và gửi cùng với thông điệp, nhằm mục đích chứng thực cho thông điệp đó là chính xác về mặt nội dung.
* Mã hash thường được dùng kết hợp với các mật mã khoá công khai hoặc đối xứng.



A diagram of a block diagram

Description automatically generated

* Mục đích: Đảm bảo rằng thông điệp M đến từ A và không bị thay đổi trên đường truyền (do lỗi đường truyền hoặc bị tấn công)
* Hình (b) giảm được khối lượng tính toán so với hình (a), nhưng không có khả năng giữ bí mật nội dung thông điệp như hình (a).

A diagram of a block diagram

Description automatically generated

Ký hiệu:

* PR: Private key – Khóa riêng (giữ bí mật)
* PU: Public key – Khóa công khai

Phân tích

* Hình (c): Đảm bảo rằng thông điệp M đến từ A và không bị thay đổi trên đường truyền, nhưng không có khả năng giữ bí mật nội dung thông điệp. Đây chính là cơ sở của chữ kí số.
* Hình (d) có thêm khả năng bảo mật cho thông điệp.

**(\*) Phân tích SHA-512**

Khái niệm

* Thông điệp đầu vào có chiều dài tối đa nhỏ hơn 2128 bít, chia thành N khối 1024 bít
* Đầu ra là một giá trị hash 512 bít

Quá trình thực hiện

* **Bước 1: Nối các bit đệm**. Thông điệp được độn thêm các bit sao cho chiều dài của nó khi chia cho chiều dài khối 1024 thì dư 896
* **Bước 2: Nối giá trị chiều dài**. Một giá trị 128 bit nhị phân được nối thêm vào cuối thông điệp. Khối này là một số nguyên không dấu, nó chứa chiều dài thực của thông điệp ban đầu (128+896=1024)
* Kết quả của hai bước đầu là thu được thông điệp có chiều dài là bội số của 1024 bit. Tổng chiều dài của thông điệp mới này là *N* × 1024 bit. Các khối của thông điệp được kí hiệu là *M*1, *M*2,..., *M*N.
* **Bước 3: Khởi tạo bộ đệm giá trị hash**. Một bộ đệm 512 bit được sử dụng để lưu trữ các kết quả trung gian và cuối chu trình sẽ là kết quả của hàm hash. Bộ đệm gồm tám thanh ghi 64-bit (*a*, *b*, *c*, *d*, *e*, *f*, *g*, *h*). Các thanh ghi này được khởi tạo với các số nguyên (thập lục phân) 64-bit tương ứng như sau: a = 6A09E667F3BCC908, b = BB67AE8584CAA73B, c = 3C6EF372FE94F82B, d = A54FF53A5F1D36F1, e = 510E527FADE682D1, f = 9B05688C2B3E6C1F, g = 1F83D9ABFB41BD6B, h = 5BE0CDI9137E2179
* **Bước 4: Xử lý thông điệp theo các khối 1024-bit (128-word)**. Trọng tâm của thuật toán là một module chứa 80 vòng; module này được kí hiệu là F. Cấu trúc logic của thuật toán được miêu tả trong hình sau:

A diagram of a schedule

Description automatically generated

* Trong đó:
  + Wi là các giá trị 64 bít được lấy ra từ 1024 bít của khối.
  + Ki là các hằng số ngẫu nhiên dài 64 bít, chúng được cộng vào mỗi vòng nhằm loại bỏ tính đều đặn của dữ liệu đầu vào.
* **Bước 5: Đầu ra.** Sau khi tất cả *N* khối 1024-bit đã được xử lý, đầu ra của giai đoạn thứ *N* là một thông điệp sắp xếp (message digest) hay mã hash, là một giá trị 512-bit.

**(\*) MAC**

* MAC (Message Authentication Code - Mã chứng thực thông điệp) là một kĩ thuật mã hoá nhằm biến đổi thông điệp đầu vào M (có kích thước bất kì) thành một khối dữ liệu nhỏ có kích thước cố định ở đầu ra.
* Khối dữ liệu đầu ra đó được gọi là mã MAC của thông điệp M. Nếu M bị thay đổi thì mã MAC của nó cũng sẽ thay đổi theo.
* Giống như mã hash, mã MAC thường được đính kèm và gửi cùng với thông điệp, nhằm mục đích chứng thực cho thông điệp đó là chính xác về mặt nội dung.
* Sự khác biệt cơ bản của MAC với các hàm hash là: Kĩ thuật mã hoá của MAC có sử dụng mật khoá *K*:
  + ***MAC* = C(*K*, *M*)**
  + *M* = Thông điệp đầu vào.
  + C = Hàm MAC.
  + *K* = Mật khoá chia xẻ giữa bên gửi và bên nhận.
  + *MAC* = Mã chứng thực thông điệp (Mã MAC).

A diagram of a graph

Description automatically generated with medium confidence

* Nếu có kẻ tấn công thay đổi thông điệp *M*, hắn sẽ không thể tạo ra mã *MAC* phù hợp vì không biết mật khoá *K*.
* Người nhận sẽ phát hiện ra sự giả mạo khi giá trị *MAC* nhận được và giá trị *MAC* mà họ tính toán từ *M* có sự khác nhau.

Một số cách dùng khác của MAC

A diagram of a block diagram

Description automatically generated

**Bài tập (ChatGPT).** Phân biệt Hash và Mac.

**Bài giải.**

Hash (hàm băm) và MAC (Message Authentication Code - mã xác thực tin nhắn) đều là các cơ chế mật mã được sử dụng trong bảo mật thông tin, nhưng chúng có mục đích và ứng dụng khác nhau. Dưới đây là sự phân biệt giữa Hash và MAC:

Hash (Hàm Băm):

* Mục Đích: Mục đích chính của hàm băm là chuyển đổi dữ liệu đầu vào thành một giá trị băm (hash) cố định với kích thước cố định. Nó không dùng để xác thực tính toàn vẹn hay nguồn gốc của dữ liệu.
* Đặc Điểm Cơ Bản:
  + Hàm băm không sử dụng khóa. Mỗi lần đầu vào giống nhau sẽ tạo ra một giá trị băm duy nhất.
  + Không có khả năng giải mã, nghĩa là từ giá trị băm, rất khó để xác định ngược lại dữ liệu đầu vào.
* Ứng Dụng:
  + Thường được sử dụng để lưu trữ mật khẩu (mật khẩu không được lưu trữ, chỉ lưu trữ giá trị băm).
  + Được sử dụng trong việc kiểm tra tính toàn vẹn của dữ liệu (ví dụ: checksum).

MAC (Message Authentication Code):

* Mục Đích: Mục đích chính của MAC là xác thực tính toàn vẹn và xác thực nguồn gốc của dữ liệu. Nó thường sử dụng một khóa để tạo ra mã xác thực.
* Đặc Điểm Cơ Bản:
  + Sử dụng khóa để tạo ra mã xác thực, điều này làm cho MAC có thể kiểm tra nguồn gốc của dữ liệu.
  + Cho phép xác thực và ngăn chặn sự thay đổi của dữ liệu trên đường truyền.
* Ứng Dụng:
  + Thường được sử dụng để xác thực tính toàn vẹn và nguồn gốc của tin nhắn hoặc dữ liệu truyền tải giữa các bên (ví dụ: trong giao thức TLS/SSL).

Có thể được sử dụng để xác thực tính toàn vẹn của dữ liệu lưu trữ.

* Tóm Lược:
  + Hash: Chuyển đổi dữ liệu thành giá trị băm, không sử dụng khóa, không thể giải mã, thường được sử dụng để kiểm tra tính toàn vẹn và lưu trữ mật khẩu.
  + MAC: Sử dụng khóa để tạo ra mã xác thực, giúp xác thực tính toàn vẹn và nguồn gốc của dữ liệu, thường được sử dụng trong việc xác thực giao tiếp và bảo vệ tính toàn vẹn.

**Bài tập 3.** Khóa riêng, khóa công khai là gì?

**Bài giải.**

Khóa riêng (private key) và khóa công khai (public key) là hai thành phần quan trọng trong hệ thống mật mã đối xứng (asymmetric cryptography). Đây là một phương pháp mật mã sử dụng cặp khóa này để thực hiện các quá trình mã hóa và giải mã thông tin.

Khóa Riêng (Private Key)

* Là một khóa bí mật, được giữ bởi người sử dụng hoặc hệ thống.
* Được sử dụng để giải mã thông tin đã được mã hóa bằng khóa công khai tương ứng của nó.
* Việc bảo mật và giữ an toàn khóa riêng là quan trọng, vì nếu ai đó có được khóa riêng, họ có thể giải mã thông tin mà chỉ người giữ khóa đó mới có thể thực hiện được.

Khóa Công Khai (Public Key)

* Là một khóa được công bố rộng rãi và có thể được chia sẻ với bất kỳ ai.
* Được sử dụng để mã hóa thông tin và tạo chữ ký số.
* Thông tin được mã hóa bằng khóa công khai chỉ có thể được giải mã bởi khóa riêng tương ứng. Do đó, người gửi thông tin có thể sử dụng khóa công khai của người nhận để mã hóa thông tin, đảm bảo rằng chỉ người nhận mới có thể giải mã thông tin đó.

**Bài tập 4.** Phân biệt các thuật toán mã hóa Caesar, Affine, MonoAlphabetic, PonoAlphabetic

|  |  |  |  |
| --- | --- | --- | --- |
| **Mật mã** | **Tổng quát** | **Ưu điểm** | **Nhược điểm** |
| Caesar | Mỗi kí tự trong bảng chữ cái được thay thế bởi một kí tự khác cùng bảng, cách sau nó ba vị trí. | Mật mã Caesar đơn giản, dễ thực hiện | Độ an toàn không cao, dễ bị bẻ khoá bởi tấn công Brute-force do số lượng khoá quá ít (chỉ có 25 khoá) |
| Affine | Kí tự P ban đầu được thay thế bởi kí tự C | Mật mã Affine có độ phức tạp lớn hơn mật mã Caesar tổng quát, số lượng khoá cũng nhiều hơn | Độ an toàn chưa cao, dễ bị phá bởi tấn công Brute-force do số lượng khoá chưa nhiều (chỉ có 312 khoá) |
| MonoAlphabetic | - Mỗi kí tự trong bảng chữ cái được thay thế bởi một kí tự bất kì khác cùng bảng.  - Chỉ sử dụng một bảng mã (mỗi kí tự plain text được thay thế bởi một kí tự cố định), nên không giấu được tần suất xuất hiện các kí tự. | Mật mã Monoalphabetic có số lượng khoá rất lớn, khó bẻ khoá bằng phương pháp Brute -force | Dễ bị tấn công bằng cách sử dụng các kỹ thuật tần suất ký tự và bigram.  Mã hóa không thay đổi theo ngữ cảnh, nên dễ bị tấn công. |
| PolyAlphabetic | Sử dụng nhiều bảng mã khác nhau (mỗi kí tự plain text có thể được thay thế bởi nhiều kí tự khác nhau, dựa trên các khoá thay thế khác nhau) | Khả năng chống lại các tấn công tần suất ký tự và bigram. | Vẫn có thể bẻ khoá mật mã này dựa trên các thống kê về các đặc điểm tự nhiên của ngôn ngữ  Cần quản lý nhiều khóa, gây khó khăn trong quản lý khóa. |

**Bài tập 5.** Phân biệt mật mã Playfair, mật mã Hill

|  |  |  |
| --- | --- | --- |
| **Mật mã** | **Mật mã Playfair** | **Mật mã Hill** |
| Tổng quát | Mật mã Playfair sẽ thay thế từng cặp 2 kí tự trong bản rõ bởi 2 kí tự tương ứng trong ma trận khoá 5 x 5.  Lần lượt viết từng kí tự của khóa vào ma trận, từ trái sang phải, từ trên xuống dưới, bỏ các kí tự trùng lặp.  Viết các ký tự còn lại trong bảng chữ cái vào ma trận theo thứ tự, I và J được coi như một ký tự.  Mỗi ký tự trong cặp plaintext sẽ được mã hoá bằng ký tự nằm cùng hàng với nó, nhưng cùng cột với ký tự kia  Nếu cặp ký tự plaintext rơi vào cùng một hàng của ma trận thì mỗi ký tự được thay thế bởi ký tự bên phải nó.  Nếu ký tự plaintext rơi vào cột cuối cùng, thì ciphertext của nó là ký tự cùng hàng ở cột đầu tiên.  Nếu cặp ký tự plaintext rơi vào chung một cột của ma trận thì mỗi ký tự được thay thế bởi ký tự ngay sát dưới.  Nếu ký tự plaintext rơi vào hàng cuối cùng, thì ciphertext của nó là ký tự cùng cột, ở hàng đầu tiên.  Nếu hai kí tự trong plaintext giống nhau thì chúng sẽ được được cách ly bằng một ký tự đại diện, chẳng hạn là **X**. | Mật mã Hill sẽ thay thế từng nhóm *m* kí tự trong plaintext bởi *m* kí tự ciphertext.  *m* kí tự ciphertext được xác định bởi hệ *m* phương trình tuyến tính  *C*i = (*k*i1*P*1 + ki2*P*2 + …+ *k*im*P*m) mod 26 |
| Ưu điểm | Mật mã Playfair có không gian khoá lớn tương tự mật mã Monoalphabetic nên khó bẻ được khoá bằng phương pháp Brute - force  Mật mã Playfair có khả năng che giấu một phần thông tin về tần suất xuất hiện các chữ cái, nhờ thực hiện mã hoá từng cặp hai kí tự | Độ an toàn của mật mã Hill sẽ càng lớn khi sử dụng ma trận K càng lớn   Mật mã Hill có khả năng che dấu hoàn toàn tần suất xuất hiện các kí tự đơn   Mật mã Hill rất mạnh khi chống lại tấn công chỉ biết ciphertext |
| Nhược | Mật mã Playfair dễ bị tấn công từ điển nếu người tấn công có thể có được key hoặc có khả năng đoán được key dựa trên thông điệp đã biết.  Nếu người tấn công biết được một phần của thông điệp hoặc có thể đoán được một số thông điệp, họ có thể sử dụng thông tin này để giải mã các phần còn lại của thông điệp.  Đối với các ngôn ngữ có cấu trúc ngôn ngữ rõ ràng, như tiếng Anh, mô hình ngôn ngữ có thể được sử dụng để đoán hoặc giả định các từ trong thông điệp. | Mật mã Hill dễ dàng bị bẻ gãy với một tấn công biết plaintext, do có thể dễ dàng xác định ma trận K từ các cặp P-C đã biết. |

**Bài tập 6.** Phân biệt mật mã Vigenere và Vernman

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Mật mã Vigenere** |  | **Vernman** |
| Tổng quát | Mật mã Vigenère sẽ thay thế từng nhóm *m* kí tự trong plaintext bởi *m* kí tự ciphertext.  *m* kí tự ciphertext được xác định bởi hệ *m* phương trình  Cm = (Pm + km) mod 26  Pm = (Cm - km) mod 26 |  | Plaintext được biểu diễn dưới dạng một chuỗi bít nhị phân  Khoá K cũng được biểu diễn dưới dạng một chuỗi bít nhị phân (càng dài càng tốt, càng ngẫu nhiên càng tốt)  Ciphertext được sinh ra bởi phép XOR giữa plaintext với khóa K |
| Ưu điểm | Như vậy, nếu *k1= k2 = … = km* thì mật mã Vigenère sẽ trở thành mật mã Caesar tổng quát.  Khi *k1 ≠ k2 ≠ … ≠ km*: một kí tự plaintext có thể được thay thế bởi nhiều kí tự khác nhau (ứng với các *k* khác nhau), nhờ vậy có thể che giấu được tần suất xuất hiện các kí tự. |  | Với khoá *K* đủ dài và ngẫu nhiên, các thông tin mang tính thống kê của ngôn ngữ có thể được che giấu hoàn toàn.  Sự ra đời của mật mã hệ nhị phân là tiền đề cho sự ra đời của các mật mã hiện đại. |
| Nhược điểm | Mật mã Vigenère vẫn không giấu được hoàn toàn tần suất xuất hiện các kí tự, và vẫn có thể bị phân tích  Giải pháp khắc phục là sử dụng hệ thống biểu diễn thông tin không mang tính thống kê của ngôn ngữ đó là hệ nhị phân |  | Nếu *K* ngắn thì sẽ phải sử dụng *K* lặp đi lặp lại, làm giảm tính ngẫu nhiên, và có thể làm lộ một phần thông tin về thống kê tần suất.  Tuy nhiên, việc sinh ra được một khoá K dài và thực sự ngẫu nhiên như vậy sẽ đòi hỏi nhiều công sức. |

**Bài tập 7.** Kỹ thuật Rain-Fence là gì?

|  |  |
| --- | --- |
| Tên | Kỹ thuật rain-fence |
| Tổng quát | Plaintext được viết dịch xuống tuần tự theo các đường chéo rồi đọc trình tự theo các hàng. |
| Ưu điểm |  |
| Nhược điểm | Mật mã hoán vị thuần túy rất dễ nhận ra bởi nó giữ nguyên tần suất xuất hiện ký tự đơn (và làm thay đổi tần suất của các cặp, các bộ kí tự của plaintext)   Để tăng độ phức tạp, người ta có thể tiến hành đổi chỗ nhiều lần, hoặc kết hợp với các thuật toán mã hoá khác. |

**Bài tập 8.** Phân biệt mật mã cổ điển và hiện đại

|  |  |  |
| --- | --- | --- |
| **Tên** | **Mật mã cổ điển** | **Mật mã hiện đại** |
| Đối tượng | Những ký tự (có thể là ký tự chữ viết, ký tự số hoặc các ký tự tượng hình) | Các bit thông tin |
| Công cụ thực hiện | Thủ công, cơ học hay các máy điện toán đơn giản | Sử dụng các thuật toán mật mã đối xứng và không đối xứng, bao gồm DES, AES, RSA, ECC, và nhiều thuật toán hash như SHA-256 |
| Ứng dụng | Thường được sử dụng cho việc bảo vệ thông tin quân sự và chính trị, nhưng ít hiệu quả trong môi trường mạng ngày nay. | Được sử dụng rộng rãi trong bảo mật thông tin, giao dịch trực tuyến, đảm bảo tính toàn vẹn và xác thực. |
| Yếu tố thời gian | Có từ lâu đời | Phát triển từ cuối thế kỷ 20 đến hiện tại. |

**Bài tập 9.** Các ưu điểm của các kỹ thuật mã hóa cổ điển

Các kĩ thuật mã hóa cổ điển gồm:

Kỹ thuật thay thế: Thuật toán mã hoá sẽ thay thế mỗi kí tự trong bản rõ bằng một kí tự khác

Đơn giản, dễ thực hiện

Độ an toàn không cao, dễ bị bẻ khóa

Kỹ thuật chuyển dịch – hoán vị: Các kí tự của plaintext sẽ được hoán đổi vị trí cho nhau để tạo thành ciphertext

Khó bẻ khóa hơn kỹ thuật thay thế

Che dấu được một phần thông tin về tần suất xuất hiện các kĩ tự trong mật mã

Mật mã hoán vị thuần túy rất dễ nhận ra bởi nó giữ nguyên tần suất xuất hiện ký tự đơn (và làm thay đổi tần suất của các cặp, các bộ kí tự của plaintext)

Để tăng độ phức tạp, người ta có thể tiến hành đổi chỗ nhiều lần, hoặc kết hợp với thuật toán mã hóa khác.

|  |  |
| --- | --- |
| Kỹ thuật thay thế | Kỹ thuật hoán vị |
| Có tác dụng làm *xáo trộn* thông tin thống kê của plaintext, làm phức tạp hóa mối quan hệ thống kê giữa ciphertext và mật khoá, nhằm ngăn cản nỗ lực tìm khoá. | Có tác dụng làm *khuếch tán* thông tin thống kê của plaintext, pha loãng các cấu trúc thống kê của plaintext ra một phạm vi rộng hơn, làm phức tạp hóa mối quan hệ thống kê giữa ciphertext và plaintext. |

**Bài tập 10.** Mã hóa đa kí tự là gì? Ưu và nhược điểm của loại mã hóa này.

Mã hóa đa ký tự (Polyalphabetic encryption) là một phương pháp mã hóa mà mỗi ký tự trong văn bản được ánh xạ với nhiều ký tự mã hóa tương ứng, thay đổi theo ngữ cảnh hoặc theo quy luật nào đó. Một số thuật toán mã hóa đa ký tự nổi tiếng bao gồm Vigenère, Autokey, và Playfair.

Ưu điểm của mã hóa đa ký tự

* Chống lại tấn công tần suất ký tự: Mã hóa đa ký tự làm cho mối quan hệ giữa các ký tự gốc và ký tự mã hóa trở nên phức tạp, khó dự đoán, do đó khó bị tấn công bằng cách phân tích tần suất xuất hiện của các ký tự.
* Tăng độ phức tạp: So với một số phương pháp mã hóa đơn giản như mã hóa Caesar, mã hóa đa ký tự có độ phức tạp cao hơn, giúp làm tăng độ khó khăn của quá trình giải mã.
* Khả năng chống lại các phương pháp tấn công thông thường: Mã hóa đa ký tự thường khó bị tấn công bằng cách sử dụng phương pháp tần suất chữ cái, bigram hoặc trigram, nếu thực hiện đúng.

Nhược điểm của mã hóa đa ký tự:

* Khó quản lý khóa: Nếu khóa không được quản lý cẩn thận, có thể dẫn đến việc giảm tính an toàn của hệ thống mã hóa đa ký tự.
* Khả năng bị tấn công Kasiski: Một số thuật toán mã hóa đa ký tự, như Vigenère, có thể bị tấn công bằng cách sử dụng phương pháp tìm kiếm khoảng cách giữa các trùng lặp trong văn bản mã hóa.
* Không thay đổi theo ngữ cảnh: Một số phương pháp mã hóa đa ký tự không thay đổi dựa vào ngữ cảnh của văn bản, điều này có thể tạo ra các mô hình dự đoán và tấn công.
* Không an toàn nếu không được sử dụng đúng cách: Nếu khóa được sử dụng một cách dễ đoán hoặc nếu quy luật của mô hình mã hóa được phát hiện, mô hình mã hóa đa ký tự có thể trở nên yếu đuối.

**Bài tập chương 3.4.** Chữ ký số và giao thức chứng thực

**Giao thức chứng thực**

* Giao thức là tập hợp các quy tắc để các bên tham gia liên lạc trao đổi thông tin với nhau
* Các giao thức chứng thực là giao thức giúp các bên tham gia liên lạc tự nhận dạng lẫn nhau và chuyển giao khoá phiên

Giao thức chứng thực dùng mã hóa đối xứng

* Giữa hai bên liên lạc A và B cần có một bên thứ ba gọi là KDC (Key Distribution Center - Trung tâm phân phối khóa tin cậy)
* Bên A sẽ chia sẻ khoá Ka với KDC
* Bên B sẽ chia sẻ khoá Kb với KDC
* KDC sẽ sinh ra khoá phiên Ks (session key). Ks chỉ tồn tại trong một thời gian ngắn để A và B thực hiện phiên liên lạc.

Giao thức Needham-Schroeder

1. A 🡪 KDC: IDA || IDB || N1
2. KDC 🡪 A: E(Ka, [Ks || IDB || N1 || E(Kb, [Ks || IDA])])
3. A 🡪 B: E(Kb, [Ks || IDA])
4. B 🡪 A: E(Ks, N2)
5. A 🡪 B: E(Ks, f(N2))

Nhận xét

* N1, N2 là các giá trị nonce (lời gọi), có tác dụng chống giả mạo
* Khi A gửi cho KDC một nonce (N1), theo quy ước KDC sẽ phải trả lời A bằng một giá trị tương ứng với N1, nếu không có nghĩa là thông điệp trả lời đã bị giả mạo

Quá trình

* KDC sinh ra khóa phiên Ks rồi gửi cho A
* A gửi thông điệp đã mã hóa (có chứa Ks) cho B
* B gửi lại A một nonce (N2), chứng tỏ B đã nhận được Ks, và muốn kết nối với A
* A trả lời B bằng một thông điệp có chứa N2. Phiên kết nối giữa A và B hình thành. Sau đó A và B có thể trao đổi các thông tin được mã hóa bởi Ks

Nhược điểm

* Chưa có cơ chế kiểm tra thời hạn sử dụng của khóa phiên Ks.
* Nếu kẻ tấn công (bằng cách nào đó) có được một khóa phiên Ks cũ, anh ta có thể dùng nó để đóng giả A nhằm lừa gạt B

Cụ thể

* Kẻ tấn công (X) lắng nghe và lấy được một thông điệp A gửi cho B ở bước 3 (có chứa khóa phiên cũ mà anh ta đã biết), sao chép thông điệp đó lại để dùng sau này.
* Khi cần tấn công, X chặn một thông điệp A gửi cho B ở bước 3, thay thế nó bằng thông điệp cũ đã lưu, và gửi nhại lại cho B
* Khi B trả lời ở bước 4 (dùng Ks cũ), X chặn thông điệp này và lấy được N2. Từ đó X có thể đóng giả A để nói chuyện với B

Khắc phục

* Cần gắn thêm nhãn thời gian T để thiết lập thời hạn sử dụng cho khóa phiên (xem giao thức Denning sau đây)

Giao thức Denning

1. A 🡪 KDC: *ID*A||*ID*B
2. KDC 🡪 A: E(*K*a, [*K*s||*ID*B||*T*||E(*K*b, [*K*s||*ID*A||*T*])])
3. A 🡪 B: E(*K*b, [*K*s||*ID*A||*T*])
4. B 🡪A: E(*K*s, *N*1)
5. A 🡪 B: E(*K*s, *f*(*N*1))

Nhận xét

* Khóa phiên Ks được gắn kèm với nhãn thời gian T
* B chỉ chấp nhận thông điệp (chứa khóa phiên Ks) ở bước 3 nếu giá trị của nhãn thời gian T đủ gần với thời gian hiện hành của B, bao gồm độ trễ mạng và sai số cho phép
* Đồng hồ của các bên tham gia liên lạc phải được đồng bộ hóa với nhau.

Lưu ý:

* Các giao thức trên chỉ áp dụng cho trường hợp các bên tham gia liên lạc đồng thời online trên mạng (ví dụ dịch vụ Chat)
* Nếu một bên tham gia liên lạc không online (ví dụ dịch vụ Email) thì cần có giải pháp khác

Trường hợp A, B không đồng thời trực tuyến (Chứng thực một chiều)

1. A 🡪 KDC: *ID*A||*ID*B||*N*1
2. KDC 🡪 A: E(*K*a, [*K*s||*ID*B||*N*1||E(*K*b, [*K*s||*ID*A])])
3. A 🡪 B: E(*K*b, [*K*s||*ID*A])||E(*K*s, *M*)

Nhận xét:

* Phương pháp này không sử dụng nhãn thời gian T nên vẫn có nguy cơ bị tấn công nhại.
* Tuy nhiên, việc xử lý email thường không tức thời, nên người nhận B chủ yếu dùng khoá phiên Ks để giải mã thông điệp M, chứ không dùng Ks để tiếp tục giao tiếp với A.

Giao thức chứng thực dùng mã hoá khoá công khai (của Denning)

* Giữa hai bên liên lạc A và B cần có một bên thứ ba gọi là AS (Authentication Server – Server chứng thực)
* AS không sinh ra khoá phiên Ks mà sẽ đưa ra “chứng chỉ” khoá công khai
* Khóa phiên được chọn và mã hóa bởi A

Quá trình

1. A 🡪 AS: *IDA*||*IDB*
2. AS 🡪 A: E(*PRas*, [*IDA*||*PUa*||*T*])||E(*PRas*, [*IDB*||*PUb*||*T*])
3. A 🡪 B: E(*PRas*, [*IDA*||*PUa*||*T*])||E(*PRas*, [*IDB*||*PUb*||*T*])||E(*PUb*, E(*PRa*, [*Ks*||*T*]))

Nhận xét

* AS sinh ra chứng chỉ khoá công khai (chứa khoá công khai của người dùng), mỗi chứng chỉ chỉ có hiệu lực trong khoảng thời gian T
* Khi A nhận được chứng chỉ do AS gửi, A sẽ biết được khoá công khai của B, và dùng nó để chuyển giao khoá phiên Ks cho B
* Khoá phiên được mã hoá 2 lần: Lần 1 bởi PRa để chứng thực, lần 2 bởi PUb để giữ bí mật

**Chữ ký số là**

* Chữ ký số là một cơ chế chứng thực cho phép tác giả thông điệp gắn thêm một đoạn mã vào thông điệp. Đoạn mã này đóng vai trò như một chữ ký.
* Chữ ký được tạo ra bằng cách tính giá trị hash của thông điệp và mã hóa nó bằng khóa riêng của tác giả.
* Chữ ký bảo đảm về nguồn gốc và tính toàn vẹn của thông điệp

**Tính chất chữ ký số**

* Phải xác minh được tác giả cùng thời gian của chữ ký.
* Phải chứng thực được nội dung thực sự vào thời gian ký.
* Phải kiểm tra được bởi một bên thứ ba, để giải quyết các tranh chấp.

**Phân loại**

* Chữ kí số trực tiếp
* Chữ kí số trọng tài

A diagram of a block diagram

Description automatically generated

A diagram of a diagram

Description automatically generated

**Phân tích chữ ký số trực tiếp**

* Loại chữ ký này chỉ liên quan đến hai bên tham gia (nguồn và đích).
* Một chữ ký số có thể tạo ra bằng cách mã hoá toàn bộ thông điệp bằng khoá riêng của người gửi (hình a), hoặc mã hoá mã hash của thông điệp với khoá riêng của người gửi (hình b).
* Có thể tăng cường tính bảo mật bằng cách thêm một lần mã hoá toàn bộ thông điệp và chữ ký bằng mật khoá chia sẻ (hình c) , hoặc bằng khoá-công-khai của của người nhận (hình d)
* Nhược điểm:
  + Chưa ghi lại thời gian phát sinh chữ kí
  + Thiếu cơ chế để xử lý khi xảy ra tranh chấp.
  + Người gửi có thể phủ nhận chữ kí của mình

**Phân tích chữ ký số trọng tài**

* Những nhược điểm của chữ ký số trực tiếp có thể được giải quyết nhờ vào bên thứ ba tin cậy (trọng tài).
* Giả sử cần gửi thông điệp từ X đến người nhận Y. Trước hết X phải kí vào thông điệp rồi gửi tới trọng tài A.
* A sẽ tiến hành kiểm tra thông điệp và chữ kí, nếu thấy hợp lệ thì sẽ gắn thêm thời gian và nhãn “đã kiểm tra” vào thông điệp, rồi gửi cho Y.
* Trọng tài đóng một vài trò chủ yếu và nhạy cảm trong hình thức này
* Tất cả các bên tham gia đều phải thỏa thuận về sự tin cậy tuyệt đối để cơ chế phân xử thực sự có hiệu quả.

**Ví dụ chữ ký số trọng tài**

Kí hiệu: X = người gửi, Y = người nhận, A = trọng tài, M = thông điệp, T = timestamp, nhãn thời gian để xác định thời gian phát sinh chữ kí, IDX = Tên người gửi X, Kxa = Khóa mà X và A biết, Kay = Khóa mà A và Y biết

**Ví dụ 1:** **Mã hóa đối xứng, trọng tài thấy nội dung thông điệp**

(1) X 🡪A: IDX|| M || E(Kxa, [IDX || H(M)])

(2) A 🡪Y: E(Kay, [IDX || M || E(Kxa, [IDX || H(M)]) || T])

**(\*) Bên gửi (1)**

* X gửi A đoạn mã có : Tên người gửi X (IDx) được ghép với thông điệp M và ghép với hàm mã hóa E gồm: hàm Hash H(M) ghép với IDx sau đó đưa qua hàm mã hóa E với Khóa Kxa (X,A biết).
* A nhận được đoạn mã từ X :
  + A tạm thời tin người gửi thông điệp là X và nhìn thấy thông điệp M.
  + Thông điệp M đưa qua hàm Hash thu được mã H(M) (1)
  + Giải mã hàm E với khóa Kxa thu được IDx và H(M) (2)
  + So sánh 2 IDx để biết thông điệp do X gửi
  + So sánh 2 thông điệp hash (1) & (2) để xác thực nội dung thông điệp không bị thay đổi.
* A xác nhận đúng, ký và gắn nhãn thời gian T và mã hóa toàn bộ đoạn mã bằng hàm E với khóa Kay. Làm xong tất cả 3 công việc trên thì A gửi đoạn mã tới Y

**(\*) Bên nhận (2)**

* Y giải mã bằng khóa Kay (A,Y biết) từ đó biết:
  + Người gửi là A (Kay)
  + Người gửi thông điệp là X (IDx)
  + Nội dung thông điệp là M
  + Thời gian phát sinh chữ ký là T
* X,Y phải tin tưởng tuyệt đối vào A

**(\*) Tác dụng**

* Xác thực nội dung thông điệp
* Xác thực nguồn gốc người gửi

**(\*) Nhận xét**

* Cả người gửi và người nhận đều phải tin tưởng tuyệt đối vào trọng tài A
* Nếu A không công tâm, anh ta có thể tự ý sửa đổi thông điệp, hoặc thông đồng với X để phủ nhận thông điệp đã kí, hoặc thông đồng với Y để giả mạo chữ kí của X.

**Ví dụ 2:** **Mã hóa đối xứng, trọng tài không thấy nội dung thông điệp**

(1) X 🡪A: IDX || E(Kxy, M) || E(Kxa, [IDX || H(E(Kxy, M))])

(2) A 🡪Y: E(Kay,[IDX || E(Kxy, M)]) || E(Kxa, [IDX || H(E(Kxy, M)) || T])

*K*xy: Khóa mà X và Y biết, A không biết

**(\*) Bên gửi (1)**

* X gửi A đoạn mã có : Tên người gửi X ghép với hàm mã hóa E mã hóa thông điệp M bằng khóa Kxy sau đó đem toàn bộ đoạn mã ghép với hàm mã hóa E gồm có thông điệp M được mã hóa bởi hàm E với khóa Kxy rồi toàn bộ hàm E(Kxy, M) đưa qua hàm Hash H thu được mã H(E(Kxy, M)) sau đó toàn bộ mã được ghép với IDx và cuối cùng toàn bộ được mã hóa bởi hàm E với khóa Kxa
* A nhận được đoạn mã
  + A tạm tin là người gửi thông điệp là X
  + Tiến hành giải mã hàm E với khóa Kxa đã biết thu được :IDx || H(E(Kxy,M) (1)
  + So sánh 2 IDx xác định người gửi là X .
  + Hàm mã hóa E(Kxy,M) được đưa qua hàm Hash thu được H(E(Kxy, M) (2)
  + So sánh 2 mã hash 1 & 2 xác thực thông điệp không bị thay đổi trên đường truyền
* A xác định người gửi đúng từ X và nội dung là M, A ký tên và gắn thêm nhãn thời gian T,sau đó đem toàn bộ đoạn mã mã hóa thêm lần nữa bằng khóa Kay chuyển đoạn mã tới Y

**(\*) Bên nhận (2)**

* Y nhận được đoạn mã từ A. Tiến hành giải mã bằng khóa Kay, từ đó biết được:
  + người gửi đoạn mã là A
  + Người gửi thông điệp là X (IDx)
  + Nội dung thông điệp là M (Giải mã bằng khóa Kxy)
  + Thời gian phát sinh chữ ký là T

**(\*) Tác dụng**

* Trọng tài A không thể đọc và giả mạo thông điệp M
* Bảo mật hơn vì trọng tài A không đọc được nội dung của thông điệp M do được mã hóa bởi khóa Kxy

**(\*) Nhận xét**

* Cả người gửi và người nhận đều phải tin tưởng tuyệt đối vào trọng tài A
* A không có khả năng sửa đổi thông điệp, nhưng vẫn có thể thông đồng với X để phủ nhận thông điệp đã kí, hoặc thông đồng với Y để giả mạo chữ kí của X.

**Ví dụ 3.** Mã hóa khoá-công-khai, trọng tài thấy nội dung thông điệp

(1) X 🡪A: IDX || M || E(PRx, [IDX || H(M)])

(2) A 🡪Y: E(PRa, [IDX || M || E(PRx, [IDX || H(M)]) || T])

**(\*) Bên gửi (1)**

* X gửi A đoạn mã gồm : Tên người gửi X ghép với thông điệp M ghép với hàm mã hóa E có hàm Hash H(M) được ghép với IDx rồi đem mã hóa bằng khóa riêng của X là PRx
* A nhận được đoạn mã từ X
  + A tạm tin người gửi thông điệp là X và nhìn thấy thông điệp M
  + A biết chắc rằng người gửi cho mình đúng là X(PRx)
  + Đưa hàm mã hóa E qua hàm giải mã với khóa công khai PUx thu được IDx và hàm hash H(M) (1)
  + So sánh 2 IDx để biết người gửi là X
  + Thông điệp M được đưa qua hàm Hash thu được H(M) (2)
  + So sánh 2 mã hash 1 & 2 để xác thực nội dung thông điệp M không bị thay đổi trên đường
* A xác nhận các nội dung thông điệp nhận được từ X là đúng, tiến hành ký tên và đính kèm thời gian ký tên là T, sau đó đem toàn bộ đoạn mã trên mã hóa bằng khóa riêng của a là PRa rồi gửi đến Y.

**(\*) Bên nhận (2)**

* Y sau khi nhận được đoạn mã từ A. Giải mã bằng khóa công khai PU\_A, từ đó biết được
  + Người gửi là A (Pra)
  + Sau đó biết người gửi thông điệp là X
  + Nội dung thông điệp là M
  + Thời gian phát sinh chữ ký là T
* Giải mã tiếp bằng khóa công khai PUx
  + So sánh 2 Idx
  + So sánh 2 hàm hash H(M)
* Y tự kiểm tra được nội dung thông điệp mà X gửi cho A

(\*) Tác dụng

* Xác thực nội dung thông điệp
* Xác thực nguồn gốc thông điệp

**(\*) Nhận xét.** A biết nội dung thông điệp, nhưng không thể sửa đổi thông điệp hay làm giả chữ kí

**Ví dụ 4:** Mã hóa khoá-công-khai. Trọng tài không thấy nội dung thông điệp

(1) X 🡪A: IDX || E(PRx, [IDX || E(PUy, E(PRx, M))])

(2) A 🡪Y: E(PRa, [IDX || E(PUy, E(PRx, M)) || T])

**(\*) Bên gửi (1)**

* X gửi A đoạn mã gồm: Tên người gửi X ghép với hàm mã hóa E có thông điệp M được mã hóa bởi khóa riêng PRx sau đó được mã hóa bằng khóa công khai PUy rồi đem đoạn mã ghép với IDx sau đó đem toàn bộ đoạn mã mã hóa bởi khóa riêng PRx.
* A nhận được đoạn mã từ X:
  + Biết người gửi là X(IDx), khi nhìn thấy PRx thì biết được nguồn gốc thông điệp là từ X
  + Giải mã bằng khóa công khai PUx thu được:
    - IDx, so sánh với IDx bên ngoài 🡪 tin rằng người gửi là X
    - Hàm mã hóa khóa công khai Puy 🡪 Không thể giải mã được vì không có khóa riêng của Y
* A xác nhận được người gửi là X, sau đó ký tên và đính kèm thời gian ký tên là T rồi mã hóa toàn bộ đoạn mã bằng khóa riêng PRa.

**(\*) Bên nhận (2)**

* Y nhận được đoạn mã từ A. Giải mã bằng khóa công khai PUa(Người gửi cho mình là A) thu được
  + IDx 🡪 Người gửi thông điệp là X
  + Thời gian ký tên là T
  + Giải mã bằng khóa riêng PRy sau đó giải mã bằng khóa công khai PUx thu được thông điệp M.

**(\*) Tác dụng**

* Xác thực nội dung
* Xác thực nguồn gốc

(\*) **Nhận xét 4.** A không biết nội dung thông điệp, không thể sửa đổi thông điệp hay làm giả chữ kí

**Bài tập 11 Chương 5.** Phân biệt Virus Boot, Virus File thi hành, Virus Macro

|  |  |  |  |
| --- | --- | --- | --- |
| **Virus Boot** |  | **Virus File thi hành** | **Virus Macro** |
| Loại virus này lây lan vào đoạn mã khởi động trên Boot sector hay Master Boot Record của đĩa  Khi máy tính khởi động, virus boot được kích hoạt. Nó sẽ thường trú trong bộ nhớ, chờ để lây vào một ổ đĩa mới   Nếu đem ổ đĩa nhiễm virus lắp sang một máy tính khác, rồi khởi động máy từ ổ đĩa đó, máy tính sẽ bị nhiễm virus. |  | Loại virus này lây nhiễm vào các file nhị phân thi hành được (.EXE, .COM, .DLL, .BIN, .SYS…)  Đoạn mã virus có thể được gắn vào đầu file, cuối file, hoặc giữa file   Khi file được chạy, virus sẽ được kích hoạt, nó sẽ tìm cách lây vào các file khác trong máy | Loại virus này lây nhiễm vào các macro trong các file tài liệu của MicroSoft Office (Word, Excel…)  Một số macro có khả năng tự khi hành khi mở file, cất file… Virus Macro thường nằm trong các macro tự động đó.   Khi file được mở, virus sẽ được kích hoạt, sau đó nó tìm cách lây vào các file tài liệu khác. |

**Bài tập 12 (2205) (ChatGPT).** Worm và virus máy tính có gì giống và khác nhau? Kỹ thuật tìm diệt chúng khác nhau như thế nào?

A screenshot of a computer

Description automatically generated

**Bài giải.** Worm và virus máy tính đều là các loại mã độc hại nhưng có những điểm giống nhau và khác nhau. Dưới đây là mô tả về sự tương đồng và khác biệt giữa chúng, cùng với kỹ thuật tìm diệt khác nhau:

**Giống nhau**

1. Lây nhiễm và Tự sao chép: Cả worm và virus đều có khả năng tự sao chép và lây nhiễm các tập tin hoặc hệ thống khác để lan truyền.
2. Gây hại cho Hệ thống: Cả hai đều có thể gây hại cho hệ thống bằng cách xóa dữ liệu, làm hỏng tệp tin, hoặc chiếm quyền kiểm soát của máy tính.

**Khác nhau**

1. Cách Lan Truyền
   1. Virus: Yêu cầu sự tương tác của người dùng hoặc sự kích thích từ bên ngoài để kích hoạt và lan truyền. Ví dụ, virus có thể kết hợp với các tệp tin thực thi hoặc email độc hại.
   2. Worm: Có thể tự lan truyền mà không cần sự tương tác từ người dùng. Chúng sử dụng các lỗ hổng bảo mật trong hệ thống để tự động sao chép và lan truyền.
2. Kích thước
   1. Virus: Thường nhỏ và cần một tệp tin thực thi để kích hoạt.
   2. Worm: Thường lớn hơn và có thể tự chạy mà không cần tệp tin thực thi ngoại lệ.
3. Kỹ thuật Bảo vệ
   1. Virus: Có thể được phát hiện bằng các chương trình chống virus thông thường dựa trên chữ ký hoặc hành vi của chúng.
   2. Worm: Thường yêu cầu các kỹ thuật bảo vệ phức tạp hơn do khả năng lan truyền tự động và sự đa dạng trong các phương tiện tấn công.
4. Mục Tiêu
   1. Virus: Thường nhắm vào các tệp cụ thể hoặc hệ thống máy tính.
   2. Worm: Thường nhắm vào các mạng lớn hơn và có thể lan truyền rộng rãi.
5. Kỹ thuật Tìm Diệt
   1. Chống Virus: Sử dụng chương trình chống virus để quét và xác định chữ ký của virus trong các tệp tin thực thi hoặc trong bộ nhớ hệ thống.
   2. Chống Worm: Sử dụng các công cụ chống malware và firewall để giảm khả năng bị nhiễm worm thông qua các lỗ hổng bảo mật. Cập nhật hệ điều hành và phần mềm bảo mật là quan trọng để đóng các lỗ hổng có thể được sử dụng bởi worm.

**Bài tập**. Kỹ thuật lây lan Virus, ngụy trang, cách phòng tránh.

**(\*) Kỹ thuật lây lan virus**

* Muốn lây lan thì virus phải được kích hoạt. Khi virus được kích hoạt, mã lệnh của nó sẽ tìm cách tự sao chép tới các file khác hoặc đĩa khác.
* Có nhiều cách để kích hoạt một virus, chủ yếu là lợi dụng các sơ hở của hệ thống, hoặc tìm cách “dụ” người dùng chạy file nhiễm virus.
* Hacker viết virus, gắn nó vào một chương trình (do anh ta tự viết hoặc ăn cắp), rồi phát tán chương trình đó (ví dụ bằng cách gửi nó lên một website cung cấp phần mềm miễn phí). Ai download và chạy chương trình này sẽ kích hoạt virus
* Có thể đính kèm file nhiễm virus vào email, rồi gửi cho mọi người. Khi mở file đính kèm thì virus sẽ được kích hoạt
* Virus “đóng giả” các file tài liệu hoặc Folder (bằng cách đổi tên, đổi icon…) để “lừa” người dùng bấm chuột vào
* Lợi dụng tính năng Auto Run của các ổ đĩa USB, CD để kích hoạt virus

**(\*) Kỹ thuật ngụy trang virus**

* Để chống lại các phần mềm diệt virus, virus phải có giải pháp nguỵ trang, che dấu bản thân
* Sau khi lây vào một file mới, virus thường sửa đổi lại ngày, giờ, thuộc tính file… cho giống với file chưa bị lây nhiễm
* Có thể áp dụng phương pháp nén để giữ nguyên kích thước của file sau khi lây
* Dấu mã lệnh virus vào các vùng an toàn trên đĩa, và đánh dấu chúng là các sector hỏng (đối với virus boot hoặc virus lai).
* Liên tục biến đổi mã lệnh virus để thay đổi bản thân, tránh bị nhận dạng (virus đa hình).

**(\*) Kỹ thuật lây lan worm**

* **Tìm cách thi hành chương trình từ xa:** Một số máy cho phép chạy chương trình shell ở xa mà không yêu cầu xác thực. Nếu thành công, shell ở xa sẽ tải chương trình worm về và tiếp tục lây vào các máy khác kết nối với nó
* **Tìm cách đăng nhập từ xa:** Worm có thể sử dụng chương trình đoán mật khẩu, lần lượt thử đăng nhập với một danh sách mật khẩu được chuẩn bị từ trước. Nếu đoán mật khẩu thành công, Worm có thể đăng nhập được vào bất cứ máy tính nào người dùng đó có tài khoản.
* **Tận dụng lỗi tràn bộ đệm:** Worm Morris đã tận dụng lỗi tràn bộ đệm của chương trình finger (có trên một số website) để kích hoạt Worm trên máy bị tấn công.
* **Tận dụng lỗi của hệ thống email:** Một số hệ thống email cho phép chương trình worm gửi thư có chưa phần khởi động cho người khác và thi hành nó

**(\*) Các bước lây lan của worm**

* Tìm kiếm các hệ thống khác để lây nhiễm bằng cách khảo sát các bảng host hay những nơi chứa các địa chỉ các hệ thống từ xa.
* Thiết lập kết nối tới hệ thống từ xa.
* Tự sao chép đến hệ thống từ xa và kích hoạt để tự thi hành.

**(\*) Phòng tránh**

* Thứ nhất, hãy chọn một hệ điều hành có độ an toàn cao, chế độ kernel và chế độ người dùng có ranh giới rõ ràng, người dùng và người quản trị hệ thống phải có mật khẩu đăng nhập riêng. Khi đó, nếu virus có lẻn vào thì cũng khó có thể lây vào hệ thống nhị phân.
* Thứ hai, chỉ cài đặt các phần mềm chính thống mua từ các nhà sản xuất đáng tin cậy. Không download các phần mềm không rõ nguồn gốc từ Internet.
* Thứ ba, hãy mua một chương trình diệt virus tốt và sử dụng nó thường xuyên. Hãy luôn cập nhật phiên bản mới nhất của nó từ Website của nhà sản xuất.
* Thứ tư, đừng bao giờ bấm chuột vào các file thi hành đính kèm trong email, hay các file thi hành tự nhiên xuất hiện trong ổ đĩa USB của bạn. Nói chung, phải cận thận với tất cả các file có thể chứa mã lệnh!
* Thứ năm, thường xuyên sao lưu các file quan trọng ra bộ nhớ ngoài (như đĩa USB, đĩa CD), đề phòng trường hợp máy tính của bạn bị nhiễm virus, bạn vẫn có thể khôi phục lại các file về trạng thái ban đầu.
* Thứ sáu, thường xuyên cập nhật các bản vá lỗi mới nhất của hệ điều hành, của trình duyệt web, và các phần mềm khác.

**Bài tập 13 (ChatGPT) Chương 4.2.** Giao thức SSL là gì? Nêu các bước thực hiện kết nối SSL

**Bài giải.**

* SSL một tiêu chuẩn của công nghệ bảo mật, tạo ra một liên kết được mã hóa giữa máy chủ web và trình duyệt.
* Liên kết này đảm bảo tất cả các dữ liệu trao đổi giữa máy chủ web và trình duyệt luôn được bảo mật và an toàn.
* SSL được sử dụng rộng rãi trên hàng triệu Website ở khắp thế giới, giúp bảo vệ dữ liệu an toàn trên môi trường internet
* SSL cung cấp:
  + Sự bảo mật: Các dữ liệu truyền tải giữa máy chủ và trình duyệt sẽ được mã hoá, đảm bảo tính riêng tư và toàn vẹn.
  + Khả năng chứng thực: Mỗi chứng chỉ số SSL được tạo ra cho một Website duy nhất, khẳng định độ tin cậy của Website đó.
  + Một cơ quan uy tín sẽ xác thực danh tính và độ tin cậy của Website trước khi cấp chứng chỉ SSL cho Website.

Các bước thực hiện SSL

A screenshot of a chat

Description automatically generated

1. Client Gửi Yêu Cầu Kết Nối: Client gửi yêu cầu kết nối tới server thông qua giao thức SSL/TLS.
2. Server Gửi Chứng Chỉ (Certificate): Server gửi chứng chỉ kỹ thuật số của mình tới client. Chứng chỉ chứa khóa công khai của server và thông tin xác minh.
3. Client Xác Nhận Chứng Chỉ: Client kiểm tra chứng chỉ của server để đảm bảo rằng nó được ký bởi một CA (Certificate Authority) tin cậy và không bị thu hồi.
4. Client Tạo Khóa Phiên (Session Key): Client tạo một khóa phiên (session key) mới, được sử dụng cho việc mã hóa và giải mã dữ liệu trong suốt phiên kết nối.
5. Client Gửi Khóa Phiên Đã Mã Hóa đến Server: Client sử dụng khóa công khai của server để mã hóa khóa phiên, sau đó gửi nó đến server.
6. Server Giải Mã Khóa Phiên: Server sử dụng khóa riêng tư của mình để giải mã khóa phiên đã nhận từ client.
7. Kết Nối Bảo Mật Được Thiết Lập: Với khóa phiên đã chia sẻ, kết nối SSL/TLS được thiết lập giữa client và server. Từ bước này trở đi, dữ liệu được truyền qua lại giữa client và server sẽ được mã hóa và giải mã bằng khóa phiên.
8. Gửi Xác Nhận Cho Client: Server gửi một xác nhận đến client để thông báo rằng quá trình thiết lập kết nối đã hoàn tất và dữ liệu có thể bắt đầu được truyền tải an toàn.

**Bài tập 14 chương 4.3 (slide).** SET là gì?

SET (Secure Electronic Transaction – Giao dịch điện tử an toàn) được thiết kế để bảo vệ các giao dịch bằng thẻ tín dụng qua Internet. SET ra đời vào tháng 2/1996 do sự đòi hỏi của các chuẩn an ninh cho MasterCard và Visa.

SET là một tập các giao thức an ninh và các dạng thức cho phép người dùng thanh toán bằng thẻ tín dụng trong một mạng mở (như Internet) một cách an toàn

SET cung cấp ba dịch vụ

* Cung cấp kênh liên lạc an toàn giữa các thực thể góp mặt trong một giao dịch.
* Cung cấp tính tin cậy bằng cách dùng các chứng chỉ số X.509v3.
* Bảo đảm tính riêng tư, do các thông tin chỉ được cung cấp cho các thực thể khi tham gia giao dịch và vào lúc cần thiết.

Các đặc điểm chính của SET

* Tính tin cậy của thông tin
  + Các thông tin về người dùng thẻ và tài khoản tương ứng được bảo vệ an toàn khi lưu thông trên mạng
  + Ngay cả người bán cũng không thể nắm giữ được số thẻ tín dụng của người sở hữu, số hiệu này chỉ được cung cấp cho ngân hàng phát hành thẻ
  + Mã hóa DES được sử dụng để cung cấp tính tin cậy
* Tính toàn vẹn dữ liệu
  + Thông tin thanh toán được gửi từ một người dùng thẻ tới nhà bán hàng bao gồm các thông tin đặt hàng, dữ liệu cá nhân, và các lệnh chi trả.
  + SET đảm bảo rằng nội dung các thông điệp này không bị biến đổi trên đường đi.
  + Chữ ký số RSA dùng mã hash SHA-1 được sử dụng để đảm bảo tính toàn vẹn của thông điệp.
* Chứng thực tài khoản của người dùng thẻ
  + SET cho phép các nhà bán hàng xác minh một người dùng thẻ là hợp pháp đối với một thẻ tín dụng hợp lệ và tài khoản ứng với thẻ đó.
  + Chứng chỉ số X.509v3 chứa các chữ ký RSA được sử dụng cho chức năng này.
* Chứng thực nhà bán hàng
  + SET cho phép người sở hữu thẻ xác minh được rằng nhà bán hàng đã sẵn có mối quan hệ với cơ quan tài chính hỗ trợ thanh toán bằng thẻ.
  + SET dùng các chứng chỉ số X.509v3 với các chữ ký RSA cho chức năng này

**Bài tập chương 4.1.** Ứng dụng chứng thực chứng chỉ khóa công khai là gì?

Việc phân phối khoá công khai cũng cần phải có phương pháp. Một số cách để phân phối khoá công khai như:

* Yết thị công khai
  + Người sở hữu khoá (A) có thể gửi khoá công khai của mình cho đối tác, hoặc công bố rộng rãi cho cộng đồng
  + Nhược điểm: Do không có cơ chế xác thực, bất cứ ai cũng có thể đóng giả làm A và gửi khoá công khai cho đối tác của A, khiến đối tác nhầm tưởng là đang liên lạc với A.
* Đặt trong một thư mục công cộng
  + Các khoá công khai được đặt trong một thư mục công cộng do một tổ chức đáng tin cậy quản lý.
  + Người dùng muốn công bố khoá công khai của mình thì phải đăng kí với tổ chức đó.
  + Nhược điểm: Nếu kẻ tấn công (bằng cách nào đó) chiếm được quyền truy cập của tổ chức, anh ta có thể thay thế các khoá công khai trong thư mục bằng các khoá giả mạo.
* Dùng chứng chỉ khoá công khai
  + Khoá công khai không được đặt trực tiếp trong thư mục công cộng, mà được đặt trong “chứng chỉ khoá công khai”
  + Người dùng A muốn công bố khoá công khai của mình thì phải đăng kí với một tổ chức đáng tin cậy có thẩm quyền cấp chứng chỉ, tổ chức này sẽ cấp cho A một chứng chỉ bao gồm khoá công khai của A (PUa), định danh của A (IDA), và chữ kí của tổ chức.
  + A có thể công bố rộng rãi chứng chỉ này hoặc gửi chứng chỉ cho đối tác, mà không phải lo lắng bị kẻ khác làm giả chứng chỉ.
  + Đối tác của A có thể kiểm tra tính hợp lệ của chứng chỉ dựa trên chữ kí của tổ chức.

Mô hình

A diagram of a certificate authority

Description automatically generated

* CA: Certificate Authority (Tổ chức cấp phát chứng chỉ)
* CA: là chứng chỉ khoá công khai của A. *C*A = E(*PR*auth, [*T*||*ID*A||*PU*a])
* *PR*auth là khóa riêng của tổ chức cấp phát chứng chỉ.
* *T* là một nhãn thời gian cho biết thời gian hiệu lực của chứng chỉ
* Đối tác của A có thể dùng *PU*auth để giải mã CA, nhằm kiểm tra tính hợp lệ của CA và thu được PUa:

D(*PUauth*, *CA*) = D(*PUauth*, E(*PRauth*, [*T*||*IDA*||*PUa*]))

= [*T*||*IDA*||*PUa*]

* Sau đó đối tác có thể liên lạc với A nhờ PUa.

**Bài tập chương 4.1.** Hạ tầng khoá công khai là gì?

Hạ tầng khoá công khai (PKI - Public Key Infrastructure) là một tập phần cứng, phần mềm, con người, chính sách và các thủ tục cần thiết để tạo ra, quản lý, lưu trữ, phân phối và thu hồi các chứng chỉ số dựa trên kỹ thuật mật mã bất đối xứng.

Hạ tầng khoá công khai dựa trên tiêu chuẩn X.509 được kí hiệu là PKIX

Mô hình

A diagram of a flowchart

Description automatically generated

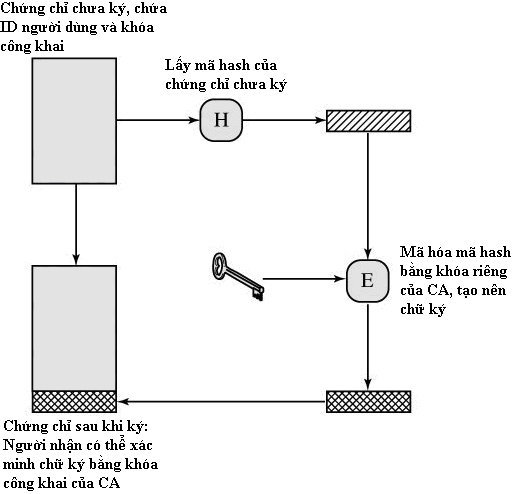
* **CA (Certification Authority**): Cơ quan cấp phát chứng chỉ, quản lý, thu hồi và công bố danh sách các chứng chỉ bị thu hồi (CRL)
* **RA (Registration Authority**): Cơ quan đăng kí cấp phát chứng chỉ, có nhiệm vụ tiếp nhận và xác minh các yêu cầu về chứng chỉ số của người dùng, rồi gửi các yêu cầu đã xác minh cho CA để thực hiện yêu cầu đó.
* **Thực thể cuối (End Entity)**: là người dùng hay các bên sử dụng dịch vụ. Người dùng có thể gửi yêu cầu xin cấp chứng chỉ trực tiếp cho CA hoặc gián tiếp qua RA
* **Bên ban hành CRL (CRL issuer)**: CA có thể trực tiếp công bố danh sách các chứng chỉ bị thu hồi (CRL), hoặc giao cho CRL issuer công bố

**Bài tập 15 chương 4.1 (slide).** Dịch vụ chứng thực X.509 là gì?

**Bài giải.**

X.509 là một tiêu chuẩn về chứng chỉ khoá công khai được chấp nhận rộng rãi trên toàn thế giới. Các chứng chỉ X.509 được sử dụng trong hầu hết các ứng dụng về an ninh mạng. Mỗi chứng chỉ X.509 ứng với một người dùng, được tạo ra bởi CA (Tổ chức cấp chứng chỉ tin cậy). Các chứng chỉ sẽ được đặt vào một thư mục công khai của CA, hoặc đặt vào thư mục của người dùng. Người dùng có thể truy xuất thư mục của CA để lấy chứng chỉ của người khác, hay tự phân phối chứng chỉ của mình cho người khác, mà không lo bị giả mạo.

Quá trình sinh chứng chỉ X.509



Hãy biểu diễn sơ đồ trên bằng công thức.

Khi A gửi chứng chỉ X.509 của mình cho B, B sẽ kiểm tra chứng chỉ đó như thế nào? (GPT)

Khi A gửi chứng chỉ X.509 của mình cho B, quy trình kiểm tra chứng chỉ thường bao gồm các bước sau đây:

1. Xác định Tính Hợp Lệ của Chứng chỉ: B sẽ kiểm tra xem chứng chỉ X.509 có hợp lệ không. Điều này bao gồm kiểm tra xem chứng chỉ được ký bởi một đơn vị cấp chứng chỉ (CA) tin cậy, có thời gian hiệu lực và không bị thu hồi (revoke) hay không.
2. Kiểm Tra Chữ Ký Số: B sẽ sử dụng khóa công khai của CA để kiểm tra chữ ký số của chứng chỉ. Nếu chữ ký được kiểm tra thành công, điều này chứng minh rằng chứng chỉ không bị giả mạo và đã được ký bởi CA.
3. Xác Nhận Danh Tính: B sẽ xác nhận danh tính của chủ sở hữu chứng chỉ (A). Thông thường, thông tin về chủ sở hữu chứng chỉ được đặt trong trường "Subject" của chứng chỉ và có thể bao gồm tên, địa chỉ email, và các thông tin xác nhận danh tính khác.
4. Kiểm Tra Quyền Lực của Chủ Sở Hữu: Nếu chứng chỉ chứa thông tin về quyền lực (ví dụ: quyền lực để ký tài liệu số), B sẽ kiểm tra xem A có quyền thực hiện các hành động nhất định không.
5. Kiểm Tra Thời Gian Hiệu Lực: B sẽ kiểm tra xem chứng chỉ có còn hiệu lực không. Nếu chứng chỉ đã hết hạn, B có thể quyết định không tin tưởng vào nó.
6. Quyết Định Tín Cậy: Dựa trên tất cả các kiểm tra trên, B sẽ quyết định xem có tin cậy vào chứng chỉ và danh tính của A hay không. Nếu tất cả các kiểm tra thành công, B có thể tin rằng chứng chỉ và thông tin về danh tính của A là hợp lệ.

**Bài tập 16 (ChatGPT).** Trong giao dịch điện tử SET, thông tin đơn hàng OI cần được gửi cho người bán. OI thường được ghép với chữ ký ghép và PIMD trước khi gửi đi. Ghép PIMD vào để làm gì, bỏ đi có được không?

**Bài giải.**

* Order Information (OI): Thông tin về đơn hàng bao gồm mô tả chi tiết sản phẩm, số lượng, giá trị, địa chỉ giao hàng, và các thông tin khác liên quan đến đơn hàng. OI giúp xác định các chi tiết quan trọng liên quan đến giao dịch.
* Payment Instructions Message Digest (PIMD): Là một giá trị băm (hash) của các thông tin thanh toán như số thẻ tín dụng, ngày hết hạn, và một số thông tin thanh toán khác. PIMD được sử dụng để đảm bảo tính toàn vẹn của thông tin thanh toán trong quá trình chuyển trực tuyến.

Khi ghép PIMD vào OI và ký chữ ký ghép (digital signature), nó giúp đảm bảo rằng thông tin thanh toán không bị thay đổi trong quá trình truyền tải. Chữ ký số và giá trị băm giúp xác định tính toàn vẹn và nguồn gốc của thông tin, cũng như đảm bảo rằng người nhận có thể xác nhận rằng thông tin đã được gửi bởi người gửi chính xác và không bị thay đổi.

Nếu bạn bỏ PIMD khỏi quá trình này, điều này có thể làm giảm đi tính toàn vẹn và an toàn của giao dịch. Mất PIMD có thể làm tăng nguy cơ về mối quan tâm về bảo mật trong quá trình thanh toán trực tuyến, nơi tính toàn vẹn của thông tin thanh toán là một yếu tố quan trọng. Do đó, thường thì không nên bỏ PIMD ra khỏi quá trình này để đảm bảo tính an toàn và toàn vẹn trong SET.

**Bài tập 17 chương 4.3 (Slide).** Các thành phần tham gia giao dịch điện tử & Các bước thực hiện giao dịch?

(\*) Các thành phần tham gia giao dịch điện tử: Người sở hữu thẻ, Bên phát hành thẻ, Người bán hàng, Cơ quan tài chính, Cổng thanh toán, Bên cấp chứng chỉ (CA)

(\*) Các bước thực hiện giao dịch

1. **Khách hàng mở tài khoản**. Khách hàng mở một tài khoản thẻ tín dụng, tại một ngân hàng có hỗ trợ thanh toán điện tử và SET.
2. **Khách hàng nhận được một chứng chỉ**. Sau khi xác minh nhận dạng thích hợp, khách hàng nhận được một chứng chỉ số X.509v3, được ngân hàng ký. Chứng chỉ này thiết lập một mối quan hệ giữa cặp khóa của khách hàng với thẻ tín dụng được cấp phát.
3. **Nhà bán hàng có các chứng chỉ của mình**. Một người bán hàng phải sở hữu hai chứng chỉ cho hai cặp khoá-công-khai của mình: một để ký các thông điệp và một để trao đổi khóa.
4. **Khách hàng đặt đơn hàng**. Khách hàng chọn các món hàng cần mua rồi gửi danh sách hàng hoá cho người bán. Người bán gửi lại khách một đơn hàng có chứa danh mục hàng hoá đã chọn, số lượng, đơn giá và tổng số tiền.
5. **Người bán được xác minh**. Cùng với đơn hàng, người bán hàng gửi một bản sao chứng chỉ, nhờ vậy khách hàng có thể xác minh được rằng mình đang giao dịch với một cửa hàng có thực và hợp lệ.
6. **Gửi đơn hàng và thông tin thanh toán**. Khách hàng gửi cả đơn hàng và các thông tin thanh toán cho người bán, cùng với chứng chỉ khách hàng của mình. Thông tin thanh toán (bao gồm chi tiết về thẻ tín dụng) được mã hóa sao cho người bán hàng không thể đọc được. Chứng chỉ khách hàng cho phép người bán xác minh khách hàng.
7. **Người bán yêu cầu cấp phép thanh toán**. Người bán gửi thông tin thanh toán cho cổng an ninh thanh toán, yêu cầu xác nhận chủ thẻ có khả năng thanh toán đơn hàng.
8. **Người bán xác nhận đơn hàng**. Người bán gửi xác nhận đơn hàng cho khách hàng.
9. **Người bán cung cấp hàng hóa hoặc dịch vụ**. Người bán chuyên chở hàng hóa hoặc áp dụng dịch vụ cho khách hàng.
10. **Người bán yêu cầu thanh toán**. Yêu cầu này được gửi tới cổng an ninh thanh toán là nơi giải quyết tất cả các thủ tục thanh toán

**Bài tập 18 chương 4.** Chữ ký kép OI và PI?

**Bài tập 19.** Khi mua bán thì người bán gửi cho người mua những thông tin gì (Liên quan đến OI và PI)?

**Bài giải.**

Thông Tin Đặt Hàng (Order Information - OI):

* Số đơn hàng (Order Number): Để theo dõi và xác nhận đơn hàng.
* Sản phẩm và Dịch vụ: Mô tả chi tiết về sản phẩm hoặc dịch vụ mà người mua đang mua.
* Số lượng và Giá trị đơn hàng: Để xác định số lượng sản phẩm và giá trị cả đơn hàng.
* Phí vận chuyển và Thuế: Thông tin về chi phí vận chuyển và thuế, nếu có.
* Địa chỉ giao hàng và Thông tin liên lạc: Để xác định nơi gửi hàng đến và liên lạc với người mua.

Thông Tin Cá Nhân (Personal Information - PI):

* Tên và Địa chỉ của Người Mua: Để xác định người nhận hàng và thông tin liên lạc.
* Phương thức thanh toán: Thông tin về phương thức thanh toán được sử dụng, ví dụ như số thẻ tín dụng hoặc thông tin tài khoản ngân hàng.
* Thông tin Liên lạc: Địa chỉ email, số điện thoại để người bán có thể liên lạc với người mua nếu cần thiết.
* Thông tin xác thực: Trong trường hợp cần, thông tin để xác minh danh tính người mua, ví dụ như chứng minh nhân dân hoặc giấy phép lái xe.

**Bài tập 20 (ChatGPT).** Nêu các bước kỹ thuật brute-force

Brute-force là một phương pháp tấn công mật khẩu bằng cách thử nghiệm tất cả các khả năng có thể của mật khẩu cho đến khi tìm ra mật khẩu đúng. Dưới đây là các bước cơ bản của kỹ thuật brute-force:

1. Xác định mục tiêu: Xác định mục tiêu của tấn công brute-force, chẳng hạn như tài khoản người dùng, mật khẩu hoặc khóa mã hóa.
2. Xác định không gian khóa: Xác định không gian khóa, tức là tất cả các khả năng có thể của mật khẩu hoặc khóa mã hóa.
3. Chuẩn bị công cụ tấn công: Sử dụng phần mềm hoặc kịch bản tấn công để tự động thực hiện việc thử nghiệm mật khẩu.
4. Chọn chiến lược thử nghiệm: Chọn chiến lược thử nghiệm mật khẩu, có thể bao gồm thử tất cả các ký tự có thể, thử theo từ điển, hoặc thử theo mô hình có thể dự đoán được.
5. Thực hiện tấn công: Thực hiện tấn công brute-force bằng cách thử nghiệm mật khẩu từ không gian khóa đã xác định. Cứ mỗi mật khẩu được thử, hệ thống sẽ kiểm tra xem mật khẩu đó có đúng không.
6. Ghi lại kết quả: Ghi lại tất cả các mật khẩu đã thử và kết quả (đúng hay sai).
7. Kiểm tra kết quả: Kiểm tra kết quả để xem liệu tấn công đã thành công hay chưa. Nếu tìm thấy mật khẩu đúng, tấn công thành công.
8. Tối ưu hóa và điều chỉnh: Tùy thuộc vào kết quả, tối ưu hóa và điều chỉnh chiến lược thử nghiệm để làm cho tấn công hiệu quả hơn.

Lưu ý rằng brute-force thường là một phương pháp chậm và đòi hỏi nhiều tài nguyên tính toán. Các biện pháp bảo mật như việc sử dụng mật khẩu mạnh, khóa kép, và giới hạn số lần thử mật khẩu được áp dụng để ngăn chặn loại tấn công này.

**Bài tập 21 chương 5 (docs).** Các kỹ thuật lấy mật khẩu, xâm nhập?

Kỹ thuật lấy mật khẩu

1. Thử các mật khẩu từng dùng với các tài khoản chuẩn gắn liền với hệ thống.
2. Thử hết tất cả các mật khẩu ngắn
3. Thử các từ trong từ điển trực tuyến của hệ thống, hoặc danh sách mà rất có khả năng được làm mật khẩu
4. Thu thập thông tin người dùng như họ tên đầy đủ, tên vợ, chồng, con cái, …
5. Thử tất cả số điện thoại người dùng, các số an sinh xã hội, số nhà, phòng làm việc
6. Thử tất cả các biển xe hợp lý của địa phương
7. Dùng trojan horse để ăn cắp mật khẩu
8. Đấu nối một đường truyền giữa người dùng từ xa với host hệ thống.

Kỹ thuật xâm nhập

1. Tìm cách lấy được mật khẩu người dùng
2. Khai thác các lỗ hổng an ninh của hệ thống (ví dụ lỗi tràn bộ đệm..)
3. Mua chuộc, dụ dỗ…

**Bài tập 22 (ChatGPT).** Phân biệt 2 chuẫn mã hóa dữ liệu AES và DES.

**Bài giải.**

Mật mã AES (Advanced Encryption Standard) và DES (Data Encryption Standard) là hai thuật toán mã hóa khác nhau được sử dụng để bảo vệ thông tin và dữ liệu. Dưới đây là một số điểm khác biệt giữa chúng:

1. Chiều dài khóa:
   1. AES: Hỗ trợ các khóa có độ dài 128, 192, hoặc 256 bit.
   2. DES: Sử dụng khóa 56 bit.
2. Độ an toàn:
   1. AES: Được coi là mạnh và an toàn, được sử dụng rộng rãi trong nhiều ứng dụng bảo mật.
   2. DES: Đã bị coi là không an toàn do độ dài khóa ngắn và đã bị thay thế bởi các tiêu chuẩn mã hóa mạnh mẽ hơn.
3. Hiệu suất:
   1. AES: Thường có hiệu suất cao hơn so với DES, đặc biệt là với các khối dữ liệu lớn.
   2. DES: Có thể trở nên chậm đối với các tập tin hoặc dữ liệu lớn.
4. Kiểu thuật toán:
   1. AES: Sử dụng thuật toán thay thế và hoán vị để mã hóa dữ liệu.
   2. DES: Sử dụng mô hình Substitution-Permutation Network (SPN).
5. Tiêu chuẩn:
   1. AES: Được chấp nhận rộng rãi và được sử dụng trong nhiều ứng dụng và giao thức bảo mật.
   2. DES: Đã bị thay thế bởi các tiêu chuẩn mới hơn như AES và 3DES.

Tóm lại, AES thường được ưa chuộng hơn DES trong các ứng dụng hiện đại do khả năng an toàn và hiệu suất cao hơn.

#### **Bài tập 23.** Firewall

# Bài tập câu 3.

**Bài tập 1 (2202).** Mã hóa bằng phép XOR

Lập một chuỗi ký tự plaintext từ bàn phím. Nhập khóa K (16 bit) từ bàn phím. Chia chuỗi thành từng khối 16 bit (nếu không chia hết cho 16 bit thì thêm kí tự quy ước).

Mã hóa chuỗi ban đầu bằng cách xor từng khối của chuỗi với K. Hiện chuỗi sau khi mã hóa màn hình. Viết chương trình giải mã.

A screenshot of a computer code

Description automatically generated

**Bài tập 2 (2205).** Mã hóa bằng công thức C = 7\*P + K

Lập một chuỗi ký tự plaintext từ bàn phím (Các ký tự nằm trong khoảng từ ‘A’ tới ‘Z’). Nhập số nguyên K thuộc Z26 từ bàn phím. Mã hóa chuỗi ban đầu theo công thức C = 7\*P + K. Các phép tính được thực hiện trong Z26.

Hiện chuỗi sau khi mã hóa màn hình. Viết chương trình giải mã.

A screenshot of a computer program

Description automatically generated

**Bài tập 3 (1805).** Mã hóa bằng công thức C = P/a + b

Lập một chuỗi ký tự plaintext từ bàn phím (Các ký tự nằm trong khoảng từ ‘A’ tới ‘Z’). Nhập số nguyên a, b thuộc Z26 từ bàn phím. Mã hóa chuỗi ban đầu theo công thức C = P/a + b. Các phép tính được thực hiện trong Z26.

Hiện chuỗi sau khi mã hóa màn hình. Viết chương trình giải mã.

A computer code with numbers and symbols

Description automatically generated

A screenshot of a computer code

Description automatically generated

A computer code with black text

Description automatically generated

A computer code with black text

Description automatically generated**}**

**Bài tập 4.** Mã hóa bằng phép cộng (Giống Caesar?)

Lập trình nhập một chuỗi ký tự từ bàn phím, cộng mỗi phần tử của chuỗi với K (K là một số nguyên nhập từ bàn phím). Hiện chuỗi mới ra màn hình. Lập trình khôi phục lại chuỗi ban đầu.

A screenshot of a computer program

Description automatically generated

**Bài tập 5.** Mã hóa Caesar tổng quát

Lập trình nhập một chuỗi ký tự từ bàn phím, mã hóa chuỗi bằng thuật toán Caesar tổng quát với khóa K nhập từ bàn phím. Hiện chuỗi mới. Lập trình giải mã để khôi phục lại chuỗi ban đầu.

**Bài tập 6.** Mã hóa Caesar, giải mã Brute-force

Lập trình bẻ khóa mật mã Caesar bằng phương pháp Brute-force. Đầu vào chương trình là chuỗi ký tự cipher text thu được từ bài trên.

A computer screen shot of text

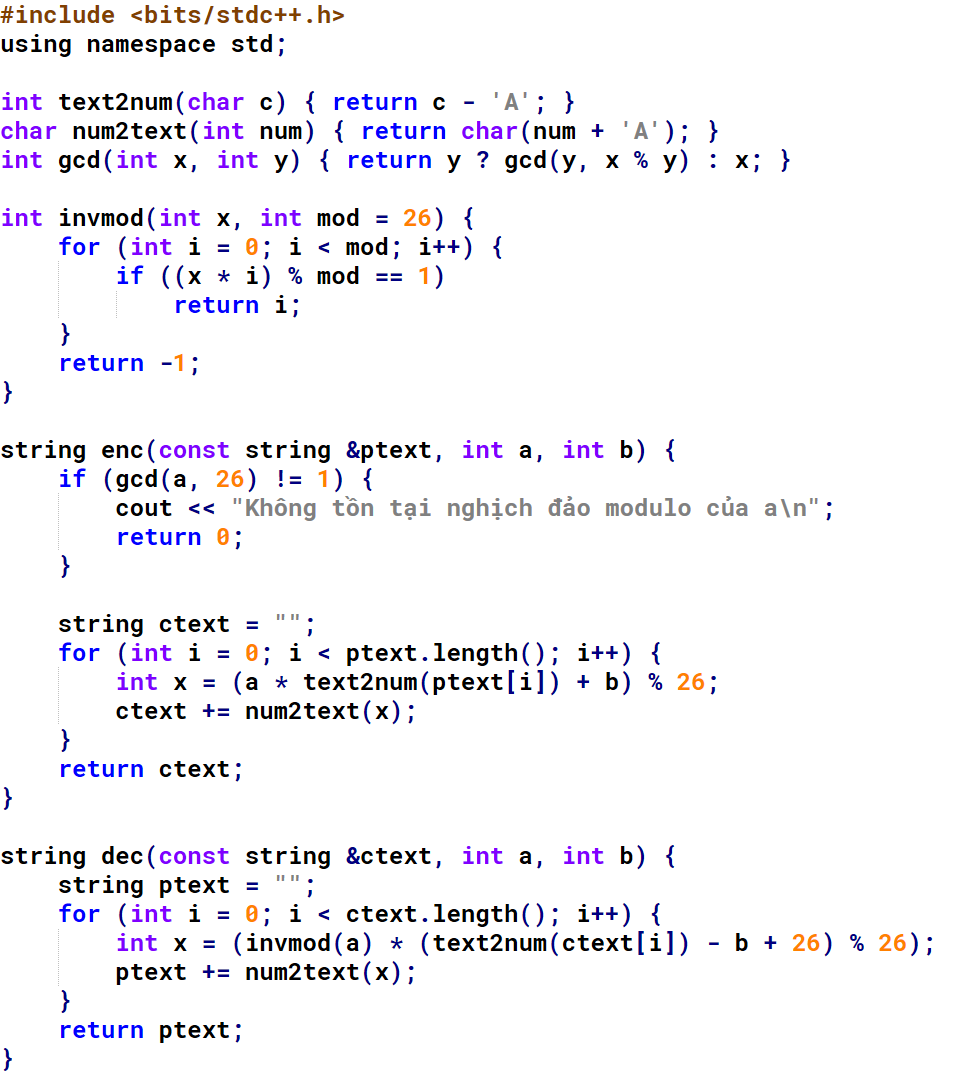
Description automatically generated

**Bài tập 7.** Mã hóa Affine

Lập trình nhập một chuỗi ký tự từ bàn phím, mã hóa chuỗi bằng thuật toán Affine với cặp số {a, b} nhập từ bàn phím. Hiện chuỗi mới ra màn hình.

Lập trình giải mã Affine để khôi phục lại chuỗi ban đầu:

* Đầu vào chương trình là cặp số {a, b} và chuỗi ký tự cipher text từ câu trên.
* Đầu ra chương trình là chuỗi ký tự plain text.



A screenshot of a computer code

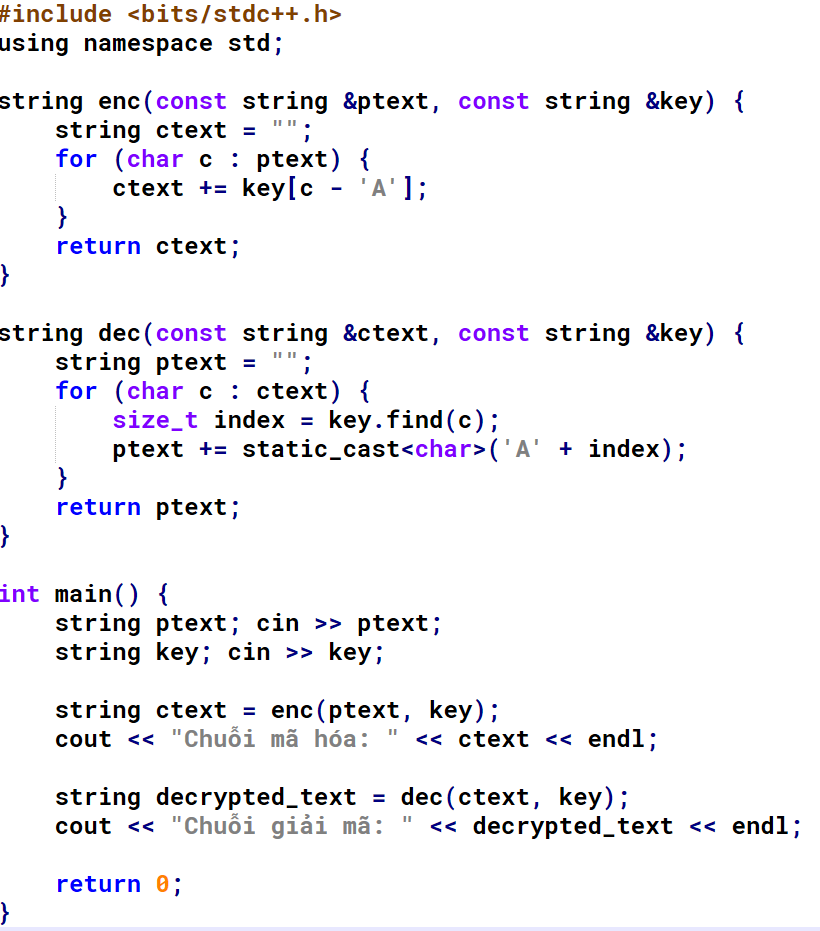
Description automatically generated

**Bài tập 8.** Mã hóa Monoalphabetic

**Phần 1.** Lập trình nhập một chuỗi ký tự từ bàn phím, mã hóa chuỗi bằng thuật toán Monoalphabetic với khóa K nhập từ bàn phím (Khóa K là một chuỗi gồm 26 chữ cái có trật tự bất kỳ). Hiện chuỗi mới ra màn hình.

**Phần 2.** Lập trình giải mã Monoalphabetic để khôi phục lại chuỗi ban đầu:

* Đầu vào chương trình là khóa K và chuỗi ký tự cipher text từ phần 1.
* Đầu ra chương trình là chuỗi ký tự plain text



**Bài tập 9.** Mã hóa Playfair

**Phần 1.** Mã hóa tay

Chọn một từ khóa bất kỳ rồi xây dựng ma trận khóa Playfair. Chọn một plaintext bất kỳ, áp dụng ma trận khóa trên để tạo ra ciphertext.

Thử giải mã ciphertext rồi so sánh với plaintext ban đầu.

Ví dụ. PT = DAIHOCTHUYLOI, K = CONGNGHE

**Bài giải.**

**Ma trận khóa K**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| C | O | N | G | H |
| E | A | B | D | F |
| I/J | K | L | M | P |
| Q | R | S | T | U |
| V | W | X | Y | Z |

P = DAIHOCTHUYLOI

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P | DA | IH | OC | TH | UY | LO | IX |
| C | FB | PC | NO | UG | TZ | KN | LV |

**Phần 2.** Code

Chọn một từ khóa bất kỳ rồi xây dựng ma trận khóa Playfair. Chọn một plaintext bất kỳ, áp dụng ma trận khóa trên để tạo ra ciphertext.

Thử giải mã ciphertext rồi so sánh với plaintext ban đầu

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

A computer code with black and white text

Description automatically generated

**Bài tập 10.** Mật mã Hill

Với plaintext là “paymoremoney” và sử dụng với mật khóa K nói trên, hãy xác định ciphertext. Giải mã ciphertext thu đươc bằng cách nhân nó với inv(K), rồi so sánh kết quả với plaintext ban đầu.





Mã hóa

A table with numbers and symbols

Description automatically generated

NVPKCTLFUTTL

Giải mã

A black background with numbers and equations

Description automatically generated

PAYMOREMONEY

**Bài tập 13.** Mã hóa Rain-Fence

Nhập một chuỗi plaintext. Mã hóa chuỗi bằng kỹ thuật Rain-Fence, hiện ciphertext ra màn hình. Giải mã ciphertext.

**Hướng giải.** Plaintext được viết dịch xuống tuần tự theo các đường chéo rồi đọc trình tự theo các hàng.

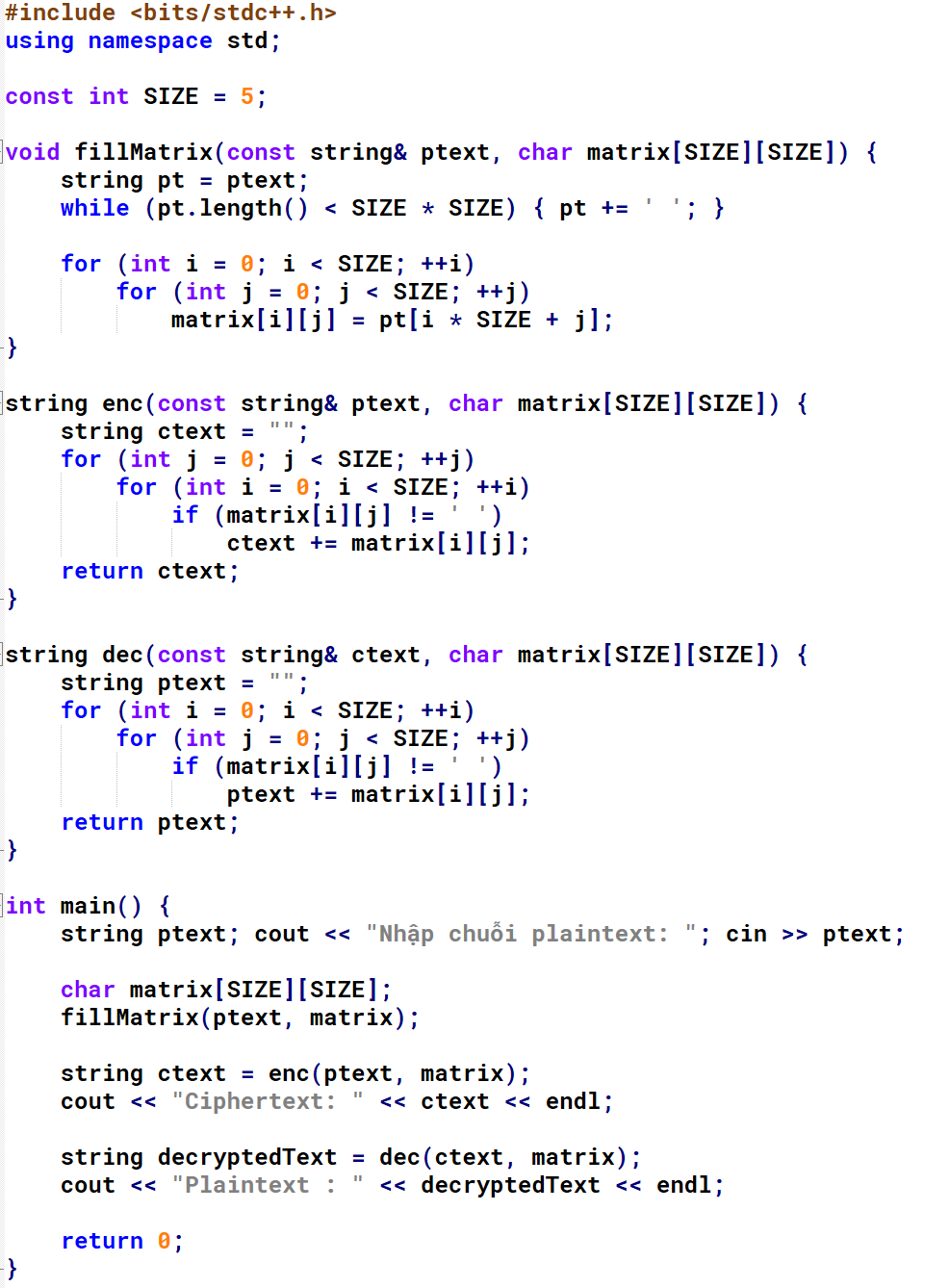
A screenshot of a computer code

Description automatically generated

**Bài tập 14.** Code đổi hàng cột

Nhập một chuỗi plaintext(dài không quá 25 ký tự). Sắp xếp các ký tự của chuỗi vào một ma trận 5x5(lần lượt theo từng hàng). Đọc các phần tử của ma trận theo từng cột để tạo thành ciphertext.

Giải mã ciphertext, so sánh kết quả với chuỗi ban đầu.



**Bài tập 15.** Mật mã Feistel

Lập trình mô phỏng hoạt động của mật mã Feistel đơn giản với 2 vòng xử lý:

* Nhập một khối plaintext từ bàn phím, khối có độ dài m = 2w = 16 bít.
* Nhập khóa K(dài 8 bít) từ bàn phím. Khóa K\_i được sinh ra từ khóa K nhờ phép dịch trái K\_i lần.

Hàm F thực hiện phép cộng giữa R\_{i-1} với K\_i. Hiện khối ciphertext ra màn hình.

A computer code on a white background

Description automatically generated

**Bài tập 16.** Mật mã RSA

* Chọn một cặp số nguyên tố p, q
* Thực hiện các bước tạo khóa, mã hóa và giải mã theo thuật toán RSA

**Bài giải.**

Công thức mã hoá: *C* = *Pa* mod *n* (1)

Công thức giải mã: *P* = *Cb* mod *n* (2)

Trong đó:

* *n = p.q* là tích của hai số nguyên tố lớn
* *P, C* thuộc *Zn*
* *a* và *b* là các số nguyên thoả mãn điều kiện: *Pab* mod *n = P*
* Có thể đạt được điều đó nếu: *ab mod Φ = 1 (\*),* với *Φ = (p* − *1)(q* − *1)*

Điều đó chỉ đúng nếu *a* (và *b)* có quan hệ nguyên tố với *Φ*: *ƯSCLN(Φ, a) = 1*

Công thức *(\*)* tương đương với: *b = a-1 mod Φ*

Tạo khóa

* Chọn 2 số nguyên tố lớn *p*, *q*
* Tính *n* = *pq*
* Tính *Φ = (p − 1)(q − 1)*
* Chọn *a* sao cho *ƯSCLN(Φ,a) = 1*; *a* < *Φ*
* Tính *b = a-1 mod Φ*

Để mã hoá cần phải biết *a* và *n*. Khoá công khai chính là *K1 = (a, n)*

Để giải mã cần biết *b* và *n*. Khoá bí mật là *K2 = (b, n)*

**Bài tập 17.** Mật mã Vigenere tay to + Code

Cho chuỗi Plaintext sau : **KHONGBENHLATIENSUONGTUYETTRAN**

Hãy mã hóa chuỗi nói trên bằng mật mã Vigenere, với khóa K là họ tên của sinh viên, m là độ dài họ tên.

Giải mã ciphertext thu được và so sánh plaintext ban đầu.

**Bài giải.**

**(\*) Phần mã hóa**

Khóa K: **HOANGTHIMAI** 🡪 m = 11

Plaintext: **KHONGBENHLATIENSUONGTUYETTRAN**

Do m = 11, ta sẽ tách thành plaintext thành từng nhóm 11 ký tự:

**KHONGBENHLA** / **TIENSUONGTU** / **YETTRAN**

Viết theo dạng số là:

10 7 14 13 6 1 4 13 7 11 0 / 19 8 4 13 18 20 14 13 6 19 20 / 24 4 19 19 17 0 13

Từ khóa HOANGTHIMAI tương ứng với K = (7, 14, 0, 13, 6, 19, 7, 8, 12, 0, 8)

Cộng từng nhóm 11 ký tự của plaintext với K ta có:

10 7 14 13 6 1 4 13 7 11 0 / 19 8 4 13 18 20 14 13 6 19 20 / 24 4 19 19 17 0 13

7 14 0 13 6 19 7 8 12 0 8 / 7 14 0 13 6 19 7 8 12 0 8 / 7 14 0 13 6 19 7

17 21 14 0 12 20 11 21 19 11 8 / 0 22 4 0 24 13 21 21 18 19 2 / 5 18 19 6 23 19 20

* Ciphertext tương ứng là: RVOAMULVTLI / AWEAYNVVSTC / FSTGXTU

**(\*) Phần giải mã**

Khóa K: HOANGTHIMAI 🡪 m =11

Ciphertext: RVOAMULVTLI / AWEAYNVVSTC / FSTGXTU

Do m = 11, ta sẽ tách ciphertext thành từng nhóm 11 ký tự:

RVOAMULVTLI / AWEAYNVVSTC / FSTGXTU

Viết theo dạng số là:

17 21 14 0 12 20 11 21 19 11 8 / 0 22 4 0 24 13 21 21 18 19 2 / 5 18 19 6 23 19 20

Trừ từng nhóm 16 ký tự của Ciphertext với K ta có:

17 21 14 0 12 20 11 21 19 11 8 / 0 22 4 0 24 13 21 21 18 19 2 / 5 18 19 6 23 19 20

7 14 0 13 6 19 7 8 12 0 8 / 7 14 0 13 6 19 7 8 12 0 8 / 7 14 0 13 6 19 7

10 7 14 13 6 1 4 13 7 11 0 / 19 8 4 13 18 20 14 13 6 19 20 / 24 4 19 19 17 0 13

* Plaintext : KHONGBENHLATIENSUONGTUYETTRAN

**(\*) Code**

A screenshot of a computer program

Description automatically generated

**Bài tập 18.** Mật mã Vernman tay to + Code

Giả sử plaintext là “**TIEN VE HA NOI 1945**” (các ký tự trong chuỗi có thể được biểu diễn dưới dạng nhị phân bảng mã chuẩn ASCII), khóa K là họ và tên sinh viên dưới dạng mã nhị phân. Hãy mã hóa chuỗi ban đầu bằng phương pháp Vernman.

Giải mã chuỗi thu được rồi so sánh với chuỗi ban đầu.

**Phương pháp giải.** C = P xor K

**Bài giải.**

Khóa K = “HOANG THI MAI” = 48 4F 41 4E 47 20 54 48 49 20 4D 41 49

= 0100 1000 0100 1111 0100 0001 0100 1110 0100 0111 0010 0000 0101 0100 0100 1000 0100 1001 0010 0000 0100 1101 0100 0001 0100 1001

PT = “TIEN VE HA NOI 1945” = 54 49 45 4E 20 56 45 20 48 41 20 4E 4F 49 20 31 39 34 35

= 0101 0100 0100 1001 0100 0101 0100 1110 0010 0000 0101 0110 0100 0101 0010 0000 0100 1000 0100 0001 0010 0000 0100 1110 0100 1111 0100 1001 0011 0001 0011 1001 0011 0100 0011 0101

**(\*) Phần mã hóa**

Plaintext = 0101 0100 0100 1001 0100 0101 0100 1110 0010 0000 0101 0110 0100 0101 0010 0000 0100 1000 0100 0001 0010 0000 0100 1110 0100 1111 0100 1001 0010 0000 0011 0001 0011 1001 0011 0100 0011 0101

Khóa K = 0100 1000  0100 1111 0100 0001 0100 1110 0100 0111 0010 0000 0101 0100

0100 1000 0100 1001 0010 0000 0100 1101 0100 0001 0100 1001

Ciphertext = 0001 1100 0000  0110 0000 0100 0000 0000 0110 0111 0111 0110 0001 0001 0110 1000 0000 0001 0110 0001 0110 1101 0000 1111 0000 0110 0000 0001  0110 1111 0111  0000 0111  0111 0111 0011 0001 0101

**(\*) Phần giải mã**

Ciphertext = 0001 1100 0000  0110 0000 0100 0000 0000 0110 0111 0111 0110 0001 0001 0110 1000 0000 0001 0110 0001 0110 1101 0000 1111 0000 0110 0000 0001  0110 1111 0111  0000 0111  0111 0111 0011 0001 0101

Khóa K = 0100 1000  0100 1111 0100 0001 0100 1110 0100 0111 0010 0000 0101 0100

0100 1000 0100 1001 0010 0000 0100 1101 0100 0001 0100 1001

Plaintext = 0101 0100 0100 1001 0100 0101 0100 1110 0010 0000 0101 0110 0100 0101

0010 0000 0100 1000 0100 0001 0010 0000 0100 1110 0100 1111 0100 1001 0010 0000 0011 0001 0011 1001 0011 0100 0011 0101

**(\*) Code**

**A computer screen shot of a code

Description automatically generated**

**Bài tập 19.** Mật mã Affine (mã hóa bằng tay)

Cho chuỗi plaintext sau “**KHOACONGNGHETHONGTIN**”

Hãy mã hóa chuỗi nói trên bằng mật mã Affine, với khóa K là cặp số {a,b} lấy từ 2 số cuối của mã sinh viên. Trong đó, a là số hàng chục (nếu a chẵn thì cộng thêm 1), b là số hàng đơn vị.

Giải mã ciphertext thu được và so sánh với plaintext ban đầu.

**Bài giải.**

K = {a,b} = {3, 9},  inv(a)= 9

P = KHOACONGNGHETHONGTIN

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P\_i | K | H | O | A | C | O | N | G | N | G | H | E | T | H | O | N | G | T | I | N |
|  | 10 | 7 | 14 | 0 | 2 | 14 | 13 | 6 | 13 | 6 | 7 | 4 | 19 | 7 | 14 | 13 | 6 | 19 | 8 | 13 |

**Công thức mã hóa**: C\_i = E(P\_i, {a,b}) = (aP\_i + b) mod 26 = (3P\_i + 9) mod 26

Áp dụng công thức trên ta có

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ci | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 | C12 | C13 | C14 | C15 | C16 | C17 | C18 | C19 | C20 |
|  | 13 | 4 | 25 | 9 | 15 | 25 | 22 | 1 | 22 | 1 | 4 | 21 | 14 | 4 | 25 | 22 | 1 | 14 | 7 | 22 |
|  | N | E | Z | J | P | Z | W | B | W | B | E | V | O | E | Z | W | B | O | H | W |

**Công thức giải mã:** P\_i = D(C\_i, {a,b}) = inv(a)(C\_i – b) mod 26 = 9(C\_i – 9) mod 26

Áp dụng công thức trên ta có

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ci | N | E | Z | J | P | Z | W | B | W | B | E | V | O | E | Z | W | B | O | H | W |
| Pi | 10 | 7 | 14 | 0 | 2 | 14 | 13 | 6 | 13 | 6 | 7 | 4 | 19 | 7 | 14 | 13 | 6 | 19 | 8 | 13 |
|  | K | H | O | A | C | O | N | G | N | G | H | E | T | H | O | N | G | T | I | N |

* KL: Giải mã được kết quả trùng với plaintext ban đầu

**Bài tập 20**. Tính mã Hash 1

Nhập một chuỗi ký tự từ bàn phím, chia chuỗi thành từng khối 64 bit (ứng với 8 ký tự) rồi tính mã hash của chuỗi theo phương pháp:

* Giả sử thông điệp đầu vào được chia thành N khối, mối khối dài m bít
* Mã hash dài m bít được tính bằng cách XOR các bít tương ứng của tất cả các khối: H\_i = b\_i1 XOR b\_i2 XOR ... XOR b\_iN. Trong đó: H\_i là bit thứ i của mã hash, 1 <= i <= m, b\_ij là bit thứ i của khối thứ j.

(Nếu độ dài chuỗi không phải là bội số của 8 thì có thể chèn thêm các ký tự quy ước)

**Bài giải.**

**A computer screen shot of a program

Description automatically generated**

**Bài tập 21.** Tính mã Hash 2

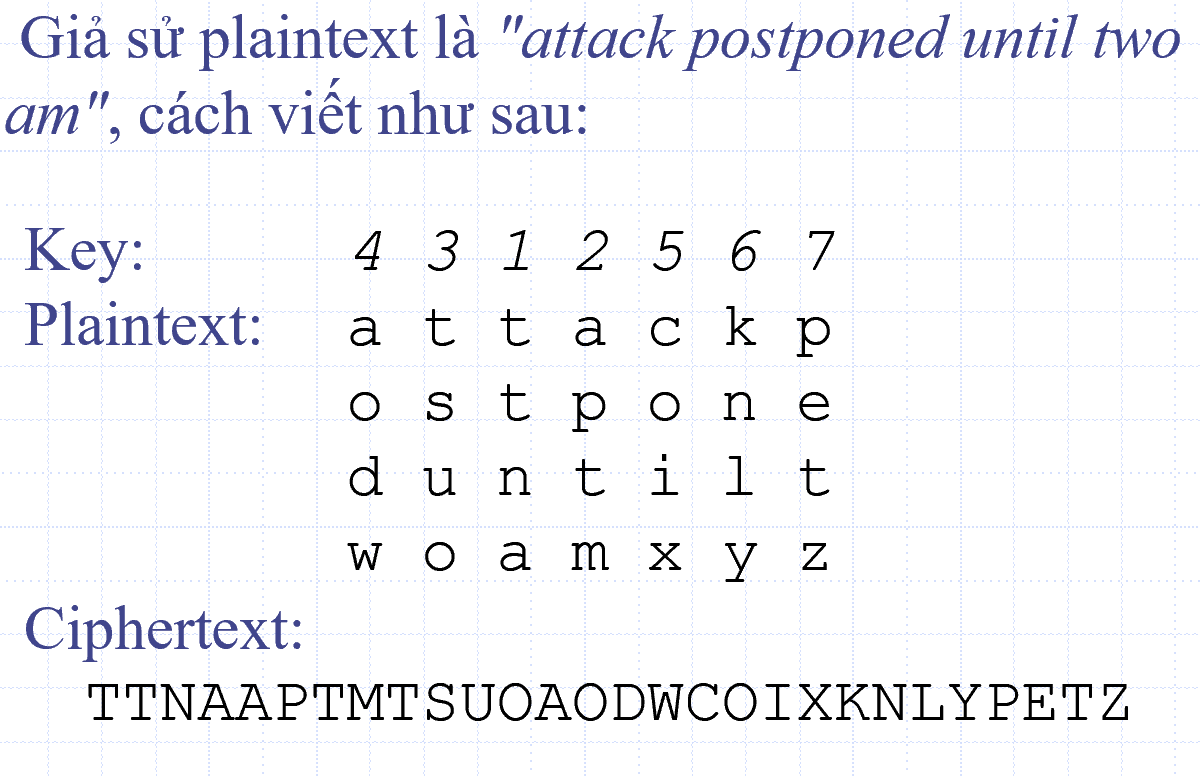
Giả sử thông điệp đầu vào được chia thành N khối, mỗi khối dài m bít. Để cải thiện tính ngẫu nhiên của đầu vào so với bài trên, ta sẽ thực hiện phép quay phải trước khi tiến hành XOR:

* Giá trị hash ban đầu được đặt bằng 0
* Thực hiện N vòng lặp (i từ 0 → N-1), mỗi vòng làm các công việc sau:
  + Quay phải một bít giá trị hash hiện tại
  + XOR giá trị hash với khối dữ liệu thứ i

A computer screen shot of text

Description automatically generated

**Bài tập 22.** Mã hóa Rain-Fence hoán vị cột



A screenshot of a computer program

Description automatically generated

A computer code with text

Description automatically generated