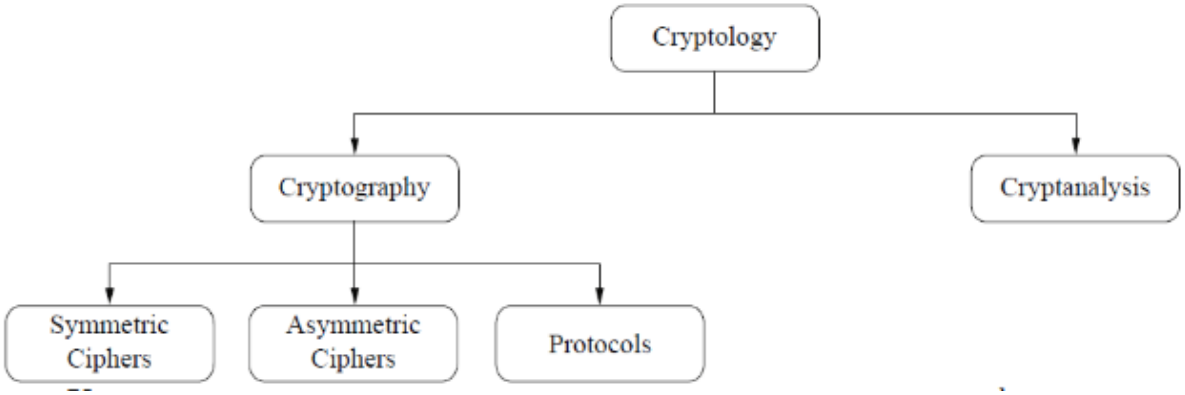
Bài giảng 1
Lịch sử mã hóa. Phân loại hệ thống mật mã.

Chủ đề bài giảng

1. Hệ mật mã đối xứng. Mã hóa lịch sử. Mã dịch chuyển, mã thay thế và hoán vị.
2. Mã khối. Thuật toán DES.
3. Mã khối AES (Rijndael).
4. Mã khối. Tiêu chuẩn GOST 28147-89.
5. Chế độ hoạt động của các thuật toán đối xứng. Các chế độ ECB, CBC, OFB, CFB.
6. Mã dòng, thanh ghi dịch với phản hồi tuyến tính (LFSR).
7. Phân phối khóa cho các thuật toán đối xứng. Các giao thức Ếch miệng rộng, Needham-Schroeder, Otway-Rees, Kerberos.
8. Số học đồng dư, thuật toán Euclid mở rộng, nghịch lý ngày sinh. Ứng dụng trong mật mã.
9. Mật mã khóa công khai. Các bài toán toán học làm nền tảng.
10. Mật mã khóa công khai. Thuật toán RSA.
11. Phân phối khóa bằng mật mã khóa công khai. Thuật toán Diffie-Hellman.
12. Chữ ký số. Ứng dụng mật mã khóa công khai để tạo chữ ký số. Chữ ký dựa trên RSA.
13. Hàm băm. MD5, SHA-1.
14. Phân tích mật mã. Tấn công kênh bên.

Phân loại hệ mật mã



Hình ảnh mô tả **phân loại hệ mật mã (классификация криптосистем)** với sơ đồ cấu trúc như sau:

1. **Cryptology (Mật mã học)**
    - Là khoa học nghiên cứu về **phương pháp mã hóa và giải mã**.
    - Bao gồm hai nhánh chính:
        - **Cryptography (Mật mã học - Криптография)**

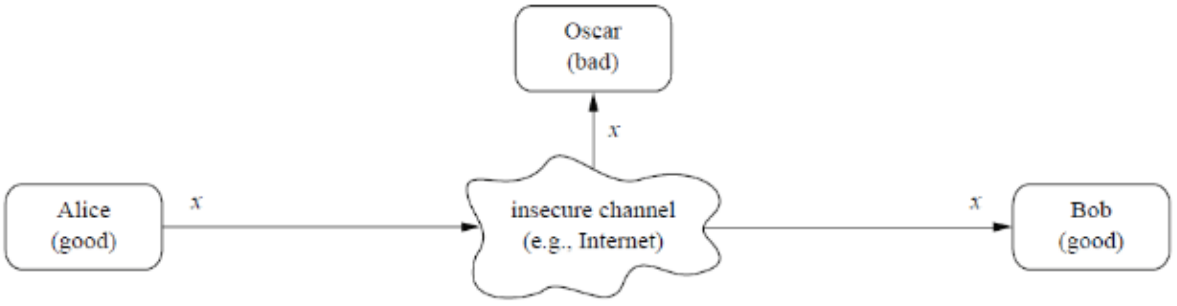
        - Nghiên cứu và phát triển các phương pháp **mã hóa (encryption) và giải mã (decryption)** dữ liệu.
        - Được chia thành ba nhóm chính:
            - **Symmetric Ciphers (Mã hóa đối xứng)** – Sử dụng cùng một khóa để mã hóa và giải mã (ví dụ: AES, DES).
            - **Asymmetric Ciphers (Mã hóa bất đối xứng)** – Sử dụng một cặp khóa (công khai và bí mật), ví dụ: RSA, Diffie-Hellman.
            - **Protocols (Giao thức mã hóa)** – Các giao thức bảo mật giúp trao đổi khóa và đảm bảo tính toàn vẹn của dữ liệu (ví dụ: SSL/TLS).
    - **Cryptanalysis (Phân tích mật mã - Криптоанализ)**
        - Tập trung vào việc **phá mã, đánh giá điểm mạnh và yếu của các thuật toán mã hóa**.
        - Dùng để tìm ra lỗ hổng trong các hệ thống mật mã hoặc kiểm tra độ an toàn của thuật toán.

Kết luận

- **Cryptology** là lĩnh vực rộng bao gồm cả **Cryptography (mật mã học)** và **Cryptanalysis (phân tích mật mã)**.
- **Cryptography** tạo ra các hệ thống mã hóa để bảo mật dữ liệu.
- **Cryptanalysis** kiểm tra và tìm cách phá mã để đánh giá tính an toàn của các hệ thống đó.
- Cả hai đều quan trọng trong việc đảm bảo bảo mật thông tin trong nhiều lĩnh vực như truyền thông, tài chính và quân sự.

"Взаимодействие по незащищенному каналу"

(Tương tác qua kênh không an toàn)



1. Nội dung sơ đồ

Sơ đồ mô tả việc trao đổi thông tin qua một **kênh không an toàn** (ví dụ: Internet), trong đó có nguy cơ bị nghe lén hoặc tấn công.

- **Alice (good)**: Là người gửi thông tin.
- **Bob (good)**: Là người nhận thông tin.
- **Insecure channel (kênh không an toàn)**: Đại diện cho mạng truyền thông có rủi ro (ví dụ: Internet).
- **Oscar (bad)**: Là kẻ tấn công có thể nghe lén hoặc can thiệp vào dữ liệu được truyền qua kênh không an toàn.

2. Ý nghĩa của sơ đồ

- Khi **Alice gửi thông tin (x)** cho **Bob**, dữ liệu phải đi qua một **kênh không an toàn**.
- **Oscar (kẻ xấu)** có thể **chặn, nghe lén hoặc sửa đổi dữ liệu** trong quá trình truyền.
- Đây là một vấn đề quan trọng trong bảo mật thông tin và là lý do cần sử dụng **các kỹ thuật mã hóa** để bảo vệ dữ liệu.

3. Giải pháp bảo mật

Để bảo vệ thông tin khi truyền qua kênh không an toàn, có thể áp dụng các phương pháp sau:

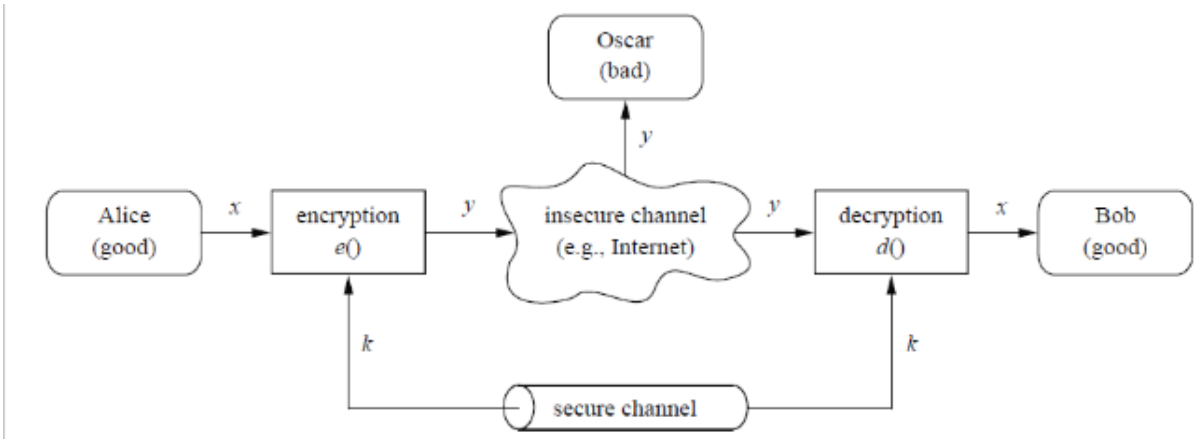
- **Mã hóa (Encryption)**: Dữ liệu được mã hóa trước khi gửi đi (ví dụ: AES, RSA).
- **Chữ ký số (Digital Signature)**: Xác minh tính toàn vẹn và xác thực nguồn gốc dữ liệu.
- **Giao thức bảo mật (Secure Protocols)**: Sử dụng HTTPS, TLS/SSL để bảo vệ dữ liệu truyền qua Internet.

4. Kết luận

Sơ đồ minh họa vấn đề quan trọng trong bảo mật thông tin – **nguy cơ nghe lén và tấn công trên kênh truyền không an toàn**. Để bảo vệ dữ liệu, cần sử dụng **các phương pháp mã hóa và xác thực** trong truyền thông mạng.

"Симметричный шифр как защита от перехвата"

(Mã hóa đối xứng như một biện pháp bảo vệ chống chặn dữ liệu)



1. Mô tả sơ đồ

Sơ đồ minh họa cách **mã hóa đối xứng** giúp bảo vệ dữ liệu khi truyền qua một kênh không an toàn.

- **Alice (người gửi, tốt)**: Gửi dữ liệu **x** (bản rõ – plaintext).
- **Encryption (e())**: Mã hóa **x** thành **y** (bản mã – ciphertext) bằng khóa **k**.
- **Kênh không an toàn (insecure channel)**: Dữ liệu bị gửi qua Internet, nơi **Oscar (kẻ xấu)** có thể chặn được bản mã **y**.
- **Decryption (d())**: Bob (người nhận) sử dụng cùng khóa **k** để giải mã **y** và khôi phục lại **x**.
- **Kênh an toàn (secure channel)**: Dùng để truyền **khóa k** giữa Alice và Bob trước khi truyền dữ liệu chính.

2. Giải thích các khái niệm

- **x** – **Plaintext** (Bản rõ): Dữ liệu gốc trước khi mã hóa.
- **y** – **Ciphertext** (Bản mã): Dữ liệu đã được mã hóa, không thể đọc nếu không có khóa.
- **k** – **Khóa mật mã**: Khóa dùng để mã hóa và giải mã.
- **Không gian khóa**: Tập hợp tất cả các khóa có thể có.

3. Cách bảo vệ chống chặn dữ liệu

- **Mã hóa đối xứng** bảo vệ nội dung tin nhắn ngay cả khi kẻ tấn công **Oscar** chặn được bản mã **y**, vì **không có khóa k**, hắn không thể giải mã.
- Tuy nhiên, **khóa k phải được chia sẻ an toàn** giữa Alice và Bob thông qua một **kênh bảo mật** (secure channel).

4. Hạn chế của mã hóa đối xứng

- **Cần một kênh an toàn để trao đổi khóa k** trước khi liên lạc.
- Nếu kẻ tấn công lấy được **k**, toàn bộ hệ thống sẽ bị xâm phạm.

5. Kết luận

- **Mã hóa đối xứng** giúp bảo vệ dữ liệu khỏi bị nghe lén, nhưng yêu cầu **quản lý khóa an toàn**.
- Hệ thống này thường được sử dụng trong **AES, DES** và các giao thức như **TLS** để bảo vệ dữ liệu truyền qua Internet.
- Để giải quyết vấn đề phân phối khóa, ta có thể kết hợp **mã hóa khóa công khai** (ví dụ: Diffie-Hellman, RSA) để trao đổi khóa an toàn hơn.

Các giai đoạn phát triển chính của mật mã học

Theo Babash và các cộng sự trong tác phẩm **"Lịch sử mật mã"**, lịch sử mật mã được chia thành **ba giai đoạn chính**:

1. Mật mã sơ khai (Naïve Cryptography) – Trước thế kỷ XV

- Đây là giai đoạn ban đầu của mật mã, khi con người sử dụng **các phương pháp đơn giản** để che giấu thông tin.
- Ví dụ:
    - **Mã Caesar**: Dịch chuyển ký tự trong bảng chữ cái.
    - **Scytale của Sparta**: Dùng một dải giấy quấn quanh trụ để mã hóa.
    - **Chữ viết tượng hình bí mật** trong các nền văn minh cổ đại.

2. Mật mã chính thức (Formal Cryptography) – Trước thế kỷ XX

- Giai đoạn này đánh dấu sự phát triển của các **thuật toán mã hóa có cấu trúc rõ ràng hơn**.
- Ví dụ:
    - **Mật mã Vigenère**: Sử dụng nhiều bảng chữ cái thay đổi liên tục để mã hóa.
    - **Hệ thống mã hóa Playfair, Enigma (trong Thế chiến II)**.
    - Xuất hiện **các nguyên tắc phân tích mật mã** (phá mã) như phương pháp **tần suất ký tự** của Al-Kindi.

3. Mật mã toán học (Mathematical Cryptography) – Từ thế kỷ XX đến nay

- Giai đoạn hiện đại, khi mật mã học dựa trên các **nguyên lý toán học phức tạp** và **lý thuyết số**.
- Đặc điểm chính:
    - **Mật mã khóa công khai** (RSA, Diffie-Hellman).
    - **Mật mã đối xứng hiện đại** (AES, DES).
    - **Hàm băm (MD5, SHA-256)**.
    - **Chữ ký số, chứng chỉ số, Blockchain**.

Kết luận

Mật mã học đã phát triển từ **các phương pháp đơn giản** sang **hệ thống mã hóa phức tạp dựa trên toán học**, giúp bảo vệ thông tin trong thời đại số.

"Шифр замены. Атака полного перебора ключа."

(Mã thay thế. Tấn công vét cạn khóa.)

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
L	F	W	P	B	Y	E	Q	J	X	C	T	K	O	G	V	A	S	R	H	U	N	D	Z	I	M

1. Giới thiệu về mã thay thế

- Đây là một loại **mã hóa đơn giản**, trong đó mỗi chữ cái trong bảng chữ cái **được thay thế bằng một chữ cái khác theo một quy tắc cố định**.
- Ví dụ trong hình:
    - Chữ cái "h" → "Q", "e" → "B", "l" → "T", "o" → "G".

- Từ **"hello"** được mã hóa thành **"QBTTG"**.

2. Tấn công vét cạn khóa (Brute Force Attack)

- Do số lượng **khóa có thể có chỉ là 26! (nếu hoán vị toàn bộ bảng chữ cái) hoặc 26 (nếu chỉ dịch chuyển như mã Caesar)**, kẻ tấn công có thể **thử từng khóa một để giải mã**.
- **Quy trình tấn công**:
  1. **Kẻ tấn công chặn được bản mã** (ciphertext) và một phần bản rõ đã biết (plaintext).
  2. **Thử tất cả các khóa có thể có**, giải mã từng bản mã để kiểm tra xem nó có khớp với phần bản rõ đã biết hay không.
  3. **Nếu tìm thấy khóa giải mã ra đúng bản rõ**, xem như khóa đã được tìm thấy.

3. Nhược điểm của mã thay thế

- **Dễ bị tấn công vét cạn**: Vì số lượng khóa hữu hạn và có thể kiểm tra hết.
- **Dễ bị tấn công phân tích tần suất**: Trong một ngôn ngữ, một số chữ cái xuất hiện thường xuyên hơn (ví dụ, trong tiếng Anh, "E" là chữ cái phổ biến nhất). Bằng cách đếm số lần xuất hiện của từng chữ cái trong bản mã, có thể đoán được cách thay thế.

4. Cách bảo vệ

- **Sử dụng mã hóa hiện đại** như AES, RSA.
- **Không sử dụng mã thay thế đơn giản** trong các hệ thống bảo mật thực tế.
- **Kết hợp nhiều phương pháp mã hóa** để tăng độ an toàn, ví dụ như **mã thay thế + hoán vị** (Permutation & Substitution Ciphers).

Kết luận

- **Mã thay thế đơn giản không an toàn** vì có thể dễ dàng bị tấn công bằng phương pháp vét cạn và phân tích tần suất.
- **Hệ thống mật mã hiện đại cần có độ phức tạp cao hơn** để chống lại các loại tấn công này.

"Какие шифры стойкие к атакам перебора?"

(Những loại mã hóa nào chống lại tấn công vét cạn?)

1. Độ bền vững của mã hóa theo độ dài khóa

Bảng trong hình đánh giá mức độ chống chịu của các mã hóa trước **tấn công vét cạn khóa (brute-force attack)** dựa trên **độ dài khóa**:

Độ dài khóa	Đánh giá độ bền vững của mã hóa
56-64 bit	**Bảo mật thấp** – có thể bị phá chỉ trong vài giờ đến vài ngày.
112-128 bit	**Bảo mật cao** – cần hàng chục năm để phá nếu không có máy tính lượng tử.

256 bit	**Bảo mật rất cao** – ngay cả khi có máy tính lượng tử, cũng cần hàng chục năm để phá vỡ.

2. Giải thích chi tiết

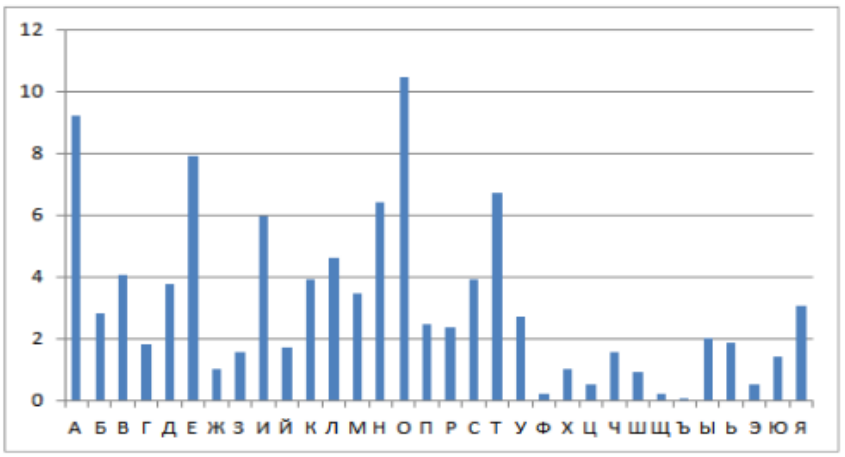
- **56-64 bit** (như mã DES)
    - **Không còn an toàn** trước tấn công vét cạn hiện nay.
    - Với **siêu máy tính**, khóa có thể bị phá chỉ trong **vài giờ đến vài ngày**.
    - Ví dụ: **DES (56-bit) bị phá vào năm 1999 bằng siêu máy tính**.
- **112-128 bit** (như 3DES, AES-128)
    - **Vẫn an toàn trước các máy tính cổ điển**.
    - Mất **hàng chục năm** để phá với các phương pháp hiện tại.
    - **Nhưng có thể bị đe dọa bởi máy tính lượng tử** trong tương lai (do thuật toán **Shor** có thể làm giảm độ an toàn).
- **256 bit** (như AES-256)
    - **Rất an toàn**, ngay cả khi có máy tính lượng tử.
    - Tấn công vét cạn vẫn mất **hàng chục năm** ngay cả khi sử dụng **các thuật toán lượng tử tiên tiến**.
    - Được khuyến nghị cho **bảo mật lâu dài** (ví dụ: dữ liệu chính phủ, tài chính).

3. Kết luận

- **Mã hóa 56-64 bit không còn an toàn** và không nên sử dụng.
- **128 bit vẫn an toàn**, nhưng có thể gặp rủi ro trong tương lai khi máy tính lượng tử phát triển.
- **256 bit là tiêu chuẩn an toàn cao nhất hiện nay**, kháng cự tốt cả tấn công cổ điển và lượng tử.

⇒ **Nếu cần bảo mật lâu dài, hãy sử dụng mã hóa 256-bit như AES-256!**

"Частотный анализ символов" (Phân tích tần suất ký tự)



1. Giới thiệu về phân tích tần suất

- Đây là một kỹ thuật **phá mã** được sử dụng để **giải mã các hệ thống mã hóa thay thế đơn giản** (như **mã Caesar, mã Vigenère**).
- **Nguyên tắc cơ bản**: Trong một ngôn ngữ, một số ký tự xuất hiện **thường xuyên hơn các ký tự khác**. Khi mã hóa văn bản, tần suất của các ký tự **vẫn giữ nguyên**, do đó có thể dùng để đoán khóa mã hóa.

2. Quy trình tấn công phân tích tần suất

1. **Xác định tần suất xuất hiện của từng ký tự trong bản mã (ciphertext).**
2. **So sánh phân bố tần suất trong bản mã với tần suất ký tự trong ngôn ngữ tự nhiên** (ví dụ, trong tiếng Nga, "O", "E", "A" là các chữ cái phổ biến nhất).
3. **Ghép nối các ký tự trong bản mã với ký tự có cùng tần suất trong ngôn ngữ**, từ đó dần **khôi phục khóa mã hóa**.

3. Ứng dụng của phương pháp này

- **Tấn công mã thay thế (Substitution Ciphers)** như **mã Caesar, mã Monoalphabetic**.
- **Tấn công mã Vigenère** bằng cách chia bản mã thành các đoạn có khóa lặp và phân tích từng đoạn riêng lẻ.
- **Được sử dụng trong mật mã học cổ điển** trước khi các thuật toán mã hóa phức tạp xuất hiện.

4. Cách phòng chống

- **Sử dụng mã hóa hiện đại** (như AES, RSA), nơi tần suất ký tự bị che giấu hoàn toàn.
- **Dùng mã hóa hoán vị kết hợp thay thế** để làm mất đi mô hình tần suất rõ ràng.
- **Nén dữ liệu trước khi mã hóa**, vì dữ liệu nén có tần suất ký tự không dễ đoán.

Kết luận

- **Phân tích tần suất là một kỹ thuật mạnh mẽ để phá mã đơn giản**, nhưng **không hiệu quả đối với mã hóa hiện đại**.
- **Các hệ thống mã hóa tiên tiến cần đảm bảo rằng bản mã không lộ ra mô hình tần suất của văn bản gốc** để tránh tấn công.

"Шифры сдвига"

(Mã hóa dịch chuyển)

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

1. Giới thiệu về mã hóa dịch chuyển

- Đây là một trong những **phương pháp mã hóa đơn giản nhất**, trong đó **mỗi chữ cái trong bảng chữ cái được thay thế bằng một chữ cái khác, dịch chuyển theo một số vị trí cố định (k).**
- Ví dụ:
    - **hello → KHOOR** khi **k = 3** (Mã Caesar).
    - Nếu **dịch lùi lại 3 vị trí**, ta có thể **giải mã** bản mã để khôi phục văn bản gốc.

2. Mã Caesar (Caesar Cipher)

- Là một **dạng đặc biệt của mã dịch chuyển**, trong đó **k = 3**.
- Ví dụ:
    - **A → D, B → E, C → F** (dịch phải 3 vị trí).
    - Nếu biết trước **k = 3**, thì việc giải mã cực kỳ dễ dàng.

3. Mã dịch chuyển có dễ bị phá vỡ không?

a) Dễ bị tấn công vét cạn (Brute-force attack)?

✔ **Có.**

- Vì **khóa k chỉ có 25 giá trị có thể có** (k từ 1 đến 25), kẻ tấn công có thể **thử tất cả các giá trị k** và tìm ra khóa đúng trong **thời gian rất ngắn**.

b) Dễ bị phân tích tần suất (Frequency Analysis)?

✔ **Có.**

- Trong bất kỳ văn bản nào, một số chữ cái xuất hiện **nhiều hơn các chữ cái khác** (ví dụ, trong tiếng Anh, "E" là chữ cái phổ biến nhất).
- Sau khi mã hóa, **tần suất chữ cái không thay đổi**, chỉ bị dịch chuyển.
- Kẻ tấn công có thể **so sánh tần suất của bản mã với tần suất của ngôn ngữ gốc**, từ đó khôi phục lại khóa **rất dễ dàng**.

4. Cách bảo vệ

- **Không sử dụng mã dịch chuyển trong bảo mật hiện đại**.
- **Kết hợp hoán vị và thay thế** (ví dụ: sử dụng **mã Vigenère hoặc AES**).
- **Sử dụng mã hóa đối xứng mạnh** như **AES hoặc DES** để tránh tấn công vét cạn và phân tích tần suất.

Kết luận

- **Mã dịch chuyển không an toàn**, dễ bị phá bằng cả **tấn công vét cạn** và **phân tích tần suất**.
- **Không nên sử dụng mã này để bảo mật thực tế**, mà chỉ nên dùng để **giảng dạy về nguyên lý mã hóa**.

"Модульная арифметика"

(Số học đồng dư – Modular Arithmetic)

1. Giới thiệu về phép toán modulo

- **Phép toán modulo (lấy dư)** là phép toán **tính phần dư khi chia một số nguyên cho một số nguyên khác**.
- Nếu **a ≡ r (mod m)**, thì **m chia hết (a - r)**.
- Trong đó:
    - **m** gọi là **modulus (mô-đun, số chia)**.
    - **r** gọi là **remainder (phần dư)**.

Ví dụ:

- 10 mod 3 = 1 (vì 10 = 3 x 3 +1, dư 1)
- 15 mod 4 = 3 (vì 15 = 3 x 4 + 3, dư 3)

2. Ví dụ trong tập hợp số hữu hạn {0,1,2,3,4,5,6} với mô-đun 7

- **$(2\times3) \bmod 7=6$**
- **$(3-2) \bmod 7=1$**
- **$(4+5) \bmod 7=2$** (vì 4+5=9, mà 9÷7 dư **2**).

3. Ứng dụng của số học đồng dư

- **Mật mã học**:
    - **RSA Cryptosystem**: Dựa trên số học mô-đun với số nguyên tố lớn.
    - **Elliptic Curve Cryptography (ECC)**: Dùng số học đồng dư trên đường cong elliptic.
- **Kiểm tra tính chẵn lẻ**: x mod 2 = 0 (số chẵn), x mod 2 = 1 (số lẻ)
- **Đồng hồ (Clock Arithmetic)**: 14 giờ + 9 giờ = 23 giờ, nhưng 23 mod 12 = 11 giờ.

Kết luận

- **Số học đồng dư là nền tảng của mật mã và lý thuyết số.**
- **Phép toán modulo giúp giới hạn giá trị trong một phạm vi nhất định (số dư)**.
- **Ứng dụng mạnh mẽ trong mật mã, kiểm tra tính chẵn lẻ, và hệ thống thời gian.**

"Шифр сдвига через модульную арифметику"

(Mã hóa dịch chuyển bằng số học đồng dư)

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

1. Cách biểu diễn mã hóa dịch chuyển bằng số học đồng dư

- Trong sơ đồ trên, các ký tự trong bảng chữ cái được ánh xạ thành **các số từ 0 đến 25**.
- Quá trình **mã hóa** và **giải mã** được thực hiện bằng **phép toán modulo 26**.

2. Công thức mã hóa và giải mã

- **Mã hóa (Encryption)**:
  $y=(x+k) \bmod 26$
  - **x**: Số thứ tự của chữ cái trong bảng chữ cái.
  - **k**: Khóa mã hóa (số lượng dịch chuyển).
  - **y**: Số thứ tự của ký tự mã hóa sau khi dịch chuyển.
- **Giải mã (Decryption)**:
  $x=(y-k) \bmod 26$
  - Khi nhận được bản mã, ta dịch ngược lại **k vị trí** để lấy lại bản rõ.

3. Ví dụ mã hóa và giải mã

Giả sử ta mã hóa chữ **"HELLO"** với **k = 3**:

- H → 7 → (7 + 3) mod 26 = 10 → K
- E → 4 → (4 + 3) mod 26 = 7 → H
- L → 11 → (11 + 3) mod 26 = 14 → O
- L → 11 → (11 + 3) mod 26 = 14 → O
- O → 14 → (14 + 3) mod 26 = 17 → R

Kết quả mã hóa **"HELLO" → "KHOOR"** (đây là **Mã Caesar với k = 3**).

Bây giờ, để giải mã "KHOOR":

- K → 10 → (10 - 3) mod 26 = 7 → H
- H → 7 → (7 - 3) mod 26 = 4 → E
- O → 14 → (14 - 3) mod 26 = 11 → L
- O → 14 → (14 - 3) mod 26 = 11 → L
- R → 17 → (17 - 3) mod 26 = 14 → O

Ta khôi phục lại "HELLO".

4. Ưu điểm và Nhược điểm của phương pháp này

- **Ưu điểm**:
    - Dễ hiểu, dễ thực hiện bằng tay hoặc máy tính.
    - Đơn giản trong việc mã hóa và giải mã bằng công thức toán học.
- **Nhược điểm**:
    - **Dễ bị tấn công vét cạn** (chỉ có 25 khóa có thể thử).
    - **Dễ bị phân tích tần suất**, vì tần suất chữ cái vẫn được bảo toàn.
    - **Không phù hợp cho bảo mật thực tế**.

Kết luận

- **Mã hóa dịch chuyển có thể biểu diễn bằng số học đồng dư modulo 26**.
- **Dễ thực hiện nhưng không an toàn trước các tấn công mật mã học**.
- **Chỉ thích hợp để minh họa nguyên lý mã hóa, không dùng cho bảo mật thực tế**.
- **Hệ thống mã hóa hiện đại (như AES, RSA) sử dụng số học đồng dư trên các số nguyên lớn để đảm bảo an toàn hơn.**

"Модульная арифметика для криптографии"

(Số học đồng dư trong mật mã)

11=x mod 7, x /in Z

11=18 mod 7

11=25 mod 7

11=4 mod 7

....

1. Ý nghĩa của phép toán đồng dư

- Biểu thức **x ≡ 11 mod 7** có nghĩa là **x khi chia cho 7 sẽ có dư 11**.
- Tuy nhiên, trong số học đồng dư, **một số có nhiều đại diện khác nhau** theo mô-đun.

2. Các ví dụ trong hình

Xét mô-đun **7**, ta có:

11 mod 7 = 4

18 mod 7 = 4

25 mod 7 = 4

4 mod 7 = 4

Kết luận: Tất cả các số này đều có cùng phần dư là **4** khi chia cho **7**.

3. Ứng dụng trong mật mã

- **Lý thuyết số** và **số học đồng dư** là nền tảng cho nhiều thuật toán mật mã như:
    - **RSA**: Sử dụng phép toán đồng dư trên số nguyên lớn để mã hóa và giải mã.
    - **Elliptic Curve Cryptography (ECC)**: Dùng số học đồng dư trên đường cong elliptic để tạo ra hệ mật mã mạnh mẽ.
    - **Hệ mật mã dựa trên phần dư đồng dư (Residue Number Systems - RNS)**: Dùng để tăng tốc độ tính toán trong hệ mật mã lượng tử.

4. Kết luận

- **Phép toán modulo không có giá trị duy nhất**, vì nhiều số nguyên khác nhau có thể có cùng phần dư.
- **Điều này rất quan trọng trong mật mã**, vì nó giúp ẩn thông tin gốc, tạo nên các thuật toán mã hóa mạnh mẽ.
- **Mật mã hiện đại sử dụng số học đồng dư trên các số rất lớn** để đảm bảo tính bảo mật chống lại tấn công vét cạn.

"Шифр перестановки"

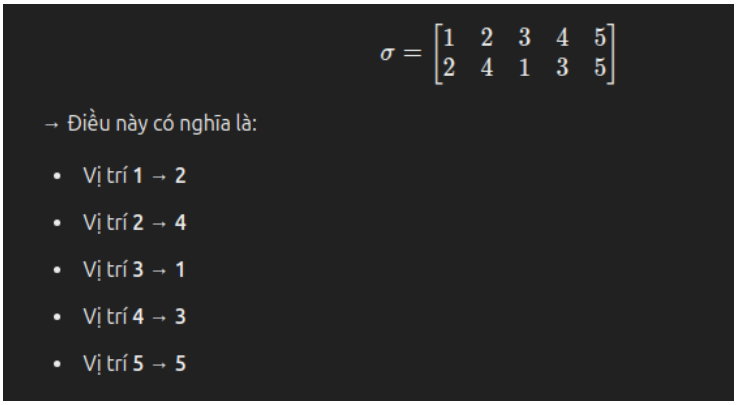
(Mã hóa hoán vị)

1. Giới thiệu về mã hóa hoán vị (Permutation Cipher)

- **Mã hóa hoán vị** là một phương pháp trong đó **các ký tự của văn bản gốc bị tráo đổi vị trí theo một quy tắc cố định**.
- Không thay đổi nội dung của từng ký tự, chỉ **thay đổi thứ tự xuất hiện**.

2. Ví dụ về mã hóa hoán vị

- **Dãy hoán vị được sử dụng**:



Ví dụ mã hóa trong hình:

1. **Bản gốc**:
   "once upon a time there was a little girl called Snow White"
2. **Sau bước 1 của hoán vị**:
   "onceu ponat imeth erewa salit tlegi rlcal ledsn owhwi te"
3. **Sau bước 2 của hoán vị**:
   "coenu npoat eimth eewra lsiat etgli crall dlsen wohwi et"
4. **Sau nhiều lần hoán vị**:
   "Coennpaoeti mtheewr alsiatetglicr alldlsenowhwiet"

3. Tính bảo mật của mã hóa hoán vị

- **Dễ bị phá bằng phân tích tần suất**: Vì số lượng ký tự không thay đổi, tần suất xuất hiện của từng ký tự vẫn giống với văn bản gốc.
- **Dễ bị tấn công bằng phân tích mẫu (Pattern Analysis)**: Nếu kẻ tấn công biết rằng chỉ có sự hoán vị, họ có thể **sắp xếp lại các ký tự dựa vào ngữ nghĩa** để tìm ra thông điệp ban đầu.
- **Có thể bị phá bằng tấn công vét cạn (Brute Force)**: Nếu độ dài khóa **n** nhỏ, có thể thử **n!** hoán vị để tìm ra khóa đúng.

4. Phân biệt mã hóa đơn bảng và đa bảng

- **Monoalphabetic Cipher (Mã đơn bảng)**:
  - Mỗi ký tự bản rõ **luôn ánh xạ vào một ký tự bản mã duy nhất**.
  - Ví dụ: **Mã Caesar, Mã thay thế đơn giản**.
  - **Dễ bị phân tích tần suất**.
- **Polyalphabetic Cipher (Mã đa bảng)**:
  - Một ký tự bản rõ có thể **ánh xạ vào nhiều ký tự bản mã khác nhau**, tùy vào vị trí hoặc điều kiện mã hóa.
  - Ví dụ: **Mã Vigenère, Mã Enigma**.
  - **Khó bị phá hơn mã đơn bảng**, vì tần suất ký tự không còn rõ ràng.

5. Cách tăng độ bảo mật

- **Kết hợp hoán vị và thay thế (Substitution + Permutation)**:
  - Mã DES sử dụng **hoán vị kết hợp với thay thế** để tạo ra hệ mã mạnh hơn.

- **Sử dụng nhiều vòng hoán vị**:
    - Nếu chỉ hoán vị một lần, kẻ tấn công có thể sắp xếp lại dễ dàng.
    - Nếu hoán vị **nhiều vòng với khóa thay đổi**, việc giải mã sẽ khó hơn.
- **Dùng mã hóa đối xứng hiện đại như AES**:
    - AES sử dụng **hoán vị, thay thế và phép toán số học đồng dư**, làm tăng đáng kể độ an toàn.

6. Kết luận

- **Mã hóa hoán vị đơn thuần không an toàn**, vì vẫn giữ nguyên tần suất chữ cái.
- **Dễ bị phá nếu chỉ sử dụng một vòng hoán vị**.
- **Phù hợp cho mục đích giảng dạy về mã hóa**, nhưng **không dùng cho bảo mật thực tế**.
- **Mật mã hiện đại (DES, AES) sử dụng hoán vị kết hợp với thay thế** để tăng cường độ bảo mật.

"Этап формальной криптографии (с XV до XX)"

(Giai đoạn mật mã chính thức – từ thế kỷ XV đến thế kỷ XX)

1. Giai đoạn mật mã chính thức là gì?

- Đây là giai đoạn trong lịch sử mật mã khi **các phương pháp mã hóa trở nên có hệ thống hơn**.
- Xuất hiện **các phương pháp tấn công mới và hệ mã mới** để chống lại chúng.

2. Đóng góp quan trọng trong giai đoạn này

a) Battista Alberti và Phân tích Tần suất (XV thế kỷ)

- **Leon Battista Alberti** (1404–1472) là một trong những nhà tiên phong trong lĩnh vực mật mã học.
- Ông đề xuất **sử dụng phân tích tần suất để phá mã monoalphabetic**.
- **Nguyên lý phân tích tần suất**:
    - Trong một ngôn ngữ, một số chữ cái xuất hiện thường xuyên hơn (ví dụ: "E" phổ biến trong tiếng Anh).
    - Khi sử dụng **mật mã thay thế đơn giản**, tần suất chữ cái không thay đổi.
    - Bằng cách thống kê, có thể **khôi phục bảng thay thế** và giải mã bản mã.
- **Tác động**: Phương pháp này **đánh dấu sự kết thúc của mã monoalphabetic đơn giản**, vì chúng dễ bị phá.

b) Blaise de Vigenère và Mật mã Đa bảng (XVI thế kỷ)

- **Blaise de Vigenère** (1523–1596) đã đề xuất một **hệ thống mã hóa mạnh hơn – Mã Vigenère**.
- **Đặc điểm của mã Vigenère**:
    - **Là mã polyalphabetic** (một ký tự bản rõ có thể mã hóa thành nhiều ký tự bản mã khác nhau).

    - Sử dụng **một bảng chữ cái dịch chuyển**, với một khóa xác định vị trí mã hóa.
    - Ví dụ, với khóa "KEY", ta có thể mã hóa "HELLO" bằng cách thay đổi dịch chuyển theo từng ký tự.
- **Lợi ích**:
    - **Khó bị tấn công bằng phân tích tần suất**, vì không có tần suất cố định cho từng chữ cái.
    - **An toàn hơn so với các mã đơn bảng như mã Caesar**.

3. Kết luận

- Giai đoạn này đánh dấu bước tiến từ **mật mã đơn giản** (monoalphabetic) sang **mật mã phức tạp hơn** (polyalphabetic).
- **Phân tích tần suất** trở thành công cụ quan trọng để phá mã đơn giản.
- **Mã hóa đa bảng** (Vigenère) được phát triển để **chống lại phân tích tần suất**.
- Mặc dù mạnh hơn mã thay thế đơn giản, **mã Vigenère vẫn có thể bị phá** nếu khóa quá ngắn hoặc có mẫu lặp.

🔹 **Hướng phát triển tiếp theo**: Sang thế kỷ XX, mật mã toán học (Mathematical Cryptography) ra đời, mở đường cho **mật mã đối xứng (AES, DES) và mã khóa công khai (RSA, Diffie-Hellman).**

"Шифр Виженера" (**Mật mã Vigenère**)

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

1. Giới thiệu về Mật mã Vigenère

- Mật mã Vigenère là một loại **mã hóa polyalphabetic** (đa bảng chữ cái), giúp **tăng cường bảo mật so với mã thay thế đơn giản**.
- **Điểm mạnh**: Không có **sự lặp lại trực tiếp của ký tự** như trong mã Caesar, giúp **chống lại phân tích tần suất**.
- **Nguyên tắc hoạt động**:
    - Mỗi ký tự trong bản rõ (plaintext) được mã hóa bằng một ký tự **khác nhau** dựa trên một khóa (key).
    - Dùng **bảng Vigenère** (hiển thị trong hình), là **tập hợp của nhiều bảng mã Caesar**, mỗi hàng dịch một ký tự theo bảng chữ cái.

2. Cách mã hóa bằng Mật mã Vigenère

Ví dụ với dữ liệu trong hình

- **Khóa (Key):** "secret"
- **Thông điệp gốc (Plaintext):** "hello"
- **Bản mã (Ciphertext):** "zincs"

Bước 1: Lặp lại khóa để khớp với độ dài của văn bản

Văn bản "hello" có 5 ký tự, trong khi khóa "secret" có 6 ký tự.
→ Lặp lại khóa: "secre" (chỉ lấy 5 ký tự đầu để mã hóa).

Bước 2: Mã hóa từng ký tự

Tra cứu từng ký tự trong **bảng Vigenère** bằng cách:

- Cột đầu tiên là **chữ cái trong khóa**.
- Hàng đầu tiên là **chữ cái trong bản rõ**.
- Điểm giao nhau là **chữ cái mã hóa**.

Plaintext	**h**	**e**	**l**	**l**	**o**
Key	s	e	c	r	e
Ciphertext	**z**	**i**	**n**	**c**	**s**

3. Cách giải mã Mật mã Vigenère

Ngược lại với mã hóa:

1. Tìm **hàng khóa** trong bảng Vigenère.
2. Xác định vị trí của ký tự bản mã trong hàng đó.
3. Cột đầu tiên của bảng sẽ cho ra **ký tự gốc**.

Ví dụ giải mã "zincs" với khóa "secret" sẽ khôi phục lại "hello".

4. Bảo mật của Mật mã Vigenère

- **Ưu điểm**:
    - **Chống lại phân tích tần suất** vì một ký tự bản rõ có thể mã hóa thành **nhiều ký tự khác nhau**.
    - **Dễ thực hiện và sử dụng**.
- **Nhược điểm**:
    - **Dễ bị phá nếu khóa quá ngắn hoặc lặp lại** (tấn công Kasiski hoặc phân tích tần suất trên các đoạn trùng lặp).
    - **Không đủ an toàn cho bảo mật hiện đại**, vì các máy tính có thể **tấn công vét cạn để tìm khóa**.

5. Cách tăng cường bảo mật

- **Dùng khóa đủ dài**: Nếu khóa có độ dài bằng văn bản và hoàn toàn ngẫu nhiên (**One-Time Pad**), mã Vigenère trở nên **không thể phá vỡ**.
- **Kết hợp với thuật toán mã hóa hiện đại** như AES để đảm bảo tính bảo mật.

6. Kết luận

- **Mật mã Vigenère là một trong những bước tiến quan trọng trong lịch sử mật mã**, giúp **nâng cấp từ mã đơn bảng (monoalphabetic) lên đa bảng (polyalphabetic)**.
- **Tuy nhiên, nó vẫn có điểm yếu**, đặc biệt là khi khóa ngắn hoặc lặp lại.
- **Ngày nay, Vigenère không còn được sử dụng để bảo mật thực tế**, nhưng vẫn là một **công cụ giáo dục quan trọng trong mật mã học**.

"Метод взлома полиалфавитных шифров"

(Phương pháp phá mã đa bảng chữ cái)

1. Giới thiệu về phương pháp Kasiski

- **Năm 1863**, nhà mật mã **Friedrich Kasiski** đã xuất bản công trình **"Тайнопись и искусство дешифрования"** (*Mật mã học và nghệ thuật giải mã*).
- Ông giới thiệu **một phương pháp phá mã tổng quát** dành cho **mã hóa đa bảng chữ cái (polyalphabetic ciphers)**, đặc biệt là **mật mã Vigenère**.
- Trước đó, người ta **tưởng rằng mã Vigenère là không thể phá**, nhưng Kasiski đã chứng minh điều ngược lại.

2. Phương pháp Kasiski – Cách thức hoạt động

Phương pháp Kasiski bao gồm **hai bước chính**:

Bước 1: Xác định độ dài khóa bằng cách phân tích lặp lại (Repeated Pattern Analysis)

- **Khi sử dụng mật mã Vigenère với một khóa lặp lại, một số cụm từ (trigram, bigram) có thể xuất hiện lặp lại trong bản mã**.
- Kasiski nhận thấy rằng **khoảng cách giữa các cụm từ lặp lại có thể chia hết cho độ dài của khóa**.
- Ví dụ:

Bản mã:
HXZOP YKX ZOP LKZOP

    - Cụm từ **"ZOP"** xuất hiện nhiều lần.
    - Khoảng cách giữa chúng là **6 ký tự** → có thể suy luận rằng **khóa có độ dài 3 hoặc 6**.

Bước 2: Phá mã bằng phân tích tần suất

- Sau khi xác định độ dài khóa **k**, bản mã được **chia thành các khối có độ dài k**.
- **Mỗi khối được xem như một mã đơn bảng (monoalphabetic cipher)**.
- **Áp dụng phân tích tần suất lên từng khối** để khôi phục lại khóa.

3. Tác động của phương pháp Kasiski

- **Lần đầu tiên, mã Vigenère có thể bị phá một cách có hệ thống.**
- **Không còn được coi là "mật mã không thể phá" như trước đó**.
- **Dẫn đến sự phát triển của các hệ mã mạnh hơn, chẳng hạn như One-Time Pad (mật mã khóa một lần).**

4. Cách bảo vệ chống lại tấn công Kasiski

- **Dùng khóa có độ dài bằng văn bản** → chuyển thành **One-Time Pad**, không thể phá được.
- **Tránh sử dụng khóa ngắn và lặp lại**.
- **Sử dụng các thuật toán mã hóa hiện đại (AES, RSA) thay vì mật mã Vigenère**.

5. Kết luận

- **Phương pháp Kasiski là một trong những bước tiến quan trọng trong phân tích mật mã**.
- **Làm suy yếu đáng kể mật mã Vigenère**, thúc đẩy sự phát triển của **các hệ mã an toàn hơn**.
- **Ngày nay, các thuật toán hiện đại như AES, RSA không bị ảnh hưởng bởi tấn công này**

"Принцип Керкгоффса"

(Nguyên tắc Kerckhoffs trong mật mã học)

1. Nguyên tắc Kerckhoffs là gì?

- **Nguyên tắc Kerckhoffs** do nhà mật mã học **Auguste Kerckhoffs** đề xuất vào năm **1883**.
- Nội dung chính:

    - **Một hệ thống mật mã vẫn phải đảm bảo tính bảo mật ngay cả khi kẻ tấn công biết mọi thông tin về hệ thống**, ngoại trừ **khóa mật mã**.
    - **Bảo mật phải dựa vào khóa chứ không phải vào tính bí mật của thuật toán**.

2. Ý nghĩa của nguyên tắc Kerckhoffs

- **Chống lại bảo mật "ẩn"**: Nếu một hệ thống chỉ an toàn khi thuật toán được giữ bí mật, nó sẽ **rất dễ bị phá** nếu thuật toán bị lộ.
- **Hệ thống phải chịu được phân tích ngược**: Hệ mã cần đủ mạnh để chống lại các cuộc tấn công ngay cả khi thuật toán đã bị công khai.
- **Tạo ra các tiêu chuẩn mã hóa an toàn**: Ví dụ, các thuật toán như **AES, RSA** đều **được công khai**, nhưng vẫn đảm bảo an toàn vì bảo mật dựa trên **khóa**.

3. Ví dụ về nguyên tắc Kerckhoffs

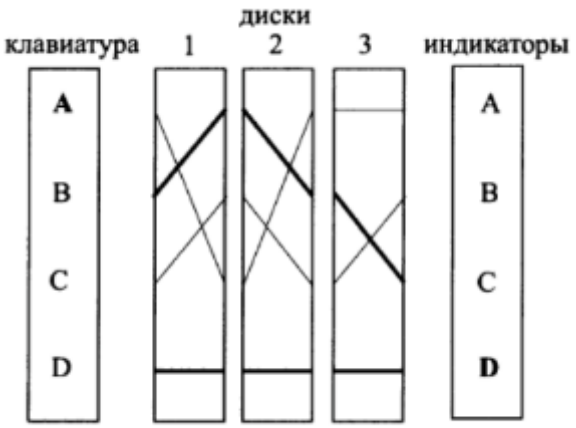
- **Hệ thống an toàn theo nguyên tắc này**:
    - **AES (Advanced Encryption Standard)**: Mọi người đều biết thuật toán AES, nhưng nếu không có khóa, dữ liệu vẫn không thể bị giải mã.
    - **RSA (Rivest-Shamir-Adleman)**: Mã hóa dựa trên toán học với số nguyên tố lớn, chỉ an toàn nếu khóa bí mật không bị lộ.
    - **TLS/SSL (Giao thức bảo mật trên Internet)**: Mọi người đều biết cách hoạt động, nhưng nếu không có khóa, không thể giải mã dữ liệu.
- **Hệ thống vi phạm nguyên tắc này**:
    - **Bảo mật "ẩn" trong phần mềm độc quyền**: Một số hệ thống giữ thuật toán mã hóa bí mật, nếu thuật toán bị lộ, hệ thống **sẽ hoàn toàn bị phá vỡ**.
    - **Mật mã xoay vòng (ROT13)**: Dễ bị phá vỡ vì thuật toán quá đơn giản.

4. Kết luận

- **Bảo mật không nên dựa vào sự bí mật của thuật toán**, mà phải dựa vào **khóa mật mã**.
- **Hệ thống mã hóa hiện đại đều tuân theo nguyên tắc Kerckhoffs** để đảm bảo tính an toàn ngay cả khi thuật toán bị công khai.
- **Một hệ mật mã an toàn là hệ mà ngay cả khi thuật toán bị lộ, vẫn không thể giải mã dữ liệu nếu không có khóa bí mật.**

"Роторные машины"

(Máy mã hóa rotor)



1. Giới thiệu về máy mã hóa rotor

- **Máy mã hóa rotor** là một thiết bị cơ điện dùng để **mã hóa và giải mã thông tin**, được sử dụng phổ biến trong **Thế chiến II**.
- **Hoạt động dựa trên các rotor xoay**, làm thay đổi liên tục các đường ánh xạ giữa các ký tự bản rõ và bản mã.
- **Máy Enigma của Đức Quốc Xã** là một trong những máy mã hóa rotor nổi tiếng nhất.

2. Cấu trúc của máy rotor (hình minh họa)

- **Клавиатура (Bàn phím)**: Người dùng nhập ký tự bản rõ (plaintext).
- **Диски (Rotor 1, 2, 3 - Các đĩa quay)**:
    - Mỗi rotor có một hệ thống **dây nối điện bên trong**, hoán vị các ký tự khi tín hiệu đi qua.
    - Sau mỗi lần nhập ký tự, **ít nhất một rotor quay**, thay đổi ánh xạ để tạo mã hóa phức tạp hơn.
- **Индикаторы (Bản mã hiển thị)**: Ký tự mã hóa xuất hiện trên màn hình hoặc bảng đèn.

3. Cách hoạt động của máy mã hóa rotor

1. Khi một phím được nhấn (ví dụ: **A**), tín hiệu điện đi qua nhiều **rotor**, mỗi rotor thực hiện một phép thay thế.
2. Sau khi đi qua tất cả các rotor, tín hiệu đến **bộ phản xạ**, sau đó quay trở lại qua các rotor một lần nữa.
3. Ký tự bản mã (ciphertext) hiển thị trên màn hình.

4. **Mỗi lần nhập ký tự, rotor đầu tiên quay, và các rotor khác có thể quay theo cơ chế nhất định** → tạo ra một chuỗi mã hóa phức tạp, không lặp lại ngay lập tức.

4. Độ bảo mật và cách phá mã

- **Tại sao máy rotor an toàn?**
    - Vì **mỗi lần nhập ký tự, ánh xạ thay đổi**, tạo ra bản mã rất phức tạp.
    - Có hàng triệu cấu hình khác nhau tùy vào số lượng rotor và cách sắp xếp chúng.
- **Làm sao máy Enigma bị phá?**
    - Các nhà mật mã như **Alan Turing và nhóm Bletchley Park** đã phát triển **"bombe"**, một máy tính giúp phân tích và tìm khóa mã hóa.
    - Các sai lầm vận hành của quân đội Đức (ví dụ: dùng các cụm từ phổ biến) giúp xác định cấu trúc của rotor.

5. Ứng dụng và ảnh hưởng

- **Sau chiến tranh**, công nghệ của máy rotor **dẫn đến sự phát triển của máy tính điện tử và mật mã hiện đại**.
- **Ngày nay, máy rotor không còn được sử dụng**, nhưng ý tưởng về **mã hóa liên tục thay đổi** đã ảnh hưởng đến thiết kế của **mật mã đối xứng (AES, DES)**.

6. Kết luận

- **Máy mã hóa rotor là một bước tiến quan trọng trong lịch sử mật mã**, đặc biệt là **máy Enigma**.
- **Mặc dù mạnh, nhưng do lỗi vận hành và tiến bộ trong mật mã học, hệ thống này đã bị phá vỡ**.
- **Những nguyên lý của nó vẫn ảnh hưởng đến các thuật toán mã hóa hiện đại ngày nay.**

"Этап математической криптографии"

(Giai đoạn mật mã toán học)

1. Giới thiệu về giai đoạn mật mã toán học

- Giai đoạn này bắt đầu từ **thế kỷ XX đến nay**, khi **mật mã học chuyển từ phương pháp thủ công sang sử dụng nền tảng toán học phức tạp**.
- Các hệ thống mã hóa hiện đại được xây dựng dựa trên **các nguyên lý toán học vững chắc**.

2. Những đặc điểm chính của giai đoạn này

a) Phát triển các ngành toán học liên quan đến mật mã

- **Lý thuyết nhóm hữu hạn (Finite Fields Theory)**: Dùng trong mã hóa **AES, RSA, ECC**.

- **Lý thuyết số (Number Theory)**: Cơ sở của các thuật toán như **RSA, Diffie-Hellman**.
- **Xác suất & Thống kê (Probability & Statistics)**: Phân tích độ mạnh của các thuật toán mã hóa.

b) Mật mã được nghiên cứu theo mô hình toán học

- **Hệ mã hóa được mô tả bằng các công thức toán học thay vì chỉ là hoán vị ký tự**.
- **Các phép toán trên số lớn (modular arithmetic) trở thành nền tảng quan trọng trong mật mã**.

c) Giới thiệu khái niệm về độ an toàn của mật mã

- **Mật mã lý thuyết (Theoretical Cryptographic Security)**:
    - Dựa trên **các bằng chứng toán học** về độ khó của việc phá mã.
    - Ví dụ: **RSA dựa trên độ khó của bài toán phân tích thừa số nguyên tố**.
- **Mật mã thực tiễn (Practical Cryptographic Security)**:
    - Đánh giá khả năng chống lại các **tấn công thực tế** (tấn công thời gian, tấn công kênh bên, brute force).

3. Các tiêu chí đánh giá độ an toàn của hệ mật mã

- **Độ dài khóa**: Khóa càng dài, càng khó bị tấn công vét cạn.
- **Độ phức tạp của thuật toán**: Cần đảm bảo tốc độ xử lý nhanh nhưng vẫn đủ an toàn.
- **Ảnh hưởng của lỗi (Error Impact)**: Một lỗi nhỏ trong hệ thống có thể làm giảm độ bảo mật không?
- **Mức độ dư thừa (Redundancy)**: Liên quan đến khả năng tối ưu dữ liệu trong quá trình mã hóa.

4. Kết luận

- **Giai đoạn mật mã toán học đánh dấu sự chuyên nghiệp hóa của lĩnh vực này**, với sự tham gia của các nhà toán học, nhà khoa học máy tính.
- **Các thuật toán hiện đại như AES, RSA, ECC được thiết kế dựa trên nền tảng toán học vững chắc**.
- **Tính bảo mật không chỉ dựa vào thiết kế thuật toán, mà còn vào độ khó của các bài toán toán học liên quan**.
- **Sự phát triển của máy tính lượng tử có thể đặt ra những thách thức mới cho mật mã học, đòi hỏi sự phát triển của các thuật toán mật mã hậu lượng tử (Post-Quantum Cryptography).**

"Криптостойкость"

(Độ an toàn mật mã - Cryptographic Strength)

1. Định nghĩa về độ an toàn mật mã

- **Криптостойкость (Cryptographic Strength)** là **khả năng của một thuật toán mật mã chống lại việc bị phá mã**.
- Độ an toàn của thuật toán mật mã đến từ **cấu trúc toán học của nó**, không chỉ dựa vào các cơ chế bảo mật bổ sung.
- Một thuật toán được coi là **mạnh** nếu nó có thể chống lại tất cả các phương pháp tấn công đã biết.

2. Phân loại độ an toàn mật mã

Có hai loại chính: **Độ an toàn tính toán (Computational Security)** và **Độ an toàn lý thuyết (Theoretical Security)**.

a) Độ an toàn tính toán (Computational Security)

- **Còn được gọi là độ an toàn thực tiễn**.
- **Phá mã là có thể**, nhưng **yêu cầu quá nhiều tài nguyên** khiến việc thực hiện **không thực tế**.
- **Ví dụ về tài nguyên**:
    - **Thời gian (Time)**: Nếu cần **hàng triệu năm** để phá mã bằng siêu máy tính, thì thuật toán vẫn được coi là an toàn.
    - **Bộ nhớ (Memory)**: Nếu tấn công yêu cầu **lưu trữ hàng tỷ terabyte dữ liệu**, điều này không khả thi.
    - **Số lượng văn bản mã hóa cần thu thập (Ciphertext Requirements)**: Nếu cần hàng triệu bản mã để tấn công, thì thuật toán vẫn an toàn trong thực tế.

✅ **Ví dụ thuật toán có độ an toàn tính toán**:

- **RSA**: Nếu sử dụng khóa **2048-bit**, việc phân tích thừa số của số nguyên lớn cần quá nhiều tài nguyên.
- **AES-256**: Mất **hàng tỷ năm** để thử tất cả các khóa có thể có (brute-force).

b) Độ an toàn lý thuyết (Theoretical Security)

- **Còn được gọi là độ an toàn toán học**.
- **Phá mã là không thể về mặt lý thuyết**, vì bài toán toán học cơ bản của thuật toán **không thể giải quyết được**.
- **Dựa trên các bài toán toán học chưa có thuật toán giải hiệu quả**, ví dụ:
    - **Phép nhân số nguyên lớn là dễ, nhưng phân tích thừa số của số lớn là khó** (RSA).
    - **Tính toán logarit rời rạc (Discrete Logarithm) là khó** (Diffie-Hellman, ECC).
    - **Các bài toán trên mạng tinh thể (Lattice-based problems)** được dùng trong mật mã hậu lượng tử (Post-Quantum Cryptography).

✅ **Ví dụ thuật toán có độ an toàn lý thuyết**:

- **One-Time Pad**: Nếu khóa **hoàn toàn ngẫu nhiên và chỉ dùng một lần**, nó không thể bị phá bởi bất kỳ phương pháp nào.
- **Mật mã lượng tử (Quantum Cryptography)**: Dựa trên nguyên lý cơ học lượng tử, việc nghe lén có thể bị phát hiện.

3. Kết luận

- **Độ an toàn mật mã có thể dựa vào khả năng tính toán hoặc lý thuyết toán học**.
- **Hầu hết các hệ thống hiện đại (RSA, AES) chỉ có độ an toàn tính toán**, nghĩa là nếu có đủ tài nguyên, chúng **có thể bị phá**.
- **Mục tiêu của nghiên cứu mật mã hiện đại là phát triển các thuật toán có độ an toàn lý thuyết cao hơn**, đặc biệt là trong kỷ nguyên máy tính lượng tử.

🔹 **Vì vậy, để đảm bảo an toàn dài hạn, các hệ thống mật mã phải liên tục được cải tiến và nâng cấp theo sự phát triển của công nghệ.**

"Криптосистемы этапа математической криптографии"

(Các hệ mật mã trong giai đoạn mật mã toán học)

1. Hệ mật mã đối xứng (Symmetric Cryptosystems)

Hệ mật mã đối xứng sử dụng **cùng một khóa để mã hóa và giải mã**. Đây là phương pháp **tốc độ nhanh**, nhưng gặp vấn đề về **phân phối khóa**.

Các thuật toán quan trọng:

1. **1960 – DES (Lucifer) – Feistel**
    - **Thuật toán mã hóa khối đầu tiên**, được IBM phát triển.
    - **Được chuẩn hóa thành DES (Data Encryption Standard) vào năm 1977**.
    - **Hiện nay không còn an toàn** do độ dài khóa chỉ 56-bit.
2. **1989 – GOST 28147**
    - **Thuật toán mã hóa đối xứng của Liên Xô (Nga)**, tương tự DES nhưng có cấu trúc khác.
    - **Vẫn được sử dụng trong một số hệ thống của Nga**.
3. **1991 – IDEA (International Data Encryption Algorithm)**
    - Được thiết kế bởi **Ascom** (Thụy Sĩ).
    - **Dùng trong PGP (Pretty Good Privacy)** để bảo vệ email.
    - **Bảo mật tốt nhưng bị hạn chế bởi bằng sáng chế**.
4. **1998 – Twofish – Bruce Schneier**
    - **Ứng cử viên trong cuộc thi chọn AES**, có thiết kế mạnh mẽ.
    - **Không được chọn làm AES, nhưng vẫn được sử dụng trong một số hệ thống bảo mật**.
5. **1998 – AES (Rijndael) – Vincent Rijmen & Joan Daemen**
    - **Được chọn làm tiêu chuẩn mã hóa của Hoa Kỳ vào năm 2002**.
    - **Hỗ trợ độ dài khóa 128-bit, 192-bit, 256-bit**, mạnh hơn DES rất nhiều.
    - **Hiện nay là tiêu chuẩn mã hóa mạnh nhất được sử dụng rộng rãi**.

2. Hệ mật mã bất đối xứng (Asymmetric Cryptosystems)

Hệ mật mã bất đối xứng sử dụng **cặp khóa công khai – khóa bí mật**. Điều này giúp **giải quyết vấn đề phân phối khóa** nhưng **tốc độ chậm hơn** so với mã hóa đối xứng.

Các thuật toán quan trọng:

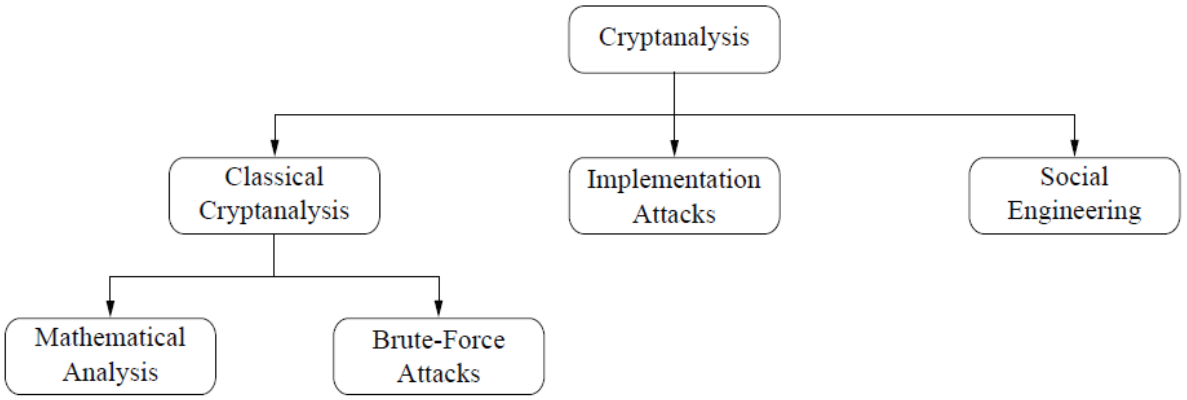
1. **1976 – Thuật toán Diffie-Hellman**
    - **Hệ thống trao đổi khóa đầu tiên**, giúp hai bên **chia sẻ khóa bí mật mà không cần gửi trực tiếp khóa**.
    - **Cơ sở của các giao thức mã hóa hiện đại như TLS/SSL**.
2. **1977 – RSA (Rivest-Shamir-Adleman)**
    - **Thuật toán mã hóa khóa công khai phổ biến nhất**, dựa trên **độ khó của bài toán phân tích thừa số nguyên tố**.
    - **Dùng trong SSL/TLS, VPN, chứng chỉ số**.
    - **Có thể bị đe dọa bởi máy tính lượng tử trong tương lai**.
3. **1985 – ElGamal Cryptosystem**
    - **Dựa trên bài toán logarit rời rạc**.
    - **Dùng trong chữ ký số DSA và hệ thống bảo mật GPG**.
4. **ГОСТ Р 34.10-2012**
    - **Tiêu chuẩn mật mã khóa công khai của Nga**, tương tự **ECDSA (Elliptic Curve Digital Signature Algorithm)**.
    - **Dùng trong chữ ký số và xác thực**.

3. Kết luận

- **Hệ mật mã đối xứng (AES, DES, GOST, IDEA)** phù hợp cho **mã hóa nhanh, nhưng cần cơ chế phân phối khóa**.
- **Hệ mật mã bất đối xứng (RSA, Diffie-Hellman, ElGamal)** phù hợp cho **bảo mật khóa, nhưng chậm hơn**.
- **Hệ mật mã hiện đại thường kết hợp cả hai phương pháp**, ví dụ:
    - **Dùng RSA/Diffie-Hellman để trao đổi khóa**.
    - **Dùng AES để mã hóa dữ liệu thực tế**.
- **Mật mã hậu lượng tử (Post-Quantum Cryptography) đang được phát triển để thay thế RSA/DH do nguy cơ từ máy tính lượng tử**.

"Криптоанализ"

(Phân tích mật mã - Cryptanalysis)



1. Định nghĩa về phân tích mật mã

- **Cryptanalysis (Phân tích mật mã)** là **quá trình nghiên cứu và tìm cách phá vỡ các hệ thống mã hóa**.
- Mục tiêu của phân tích mật mã là **khôi phục thông tin ban đầu mà không cần có khóa giải mã hợp lệ**.

2. Phân loại các phương pháp phân tích mật mã

Dựa theo sơ đồ trong hình, phân tích mật mã được chia thành **ba nhóm chính**:

A) Classical Cryptanalysis (Phân tích mật mã cổ điển)

1. **Mathematical Analysis (Phân tích toán học)**
    - Dựa trên các nguyên lý toán học để **tìm ra lỗ hổng trong thuật toán mã hóa**.
    - **Ví dụ**:
        - **Phân tích tần suất** (Frequency Analysis) – phá mã đơn bảng (Caesar, Vigenère).
        - **Tấn công logarit rời rạc** – dùng để phá RSA, Diffie-Hellman.
        - **Tấn công dựa trên phương trình đồng dư** – tìm khóa bằng cách giải hệ phương trình.
2. **Brute-Force Attacks (Tấn công vét cạn)**
    - **Thử tất cả các khóa có thể có** cho đến khi tìm được khóa đúng.
    - **Hiệu quả phụ thuộc vào độ dài khóa**:
        - **DES (56-bit)** có thể bị phá trong **vài giờ**.
        - **AES-256-bit** là **không thể phá bằng brute-force**.

B) Implementation Attacks (Tấn công vào cách triển khai)

- Không nhằm vào thuật toán mà **tấn công vào cách hệ thống thực hiện mã hóa**.
- **Ví dụ**:
    - **Tấn công kênh bên (Side-Channel Attacks)**: Dựa vào **thời gian thực thi, mức tiêu thụ điện năng, hoặc tín hiệu điện từ** để đoán khóa.
    - **Tấn công lỗi phần cứng (Fault Injection Attacks)**: Gây lỗi trong quá trình mã hóa để tìm thông tin về khóa bí mật.
    - **Tấn công dựa trên thời gian thực thi (Timing Attacks)**: Nếu thuật toán mất thời gian khác nhau để xử lý các khóa khác nhau, có thể khai thác để tìm khóa.

C) Social Engineering (Khai thác con người)

- Không tấn công vào thuật toán hay hệ thống, mà **tấn công vào con người**.
- **Ví dụ**:
    - **Phishing (Lừa đảo qua email, tin nhắn)** để lấy mật khẩu hoặc khóa.
    - **Shoulder Surfing (Quan sát người dùng nhập mật khẩu)**.
    - **Tấn công qua giả mạo (Impersonation)** – Giả mạo nhân viên IT để lấy thông tin bảo mật.

3. Kết luận

- **Phân tích mật mã không chỉ là tìm lỗ hổng toán học, mà còn bao gồm tấn công vào cách triển khai và con người**.
- **Mật mã hiện đại kết hợp bảo mật toán học với bảo vệ hệ thống trước tấn công kênh bên và tấn công tâm lý**.
- **Hệ thống an toàn nhất là hệ thống có cả mã hóa mạnh và các biện pháp bảo vệ chống lại các tấn công thực tế**.

"Базовые ограничения при использовании шифрования"

(Những hạn chế cơ bản khi sử dụng mã hóa)

1. Tổng quan về bảng so sánh

Bảng này phân tích **ba thành phần quan trọng trong hệ thống mã hóa**:

1. **Hệ thống nhận dạng và xác thực (Identification & Authentication Systems)**
2. **Hệ thống mã hóa bất đối xứng (Asymmetric Cryptosystems)**
3. **Hệ thống mã hóa đối xứng (Symmetric Cryptosystems)**

Mỗi hệ thống được đánh giá theo **chức năng chính, hạn chế khi sử dụng, và ví dụ tiêu biểu**.

2. So sánh chi tiết các hệ thống mã hóa

Tiêu chí so sánh	Hệ thống nhận dạng & xác thực	Hệ mật mã bất đối xứng	Hệ mật mã đối xứng
Chức năng chính	Tạo kênh xác thực cho mã hóa bất đối xứng.	Truyền, phân phối và đồng thuận về khóa mật mã.	Mã hóa/giải mã dữ liệu chính.
Hạn chế	Có thể có lỗi xác thực, sai sót trong lưu trữ dữ liệu xác thực.	Chỉ hoạt động an toàn khi có **kênh xác thực** đáng tin cậy.	Chỉ hoạt động nếu khóa bí mật **được chia sẻ an toàn** giữa các bên.
Ví dụ	Xác thực bằng **mật khẩu, token, sinh trắc học**.	**Diffie-Hellman (DH), RSA, ElGamal**.	**AES, GOST 34.12, IDEA**.

3. Phân tích chi tiết từng hệ thống

A) Hệ thống nhận dạng và xác thực

- **Vai trò**: Giúp xác minh danh tính người dùng và đảm bảo rằng **chỉ người dùng hợp lệ mới có quyền truy cập vào hệ thống mã hóa**.
- **Ví dụ**:
  - **Xác thực mật khẩu**: Để đảm bảo chỉ người dùng hợp pháp mới truy cập.
  - **Token bảo mật** (ví dụ: OTP, thẻ bảo mật).
  - **Sinh trắc học** (ví dụ: vân tay, nhận diện khuôn mặt).
- **Hạn chế**:
  - Có thể xảy ra **lỗi xác thực**, dẫn đến truy cập trái phép hoặc mất thông tin.
  - **Dữ liệu xác thực cần được lưu trữ an toàn**, nếu không sẽ bị tấn công (ví dụ: đánh cắp mật khẩu).

B) Hệ thống mã hóa bất đối xứng

- **Vai trò**: Dùng để **bảo mật truyền thông và phân phối khóa**, giúp thiết lập **kênh an toàn** giữa hai bên mà không cần chia sẻ khóa trước đó.
- **Ví dụ**:
  - **Diffie-Hellman (DH)**: Dùng để trao đổi khóa bí mật mà không cần truyền trực tiếp.
  - **RSA**: Mã hóa bằng khóa công khai, giải mã bằng khóa bí mật.
  - **ElGamal**: Dùng trong chữ ký số và bảo mật khóa.
- **Hạn chế**:
  - **Tốc độ chậm hơn** so với mã hóa đối xứng.
  - **Chỉ an toàn nếu được triển khai trong kênh xác thực tin cậy** (nếu không, kẻ tấn công có thể giả mạo khóa công khai).

C) Hệ thống mã hóa đối xứng

- **Vai trò**: Dùng để **mã hóa và giải mã dữ liệu với cùng một khóa bí mật**.
- **Ví dụ**:

    - **AES (Advanced Encryption Standard)**: Mã hóa dữ liệu nhanh, mạnh mẽ.
    - **GOST 34.12**: Tiêu chuẩn mã hóa của Nga, tương tự AES.
    - **IDEA (International Data Encryption Algorithm)**: Dùng trong PGP (Pretty Good Privacy).
- **Hạn chế**:
    - **Cần một cơ chế an toàn để chia sẻ khóa bí mật giữa các bên**.
    - Nếu khóa bị lộ, **toàn bộ dữ liệu có thể bị giải mã**.

4. Kết luận

- **Không có hệ thống mã hóa nào hoàn hảo, cần kết hợp nhiều phương pháp để bảo mật tối ưu**.
- **Hệ thống xác thực đóng vai trò quan trọng trong bảo vệ truy cập**.
- **Mã hóa bất đối xứng phù hợp để trao đổi khóa, trong khi mã hóa đối xứng phù hợp để bảo vệ dữ liệu lớn**.
- **Hệ thống bảo mật hiện đại thường kết hợp cả ba loại**:
    - **Xác thực người dùng** (mật khẩu, sinh trắc học).
    - **Trao đổi khóa bảo mật bằng RSA hoặc Diffie-Hellman**.
    - **Mã hóa dữ liệu bằng AES hoặc GOST**.

"Базовые ограничения при использовании методов обеспечения целостности"

(Những hạn chế cơ bản khi sử dụng các phương pháp đảm bảo tính toàn vẹn)

1. Tổng quan về bảng so sánh

Bảng so sánh ba phương pháp chính để **đảm bảo tính toàn vẹn của dữ liệu**:

1. **Методы помехоустойчивого кодирования** (Mã hóa chống nhiễu - Error-Correcting Codes)
2. **Хеширование** (Hàm băm - Hashing)
3. **Цифровая подпись** (Chữ ký số - Digital Signature)

2. So sánh chi tiết các phương pháp

Tiêu chí so sánh	Mã hóa chống nhiễu	Hàm băm	Chữ ký số
Chức năng chính	Phát hiện và sửa lỗi khi truyền, xử lý và lưu trữ dữ liệu.	Chỉ phát hiện lỗi, không sửa được.	Phát hiện lỗi, xác thực nguồn gốc và tính xác thực của dữ liệu.

Hạn chế khi sử dụng	Chỉ có hiệu quả nếu lỗi xuất hiện ngẫu nhiên (không phải lỗi cố ý từ kẻ tấn công).	Có thể bị tấn công (collision attack) nếu thuật toán băm yếu.	Cần khóa bí mật để xác thực, yêu cầu nhiều tài nguyên hơn.
Hiệu suất xử lý (tốc độ)	**Cao**	**Trung bình**	**Thấp**
Ví dụ tiêu biểu	**CRC, mã Hamming, mã BCH**	**MD5, SHA-1, SHA-256**	**DSA, ECDSA, ElGamal Signature**

3. Phân tích từng phương pháp

A) Mã hóa chống nhiễu (Error-Correcting Codes - ECC)

- **Cách hoạt động**:
  - Thêm **dữ liệu dư thừa vào thông điệp gốc** để phát hiện và sửa lỗi khi truyền dữ liệu.
  - **Ví dụ**: Khi gửi dữ liệu qua mạng, mã CRC có thể kiểm tra xem dữ liệu có bị thay đổi hay không.
- **Ưu điểm**:
  - **Có thể sửa lỗi**, không chỉ phát hiện lỗi.
  - **Tốc độ nhanh**, phù hợp cho truyền thông tin trong thời gian thực.
- **Nhược điểm**:
  - Chỉ hiệu quả với **lỗi ngẫu nhiên**, không chống được **tấn công có chủ đích**.
  - Không thể **xác thực nguồn gốc dữ liệu**.

B) Hàm băm (Hashing)

- **Cách hoạt động**:
  - Biến đổi dữ liệu đầu vào thành một **giá trị băm cố định**, giúp kiểm tra xem dữ liệu có bị thay đổi không.
  - Nếu dữ liệu thay đổi dù chỉ một chút, giá trị băm cũng thay đổi hoàn toàn.
- **Ưu điểm**:
  - **Có thể kiểm tra tính toàn vẹn dữ liệu** nhanh chóng.
  - **Bảo mật cao hơn mã hóa chống nhiễu**, vì thay đổi nhỏ trong dữ liệu làm thay đổi hoàn toàn giá trị băm.
- **Nhược điểm**:
  - **Không thể sửa lỗi** như mã hóa chống nhiễu.
  - **Một số thuật toán băm cũ (MD5, SHA-1) có thể bị tấn công va chạm (collision attack)**.

C) Chữ ký số (Digital Signature)

- **Cách hoạt động**:
  - Dùng **khóa bí mật để ký dữ liệu**, tạo ra một giá trị xác thực duy nhất.
  - Bất kỳ ai cũng có thể **kiểm tra chữ ký** bằng khóa công khai.
  - Nếu dữ liệu thay đổi, **chữ ký sẽ không hợp lệ**.

- **Ưu điểm**:
  - **Không chỉ kiểm tra tính toàn vẹn, mà còn xác thực người gửi**.
  - **Chống giả mạo tốt hơn hàm băm**.
- **Nhược điểm**:
  - **Cần hệ thống quản lý khóa công khai (PKI)**.
  - **Tốc độ chậm hơn do cần tính toán với số lớn**.

4. Kết luận

- **Mã hóa chống nhiễu (CRC, Hamming) phù hợp cho phát hiện lỗi trong truyền dữ liệu, nhưng không thể xác thực nguồn gốc dữ liệu.**
- **Hàm băm (MD5, SHA-256) mạnh hơn trong việc phát hiện thay đổi dữ liệu, nhưng không xác thực được người gửi.**
- **Chữ ký số (DSA, ECDSA) cung cấp mức bảo mật cao nhất, vừa kiểm tra tính toàn vẹn, vừa xác thực người gửi, nhưng chậm hơn các phương pháp khác.**

🔹 **Hệ thống bảo mật hiện đại thường kết hợp cả ba phương pháp để đảm bảo an toàn dữ liệu tối đa.**