TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

AN TOÀN THÔNG TIN- INFORMATION SECURITY

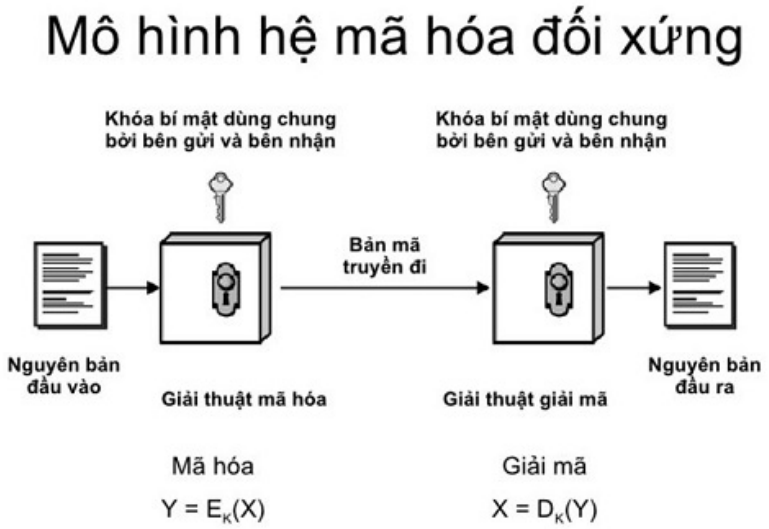
CHƯƠNG 3
KỸ THUẬT MÃ HOÁ
BÀI 7
MÃ HOÁ HIỆN ĐẠI – MÃ HÓA ĐỐI XỨNG

Giảng viên: TS. Trần Thế Vinh

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ ĐỐI XỨNG

Khái niệm:
Mã hóa khóa đối xứng là một loại sơ đồ hệ thống mã hóa trong đó một khóa giống nhau sẽ được dùng cho cả mã hóa và giải mã dữ liệu. Mã hóa đối xứng là giải pháp được sử dụng phổ biến nhất hiện nay.



MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ ĐỐI XỨNG

Phân loại mã hóa đối xứng:

Dựa trên cách các chuỗi nhị phân được xử lý, một sơ đồ mã hóa đối xứng được phân loại thành 2 loại chính:

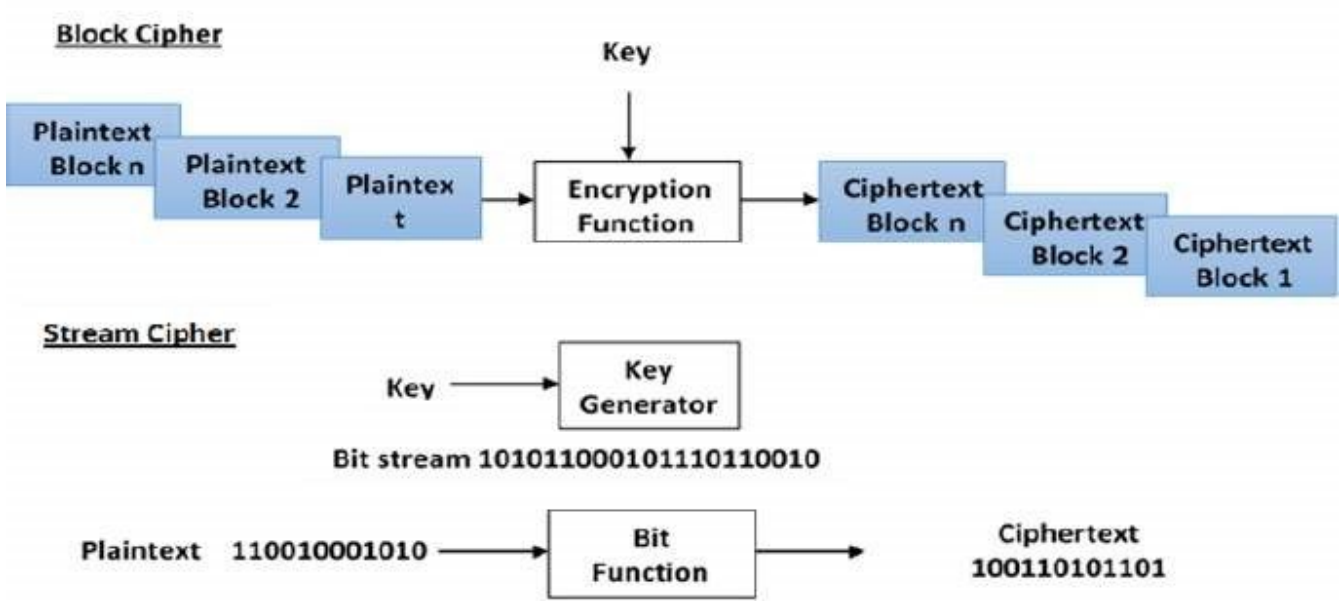
- Mã hóa Khối (Block Ciphers)
- Mã hóa Dòng (Stream Ciphers)

Mã hóa Dòng mã hóa các chữ số (thường là byte) hoặc chữ cái (trong mật mã thay thế) của một tin nhắn tại một thời điểm. Hay nói cách khác bản rõ được xử lý từng bit một, nghĩa là một bit của bản rõ được lấy và thực hiện một loạt các thao tác trên nó để tạo ra một bit bản mã. *Về mặt kỹ thuật, mật mã Dòng là mật mã Khối với kích thước khối là 1 bit.*

Mã hóa Khối lấy một số bit và mã hóa chúng trong một đơn vị duy nhất. Nếu số bit bản rõ không đủ thì đệm bản rõ để đạt được bội số của kích thước khối. Hay nói cách khác bản rõ nhị phân đơn giản được xử lý theo khối bit tại một thời điểm, tức là một khối bit bản rõ được chọn, và thực hiện một loạt thao tác trên khối này để tạo ra một khối bit bản mã. Số bit trong một khối là cố định.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI



Với các thông điệp có kích thước lớn và để tăng cường độ an toàn, thông điệp thường được chia ra thành các phần có kích thước bằng nhau, sau đó áp dụng thuật toán mã hóa trên từng phần này và ghép các phần đã mã hóa để tạo bản mã ở bên người gửi.

Ở bên người nhận, bản mã cũng được chia thành các phần có kích thước bằng nhau, sau đó thực hiện giải mã từng phần và ghép các phần đã giải mã để tạo bản rõ.

Mã hóa dòng và mã hóa khối là hai phương pháp mã hóa, trong đó mỗi phương pháp thực hiện việc chia dữ liệu để mã hóa theo cách khác nhau.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ DÒNG:

Mã hóa dòng (Stream cipher) được đặc trưng bởi thực tế là chúng mã hóa thông tin từng bit một cho mỗi chu kỳ mã hóa. Xem xét rằng trong số các hoạt động có bit chỉ có hai phép đảo ngược là tổng theo modulo 2 và XOR, việc lựa chọn nguyên tắc mã hóa là hiển nhiên - các bit của bản rõ phải bao gồm các bit của chuỗi khóa bằng cách sử dụng toán tử $\oplus$:

$$c_i = m_i \oplus k_i$$

Giải mã bằng cách tương tự:

$$m_i = c_i \oplus k_i$$

Cùng với modulo 2, có thể tính: $k_i = c_i \oplus m_i$

Do đó, khả năng bảo vệ của các mật mã hiện tại phụ thuộc hoàn toàn vào chất lượng của trình tạo dòng khoá. Rõ ràng là nếu luồng khóa chỉ bao gồm các số 0 ở đầu, thì bản mã sẽ là một bản sao chính xác của bản rõ. Thông thường, biểu thị dòng khóa của mật mã hiện tại bằng chữ cái Hy Lạp γ (gamma), do đó những mật mã như vậy được gọi là mật mã gammation. Chúng ta hãy xem xét các phương pháp chính của việc hình thành γ trong mật mã dòng hiện đại.

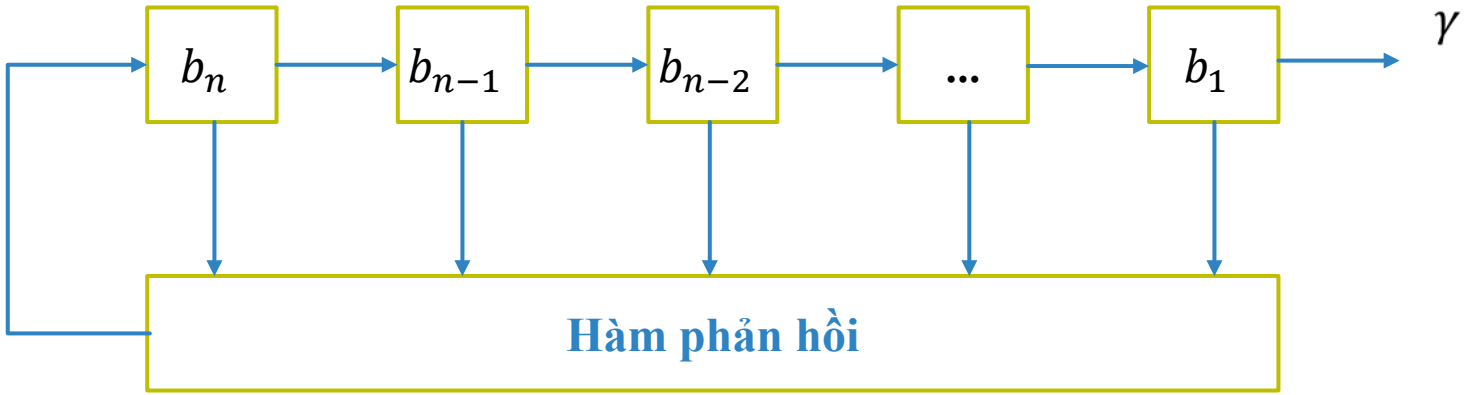
MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ DÒNG:

Thanh đăng ký dịch chuyển với phản hồi rất phổ biến để giải quyết mục đích này. Nó là một chuỗi các bit, được dịch chuyển sang phải 1 chữ số ở mỗi chu kỳ mã hóa, trong khi đầu ra từ bit ngoài cùng bên phải là đầu ra của bộ tạo và đầu vào của bit ngoài cùng bên trái được nạp bằng một giá trị được tính toán dưới dạng một số hàm của thanh ghi các bit riêng lẻ,

Khóa mã hóa của mật mã hiện tại được nhập vào sổ đăng ký trước khi bắt đầu tạo.

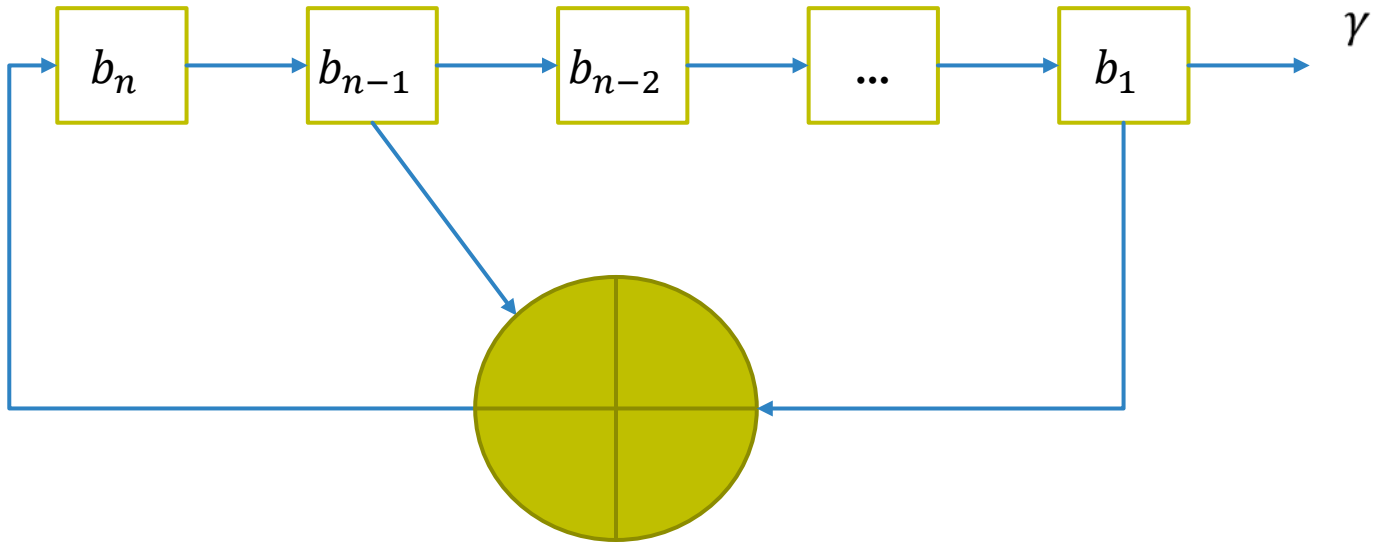


MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ DÒNG:

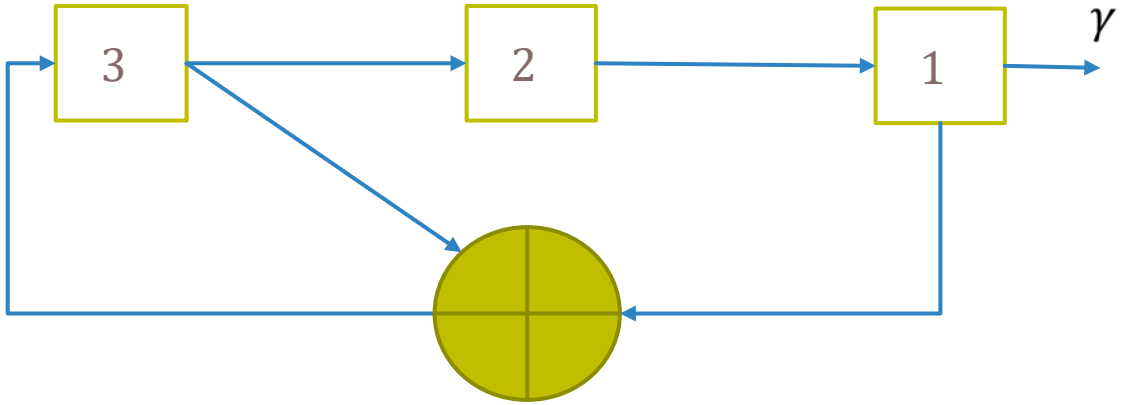
Cách đơn giản nhất để hình thành phản hồi là tính tổng theo modulo 2 của các bit riêng lẻ của thanh ghi. Trong hình dưới đây hiển thị một thanh ghi dịch chuyển với phản hồi tuyến tính (**linear-feedback shift register (LFSR)**).



MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ DÒNG:



Giá trị ban đầu được nhập là 010 và giá trị đầu ra ở bảng dưới:

Số chu kỳ	Giá trị bit của thanh đăng ký			Bit gamma
	3	2	1	
Khởi đầu	0	1	0	-
1	0	0	1	0
2	1	0	0	1
3	1	1	0	0
4	1	1	1	0
5	0	1	1	1
6	1	0	1	1
7	0	1	0	1
8	0	0	1	0

Bảng này cho thấy trạng thái của LFSR được lặp lại sau 7 chu kỳ (trạng thái ban đầu trùng với trạng thái của nó ở chu kỳ thứ 7). Lặp lại trạng thái của LFSR có nghĩa là gamma sẽ được lặp lại định kỳ. Sự lặp lại gamma làm giảm sức mạnh bảo mật của các mật mã hiện tại, cho phép nhà phân tích mật mã phân tích bản mã thu được bằng cách mã hóa trên cùng một gamma.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ DÒNG:

Do đó, khi thiết kế cấu trúc của LFSR(linear – feedback shift registers), vấn đề nảy sinh là đạt được thời gian lặp lại lớn nhất của LFSR. Đối với LFSR có độ dài n bit, chu kỳ tối đa là chu kỳ 2n-1 (trạng thái khi tất cả các bit bằng không là không thể chấp nhận được). Việc xây dựng LFSR của cấu trúc tối ưu theo quan điểm của thời kỳ lặp lại gamma có cơ sở toán học rõ ràng dưới dạng trường G_{2^n}. Cấu trúc của LFSR được mô tả bằng một đa thức có dạng:

$$b_n \cdot x^n + b_{n-1} \cdot x^{n-1} + b_{n-2} \cdot x^{n-2} + \cdots + b_2 \cdot x^2 + b_1 \cdot x^1 + 1 \quad (1)$$

Trong đó $b_i = 0$, nếu bit không liên quan đến phản hồi và $b_i = 1$, nếu có liên quan.

Vấn đề chính của LFSR là chúng dễ bị gây hại trước một cuộc tấn công dựa trên bản rõ đã biết. Ngay cả khi cấu trúc bên trong của LFSR là không xác định, một nhà phân tích mật mã sử dụng thuật toán Berlekamp-Massey trên 2N bit đã biết của bản rõ và bản mã tương ứng có cơ hội xây dựng một LFSR, tạo ra một chuỗi tương tự (vấn đề về độ phức tạp tuyến tính của LFSR).

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ KHỐI:

Mã hóa đối xứng khối liên quan đến việc thực hiện chuyển đổi mật mã trên các khối văn bản thuần túy có kích thước cố định (32, 64, 128, 256 bit). Do đó, một mật mã khối có thể được biểu diễn dưới dạng mật mã thay thế trên một bảng chữ cái rất lớn, ví dụ, có 2^{64} ký tự cho một khối 64 bit.

Bất kỳ mật mã thay thế nào cũng có thể được biểu diễn dưới dạng bảng tương ứng giữa các ký tự đầu vào và đầu ra. Tuy nhiên, kích thước của một bảng như vậy cho các mật mã khối hiện đại sẽ rất lớn (cần 2^{70} bit bộ nhớ cho một thuật toán có khối 64 bit chỉ cho một khóa mã hóa) đến nỗi việc biểu diễn sự tương ứng của dữ liệu đầu vào và đầu ra chỉ được mô tả hiệu quả bằng thuật toán.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

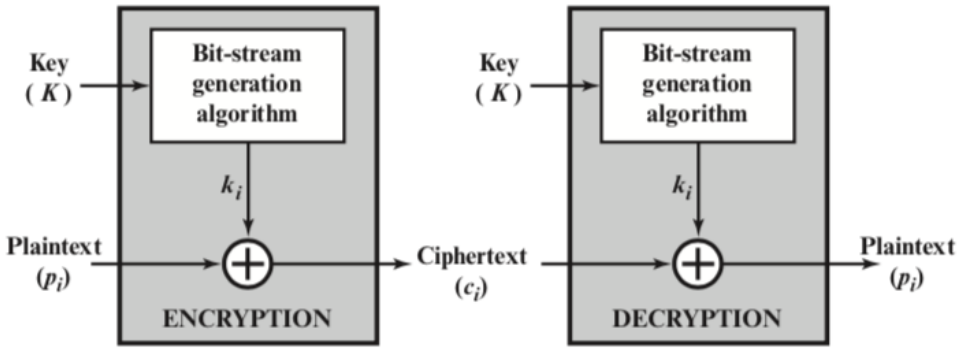
MÃ HOÁ KHỐI:

Nói chung, thuật toán mã hóa khối được mô tả như sau: Một khối văn bản đơn thuần X được chuyển đổi thành một khối mật mã Y có cùng kích thước bằng cách sử dụng một số Khóa mã hóa, có Khóa được thể hiện là:

$$Y = Encrypt(X, Key)$$

Quy trình giải mã thực hiện chuyển đổi ngược lại bằng cách sử dụng cùng một khóa:

$$X = Decrypt(Y, Key)$$



MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ KHỐI:

Các phép biến đổi Mã hóa và Giải mã diễn giải khối văn bản rõ ràng và khối văn bản được mã hóa dưới dạng số nguyên và thực hiện một số phép toán số học hoặc logic trên chúng, nhiệm vụ chính là "xáo trộn" các bit của khối bản rõ với nhau, cũng như kết nối chúng với các bit của khóa mã hóa đã sử dụng để hình thành một khối bản mã.

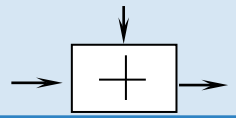
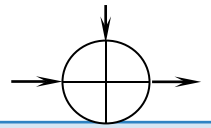
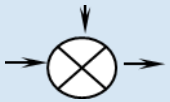
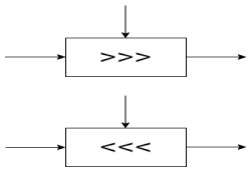
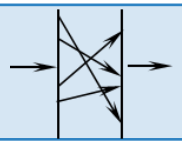
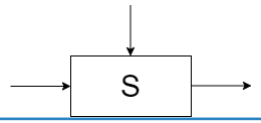
Đồng thời, đối với mật mã khối loại thông tin được biến đổi là không có ý nghĩa, nó chia nhỏ thành các khối có kích thước nhất định, diễn giải các khối này dưới dạng số nguyên trong phạm vi $[0,..,2^{k-1}]$, trong đó k là kích thước của khối và thực hiện một loạt các phép biến đổi trên các số này tuân theo các phương pháp thuật toán.

Các phép toán có thể đảo ngược đóng một vai trò đặc biệt trong việc thực hiện các phép biến đổi mật mã thành mật mã khối, vì chỉ việc sử dụng các phép toán này mới cung cấp khả năng đưa vào (khả năng đảo ngược) của tất cả các phép biến đổi mật mã. Các phép toán đảo ngược được sử dụng thường xuyên nhất trong mật mã khối hiện đại được liệt kê trong Bảng:

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ KHỐI:

Tên	Ký hiệu hình	Công thức chuyển đổi	Chuyển dổi ngược lại
Phép cộng		$X' = X + V$	Phép trừ
Tổng theo modulo 2		$X' = X \oplus V$	Tự động trả về (Automatic return)
Nhân theo modulo $2^N + 1$ (N *là kích thước khối*)		$X' = (X \cdot V) mod\ (2^N + 1)$	Nhân tử chung có thể tìm được bằng thuật toán Euclid
Dịch chuyển theo chu kỳ sang phải/trái		$X' = X\ ROR\ V$ $X' = X\ ROL\ V$	Chuyển dịch tuần hoàn theo hướng ngược lại
Hoán vị bit			
Bảng thay thế		$X' = SBox(X)$	Thay thế ngược lại

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

MÃ HOÁ KHỐI:

Đặc biệt chú ý trong Bảng trên cần được chú ý đến các hoạt động của phép nhân và thay thế bảng, vì bằng cách đưa tính phi tuyến tính vào chuyển đổi mật mã chung, làm cho thuật toán có khả năng chống phân tích mật mã tuyến tính cao hơn.

Một trong những nguyên tắc chính của việc xây dựng cấu trúc của các thuật toán mã hoá hiện đại là nguyên tắc lặp lại. Ý tưởng của nó là lặp đi lặp lại, bao gồm nhiều chu kỳ (hoặc vòng), xử lý một khối văn bản thuần túy bằng cách sử dụng một khóa đặc biệt của vòng trên mỗi chu kỳ, khóa này được tính toán trên cơ sở khóa mã hóa.

Số lượng chu kỳ có thể thay đổi vì lý do kháng mật mã và hiệu quả triển khai thuật toán: tăng số chu kỳ dẫn đến tăng độ mạnh của mật mã, nhưng làm tăng thời gian mã hóa và tiêu tốn tài nguyên máy tính. Các cấu trúc tuần hoàn như vậy thường được gọi là mạng lưới và hầu hết các mật mã khối hiện đại được xây dựng bằng cách sử dụng kiến trúc mạng lưới

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Mã hóa Khối (Block Ciphers):

Mật mã khối lấy một khối bit bản rõ và tạo ra một khối bit bản mã, thường có cùng kích thước. Kích thước của khối được cố định trong thuật toán. Việc lựa chọn kích thước khối không ảnh hưởng trực tiếp đến độ mạnh của lược đồ mã hóa. Độ mạnh của mật mã phụ thuộc vào độ dài của khóa.

Kích thước khối:

Trên lý thuyết bất kỳ kích thước nào của khối đều được chấp nhận, nhưng có các quy tắc được ghi nhớ trong khi chọn kích thước khối:

Tránh kích thước khối quá nhỏ: - Giả sử kích thước khối là n bit. Sau đó, các kết hợp bản rõ có thể là 2^n. Nếu kẻ tấn công phát hiện ra các khối bản rõ tương ứng với một số khối bản mã đã gửi trước đó, thì kẻ tấn công có thể khởi chạy một kiểu "tấn công từ điển" bằng cách xây dựng một từ điển gồm các cặp bản rõ/mã được gửi bằng khóa mã hóa đó. Kích thước khối lớn hơn khiến cuộc tấn công trở nên khó khăn hơn vì từ điển cần phải nhiều hơn.

Không nên để kích thước khối quá lớn: - Với kích thước khối quá lớn, mật mã sẽ trở nên hoạt động kém hiệu quả. Những bản rõ như vậy cần được đệm trước khi được mã hóa.
Bội số của 8 bit: Kích thước khối nên là bột số của 8 vì nó dễ thực hiện vì hầu hết bộ xử lý máy tính xử lý dữ liệu theo bội số của 8 bit.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Sự khác biệt giữa Mật mã Khối và mật mã Dòng:

Mật mã Khối	Mật mã Dòng
Chuyển đổi bản rõ thành bản mã bằng cách lấy từng khối của bản rõ	Chuyển đổi bản rõ thành bản mã bằng cách lấy 1 byte bản rõ tại một thời điểm
Sử dụng 64 bit hoặc nhiều hơn	Sử dụng 8 bit (1 byte)
Mật mã khối không phức tạp	Mật mã dòng phức tạp hơn
Mật mã khối sử dụng sự xáo trộn và khuếch tán	Mật mã dòng chỉ sử dụng sự xáo trộn
Bản rõ được mã hóa ngược rất khó	Bản rõ được mã hóa ngược dễ dàng
Các chế độ thuật toán được sử dụng trong mật mã khối là ECB và CBC	Trong mật mã dòng là CFB và OFB
Mật mã khối hoạt động trên các kỹ thuật chuyển vị như kỹ thuật rail-fence, kỹ thuật chuyển vị cột, ..v..v..	Mật mã dòng hoạt động trên các kỹ thuật thay thế.
Mật mã khối chậm hơn so với mật mã dòng	Mật mã dòng nhanh hơn mật mã khối

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Các chế độ hoạt động của mã hóa Khối:

Đối với các ứng dụng và mục đích sử dụng khác nhau mà mã hóa Khối phân ra các chế độ hoạt động cho mã hóa Khối:

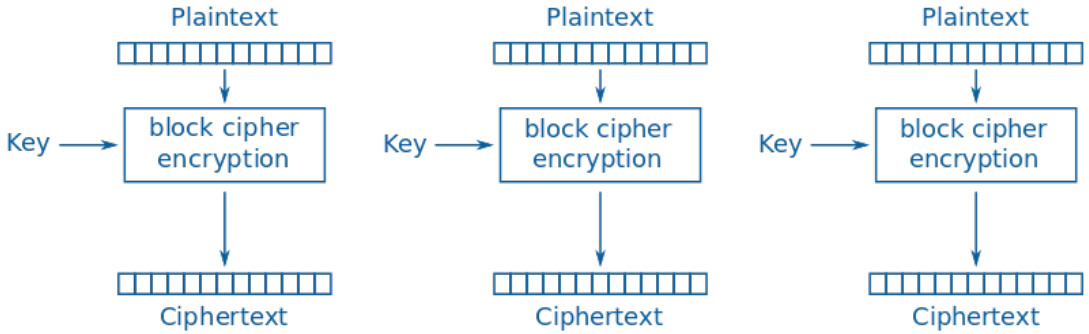
- Electronic Code Book (ECB)
- Cipher Block Chaining (CBC)
- Cipher Feedback Mode (CFB)
- Output Feedback Mode (OFB)
- Counter Mode (CTR)

MÃ HOÁ HIỆN ĐẠI

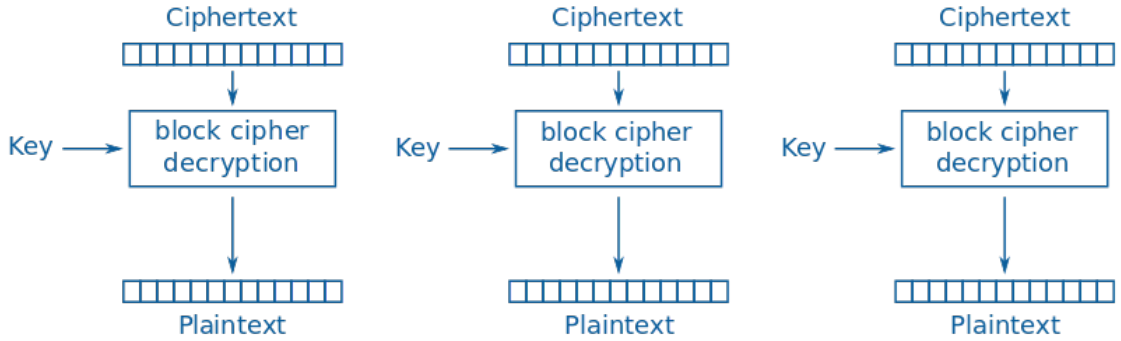
MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Electronic Code Book (ECB):

Chế độ này được gọi là chế độ thay thế đơn giản. Dễ dàng hơn vì mã hóa trực tiếp từng khối bản rõ đầu vào và đầu ra ở dạng các khối bản mã được mã hóa. Nghĩa là, nếu một thông báo có kích thước lớn hơn n bit, thì nó có thể được chia thành nhiều khối có kích thước n bit hoặc thêm bộ đệm và quy trình được lặp lại.



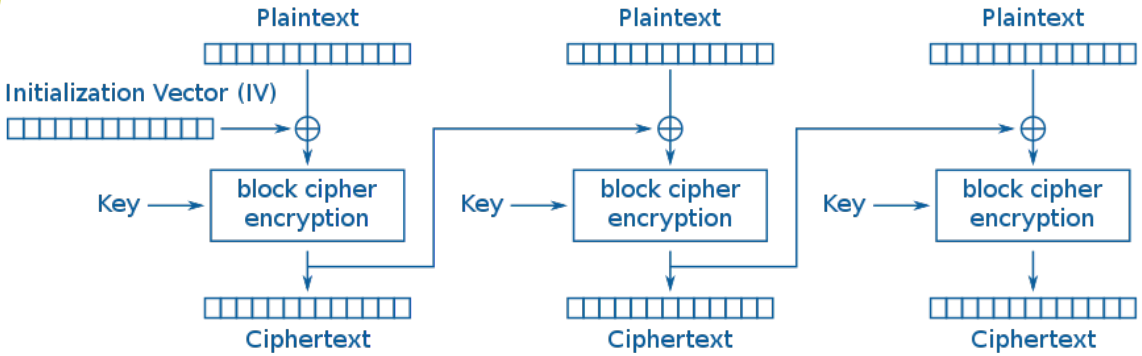
Electronic Codebook (ECB) mode encryption



Electronic Codebook (ECB) mode decryption

MÃ HOÁ HIỆN ĐẠI

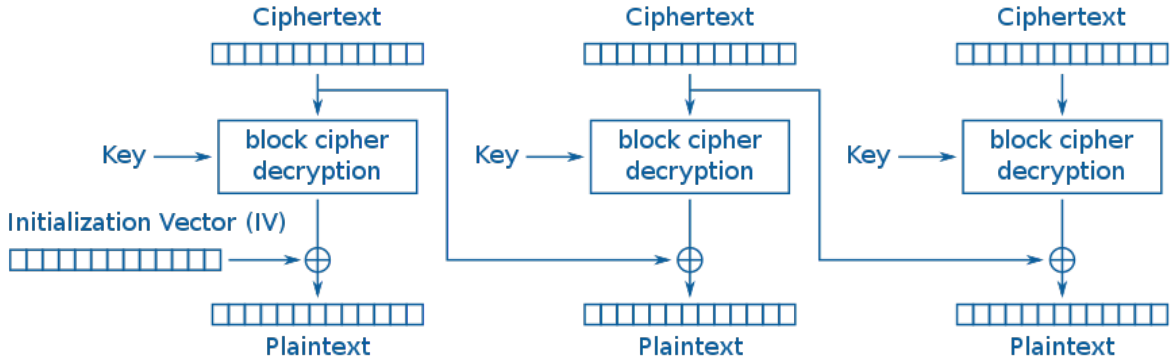
MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI



Cipher Block Chaining (CBC) mode encryption

Cipher Block Chaining (CBC):

CBC là một cải tiến được thực hiện trên ECB, do ECB dễ bị phá vỡ. Trong CBC, khối mật mã trước đó được cung cấp làm đầu vào cho thuật toán mã hóa tiếp theo sau khi XOR với khối bản rõ ban đầu. Hay nói cách khác, một khối mật mã được tạo ra bằng cách mã hóa đầu ra XOR của khối mật mã trước đó và khối bản rõ hiện tại



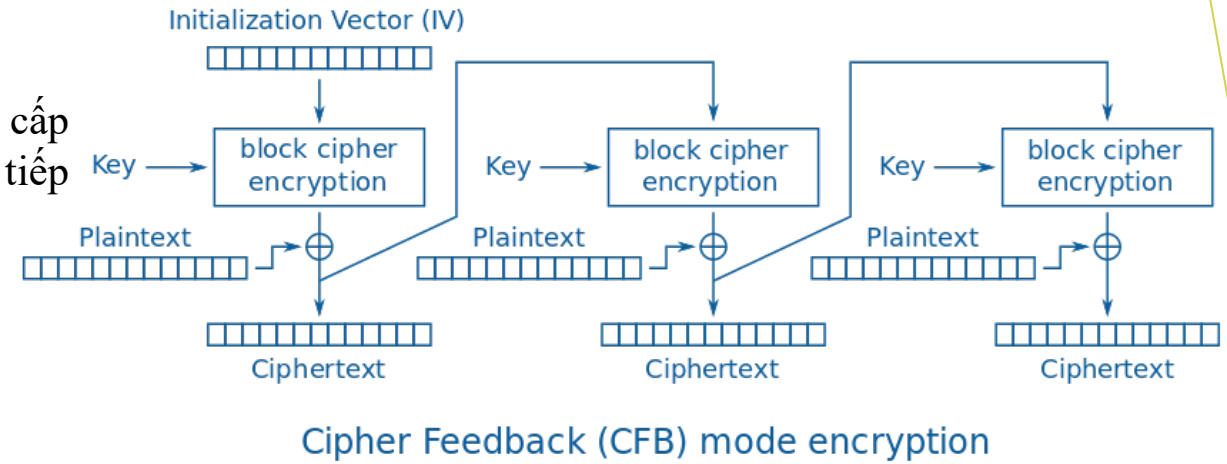
Cipher Block Chaining (CBC) mode decryption

MÃ HOÁ HIỆN ĐẠI

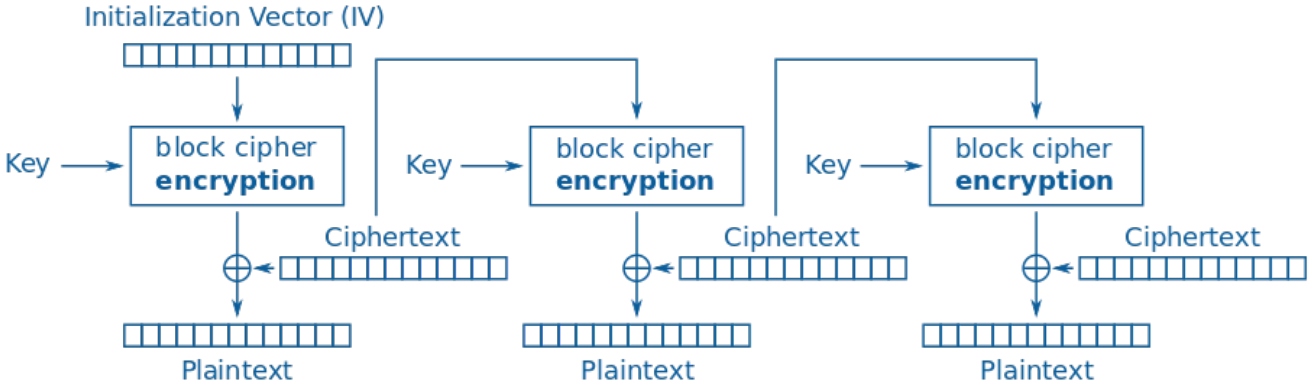
MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Cipher Feeback Mode(CFB):

Trong chế độ này, mật mã được cung cấp dưới dạng phản hồi cho khối mã hóa tiếp theo với một thông số kỹ thuật mới.



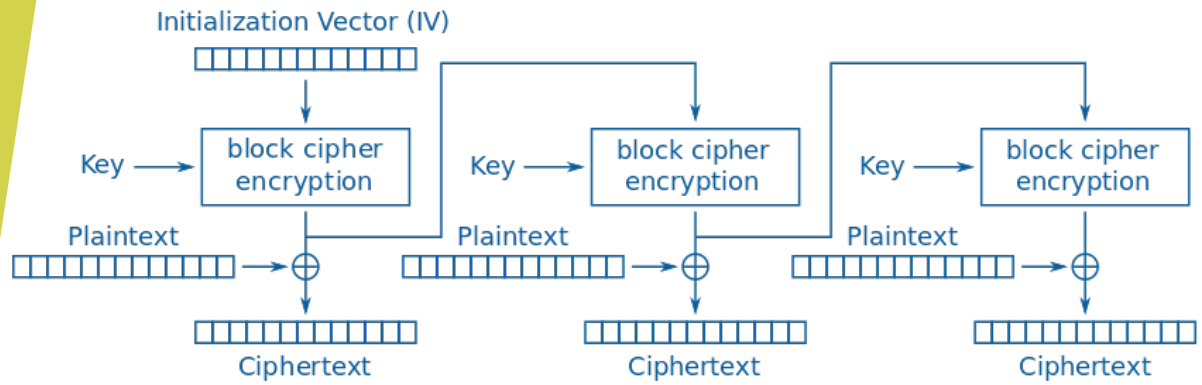
Cipher Feedback (CFB) mode encryption



Cipher Feedback (CFB) mode decryption

MÃ HOÁ HIỆN ĐẠI

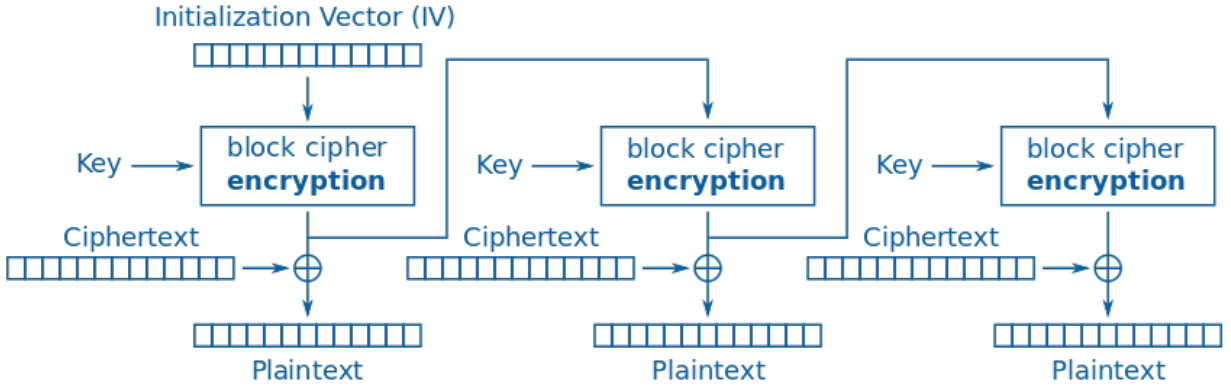
MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI



Output Feedback (OFB) mode encryption

Output Feedback Mode (OFB):

Chế độ này tuân theo qui trình gần giống với chế độ CFB ngoại trừ việc nó gửi đầu ra được mã hóa dưới dạng phản hồi thay vì mật mã thực tế là đầu ra XOR.



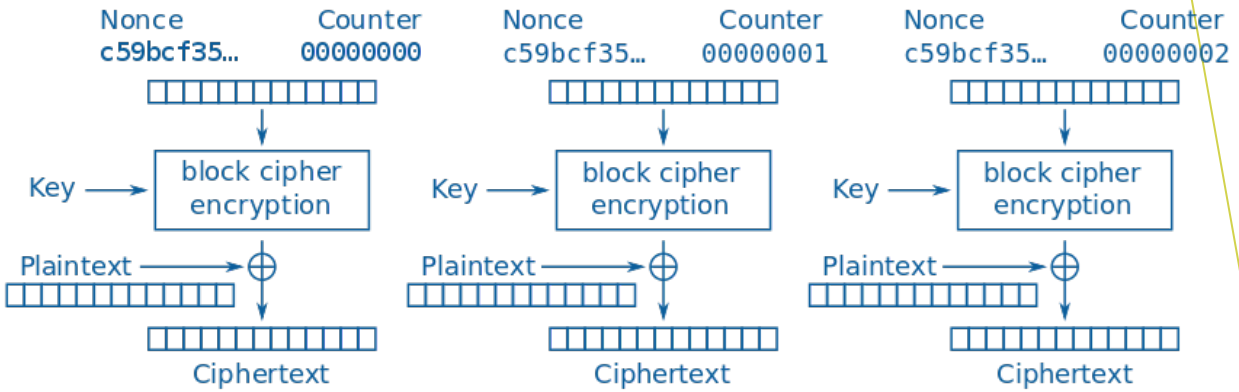
Output Feedback (OFB) mode decryption

MÃ HOÁ HIỆN ĐẠI

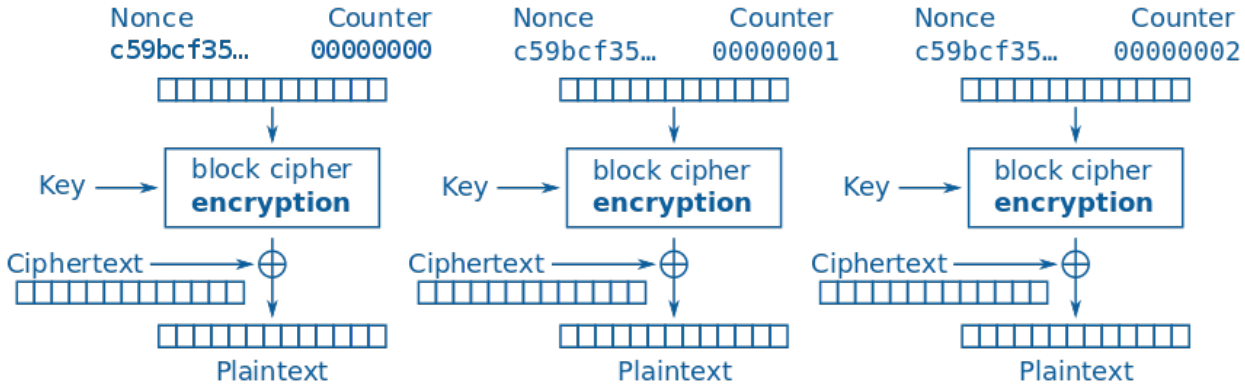
MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Counter Mode (CTR):

Chế độ này triển khai mật mã khối dựa trên một bộ đếm đơn giản. Mỗi khi một giá trị khởi tạo ngược được mã hóa và được cung cấp làm đầu vào cho XOR với văn bản gốc dẫn đến khối mã hóa. Chế độ CTR độc lập với việc sử dụng phản hồi và do đó có thể được thực hiện song song



Counter (CTR) mode encryption



Counter (CTR) mode decryption

MÃ HOÁ HIỆN ĐẠI

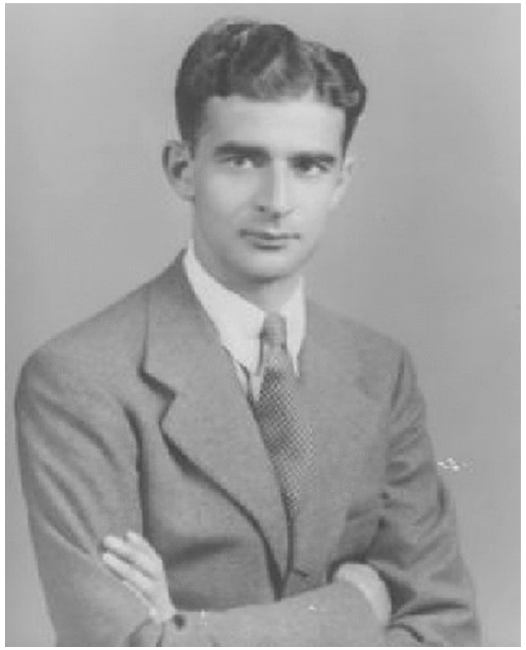
MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Khái niệm cấu trúc Feistel:

Khuếch tán ngụ ý rằng việc thay đổi bất kỳ dấu hiệu nào của bản rõ hoặc khóa ảnh hưởng đến một số lượng lớn các dấu hiệu của bản mã, điều này che giấu các thuộc tính thống kê của bản rõ và ngăn cản việc khôi phục khóa không xác định theo từng phần.

Mục đích của xáo trộn là làm cho mối quan hệ giữa khóa và bản mã càng phức tạp càng tốt. Một nhà phân tích mật mã dựa trên phân tích thống kê về văn bản được xáo trộn sẽ không nhận được bất kỳ lượng thông tin đáng kể nào về khóa đã sử dụng.

Việc sử dụng khuếch tán và xáo trộn riêng biệt mang lại sự ổn định cần thiết, một hệ thống mật mã đáng tin cậy khi sử dụng cả hai.



Horst Feistel (sinh 30/1/1915 – 14/11/1990) là một nhà mật mã học người Mỹ gốc Đức, người đã làm công việc thiết kế mật mã tại IBM, khởi xướng nghiên cứu mà đỉnh cao là sự phát triển củ DES vào những năm 1970. Cấu trúc được sử dụng trong DES được gọi là cấu trúc Feistel.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Nguyên tắc thiết kế mật mã khối:

Mật mã khối được xây dựng trong cấu trúc mật mã Feistel. Mật mã khối có số vòng (Round) và Khóa (Key) cụ thể để tạo bản mã. Để xác định mức độ phức tạp của một thuật toán, một vài nguyên tắc thiết kế sẽ được xem xét như sau:

1. Số vòng (round): Số lượng vòng được sư dụng phụ thuộc vào độ bảo mật mong muốn từ hệ thống. Số vòng nhiều hơn cung cấp hệ thống an toàn hơn. Nhưng đồng thời, nhiều vòng hơn thì quá trình mã hóa và giải mã chậm và không hiệu quả. Do đó, số lượng vòng trong các hệ thống phụ thuộc vào sự đánh đổi hiệu quả an ninh. (Trong DES có 16 round để đảm bảo an toàn hơn còn trong AES có 10 round(128 bit) làm cho thuật toán được an toàn).

2. Thiết kế hàm F: Phần cốt lõi của cấu trúc mật mã khối Feistel là Hàm vòng (Round Function). Độ phức tạp của phân tích mã (cryptanalysis) có thể bắt nguồn từ Hàm vòng tức là độ phức tạp ngày càng tăng đối với hàm round sẽ góp phần rất lớn vào sự gia tăng độ khó của hệ thống mật mã. *Để tăng độ phức tạp của round function, hiệu ứng tuyết lỡ cũng nằm trong hàm vòng, có nghĩa là bất kỳ 1 sự thay đổi nhỏ đơn lẽ nào trong bản rõ cũng sẽ tạo ra một đầu ra không chính xác.*

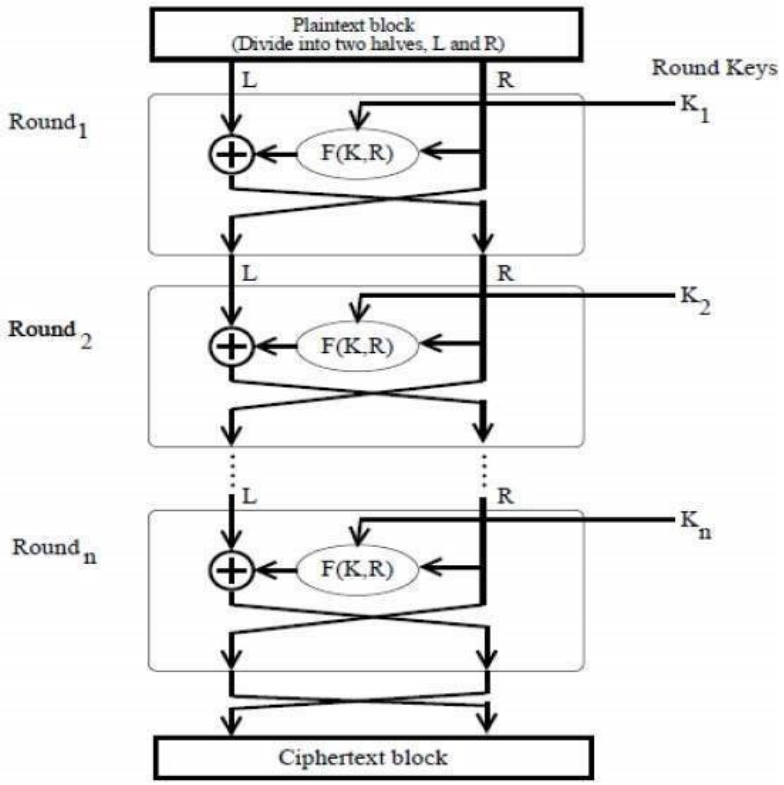
3. Thuật toán lịch trình khóa: Trong cấu trúc mật mã khối Feistel, mỗi vòng sẽ tạo ra một khóa con để tăng độ phức tạp của phân tích mật mã. Hiệu ứng tuyết lở làm cho việc lấy khóa con phức tạp hơn. Việc giải mã phải được thực hiện rất cẩn thận để có được đầu ra đúng vì hiệu ứng tuyết lở nằm trong đó.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

Cấu trúc mật mã khối Feistel:

Mã hóa Feistel không phải là một lược đồ cụ thể của mật mã khối. Nó là một mô hình thiết kế mà từ đó nhiều mật mã khối dựa trên mô hình này được tạo ra. DES là một trong những lược đồ mật mã được tạo ra từ mô hình cấu trúc Feistel.



Một hệ thống mật mã dựa trên cấu trúc mật mã Feistel sử dụng cùng 1 thuật toán cho cả mã hóa và giải mã.

Quá trình mã hóa:

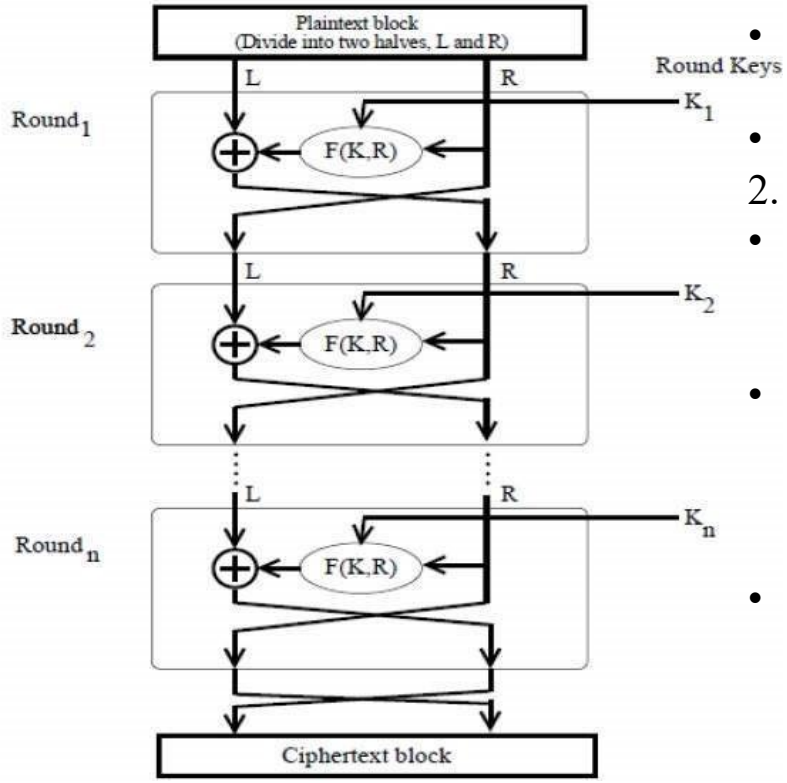
Quá trình mã hóa sử dụng cấu trúc Feistel bao gồm nhiều vòng xử lý bản rõ, mỗi vòng bao gồm một số bước "thay thế" theo sau là một bước hoán vị.

Bất kỳ một hệ thống mã hóa đều có 2 bước:

1. Sinh khóa
2. Xử lý mã hóa

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI



Việc hoán đổi cuối cùng của L và R trong bước cuối của mật mã Feistel là cần thiết. Nếu chúng ko được hoán đổi thì bản mã kết quả không thể giải mã bằng cùng một thuật toán.

1. Sinh khóa:

- Từ khóa ban đầu, tùy vào mỗi thuật toán cụ thể mà có các thuật toán sinh khóa khác nhau để tạo thành tập hợp các khóa con.
- Khóa con có độ dài bé hơn hoặc bằng khóa trạng thái (Khóa đầu vào).
- Khóa con dùng để sử dụng mã hóa theo mỗi vòng (round).

2. Xử lý mã hóa:

- Bước 1: Khối (bit) đầu vào được chia làm 2 nửa:
  - L (nửa bên trái)
  - R (nửa bên phải).
- Bước 2: Trong mỗi vòng:
  - $R_i \rightarrow L_{i+1}$,
  - $F_i(K_i, R_i) = (K_i, R_i)$,
  - $L_i \oplus F_i(K_i, R_i) = R_{i+1}$.
- Bước 3: Vòng cuối cùng thực hiện bước hoán vị đổi vị trí L và R cho nhau:
  - $R_n \rightarrow L_{n+1}$,
  - $F_n(K_n, R_n) = (K_n, R_n)$,
  - $L_n \oplus F_n(K_n, R_n) = R_{i+1}$
  - $R_{n+1} \rightarrow L_{n+1}$
  - $L_{n+1} \rightarrow R_{n+1}$

Ta có kết quả khối được mã hóa

Giải mã là quá trình ngược lại quá trình mã hóa.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ DÒNG VÀ MÃ HOÁ KHỐI

CẤU TRÚC FEISTEL

Mạng Feistel đã được chứng minh một cách đáng tin cậy là một sơ đồ mã hoá mạnh để thực hiện các phép biến đổi mật mã và nó có thể được tìm thấy trong hầu hết mọi mật mã khối hiện đại.

Ưu điểm của mạng Feistel là:

- Đơn giản hóa việc triển khai phần cứng trên cơ sở mã hoá hiện đại.
- Đơn giản hóa việc triển khai phần mềm do thực tế là một phần quan trọng của các chức năng được hỗ trợ ở cấp độ phần cứng trong các máy tính hiện đại (ví dụ: tổng theo modulo 2, v.v.)
- Nghiên cứu tốt các thuật toán dựa trên mạng Feistel

Tuy nhiên, cũng có một nhược điểm: chỉ một nửa khối đầu vào được mã hóa trong một vòng.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Sơ lược về thuật toán AES:

Thuật toán Rijndael (AES) năm 2000 đã chiến thắng trong cuộc cạnh tranh mở cho tiêu chuẩn mã hoá khối của Hoa Kỳ, được tổ chức dưới sự bảo trợ của Viện Tiêu chuẩn hóa và Công nghệ Quốc gia (NIST). Sau khi giành chiến thắng trong cuộc thi, thuật toán này đã nhận được tên thứ hai AES (Advanced Encryption Standard). Mật mã khối này phát triển một độc đáo cho cấu trúc gần đây của mạng KALST hình chữ nhật. KALST là tên viết tắt của các thuật ngữ tiếng Anh Key Addition, Substitution (thay thế dạng bảng), Linear Transposition (trộn tuyến tính).

Rijndael là một mật mã khối lặp đi lặp lại với độ dài khối thay đổi và độ dài khóa khác nhau. Chiều dài của khóa và chiều dài của khối có thể độc lập 128, 192 hoặc 256 bit.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Các phép biến đổi khác nhau hoạt động với một kết quả trung gian được gọi là *Trạng thái (State)*. Trạng thái có thể được biểu diễn dưới dạng một mảng byte hình chữ nhật. Mảng này có 4 hàng và số cột được ký hiệu là N_b và bằng chiều dài của khối chia cho 32 (khối 128, 192, và 256 bit). Khóa mã hóa cũng được biểu diễn dưới dạng một mảng hình chữ nhật có bốn hàng. Số cột được ký hiệu là N_k và bằng chiều dài của khóa chia cho 32 (khóa 128, 192, và 256 bit).

A(0,0)	A(0,1)	A(0,2)	A(0,3)	A(0,4)	A(0,5)
A(1,0)	A(1,1)	A(1,2)	A(1,3)	A(1,4)	A(1,5)
A(2,0)	A(2,1)	A(2,2)	A(2,3)	A(2,4)	A(2,5)
A(3,0)	A(3,1)	A(3,2)	A(3,3)	A(3,4)	A(3,5)

K(0,0)	K(0,1)	K(0,2)	K(0,3)
K(1,0)	K(1,1)	K(1,2)	K(1,3)
K(2,0)	K(2,1)	K(2,2)	K(2,3)
K(3,0)	K(3,1)	K(3,2)	K(3,3)

Ví dụ về biểu diễn (N_b =6) và khóa mã hóa (N_k=4)

Dữ liệu đầu vào cho mật mã được chỉ định là byte trạng thái theo thứ tự $A_{0,0}, A_{1,0}, A_{3,0}, A_{0,1}, A_{1,1}, A_{3,1}, A_{4,01}, \ldots$. Sau khi mật mã được hoàn thành, dữ liệu gốc ra khỏi các byte trạng thái theo cùng một thứ tự, số chu kỳ được chỉ định của $A_{0,0}, A_{1,0}, A_{3,0}, A_{0,1}, A_{1,1}, A_{3,1}, A_{4,01}, \ldots$ phụ thuộc vào giá trị của N_b và N_k.

Số chu kỳ của Rijdael là một hàm của chiều dài khối và khoá			
N_r	N_b = 4	N_b = 6	N_b = 8
N_k = 4	10	12	14
N_k = 6	12	12	14
N_k = 8	14	14	14

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Quá Trình mã hoá:

Giải thuật AES thực hiện mã hóa khối dữ liệu bản rõ, gồm các bước xử lý chính như sau:

- **Bước 1**: Mở rộng khóa thực hiện việc sinh các khóa vòng (Round key) dùng trong các vòng lặp từ khóa chính AES sử dụng thủ tục sinh khóa Rijndael.

- **Bước 2**: Vòng khởi tạo thực hiện hàm AddRoundKey, trong đó mỗi byte trong state được kết hợp với khóa vòng sử dụng phép XOR.

- **Bước 3**: Các vòng lặp chính, trong đó mỗi vòng thực hiện 4 hàm biến đổi dữ liệu như sau:

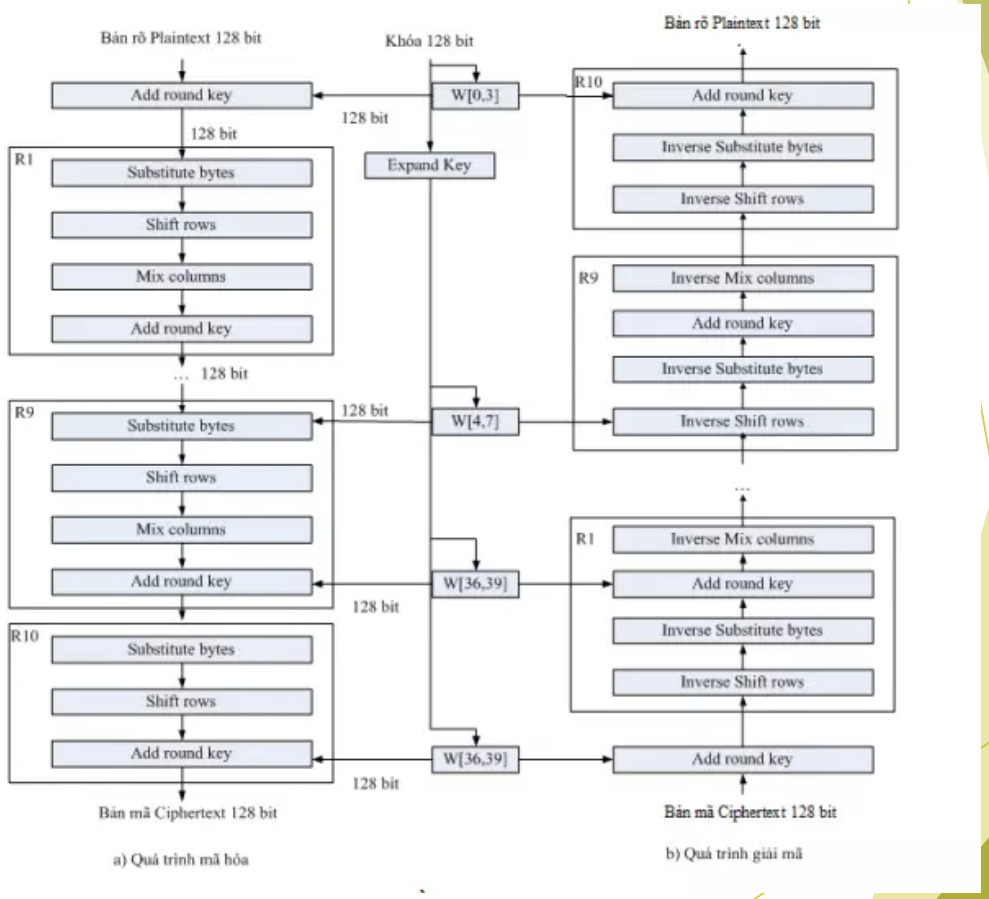
+ **SubBytes** là hàm thay thế phi tuyến tính, trong đó mỗi byte trong state được thay thế bằng một byte khác sử dụng bảng tham chiếu S-box;

+ **ShiftRows** là hàm dịch dòng, trong đó mỗi dòng trong state được dịch một số bước theo chu kỳ;

+ **MixColumns** là làm trộn các cột trong state, kết hợp 4 bytes trong mỗi cột.

+ **AddRoundKey** là hàm kết hợp state với khóa vòng sử dụng phép XOR.

- **Bước 4**: Vòng lặp cuối tương tự các vòng lặp chính, nhưng chỉ thực hiện 3 hàm biến đổi dữ liệu, bao gồm SubBytes, ShiftRows và AddRoundKey.



MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Phép biến đổi ByteSub là một phép thay thế byte phi tuyến tính được thực hiện cho từng byte trạng thái một cách độc lập. Bảng thay thế có thể đảo ngược và được xây dựng như một thành phần của hai phép biến đổi:

1. Đầu tiên, với sự trợ giúp của thuật toán Euclid, một phép nhân nghịch đảo trong GF (2^8) được tính theo modulo một đa thức $m\ (x)$ có dạng $m(x) = x^8 + x^4 + x^3 + x + 1$. '00' là được phản ánh trong chính nó.

2. Sau đó, một phép biến đổi Affine (trong GF (2)) được áp dụng, được định nghĩa như sau:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

AES S-box

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
10	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
20	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
30	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
40	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
50	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
60	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
70	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
80	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
90	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a0	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b0	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c0	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d0	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e0	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f0	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Đảo ngược ByteSub là ứng dụng thay thế byte theo bảng nghịch đảo. Điều này thu được bằng cách đảo ngược ánh xạ affine và đảo ngược phép nhân trong GF (2^8).

Trong phép biến đổi ShiftRow, các hàng trạng thái được chuyển theo chu kỳ sang các giá trị khác nhau. Dòng số 0 không di chuyển. Dòng 1 được dịch chuyển bởi C_1 byte, dòng 2 bởi C_2byte, dòng 3 bởi C_3 byte. Giá trị của C_1 , C_2 và C_3 phụ thuộc vào N_b .

Lượng dịch chuyển phụ thuộc vào độ dài của khối			
N_b	C_1	C_2	C_3
4	1	2	3
6	1	2	3
8	1	3	4

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Đảo ngược đối với ShiftRow là sự dịch chuyển theo chu kỳ của ba hàng dưới lần lượt theo các byte N_b - C_1, N_b - C_2 và N_b - C_3, sao cho byte ở vị trí j trong hàng i di chuyển đến vị trí $(j + N_b - C_i)\ mod\ N_b$. Trong phép biến đổi MixColumn, các cột trạng thái được coi là đa thức trong GF (2^8) và nhân theo modulo $x^4 + 1$ với một đa thức cố định: $c(x) = '03'x^3 + '01'x^2 + '01'x + '02'$ nghịch đảo. Điều này có thể được viết dưới dạng phép nhân ma trận. Cho $b\ (x)\ =\ c\ (x)\ \times\ a\ (x)$.

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Đảo ngược MixColumn tương tự như MixColumn. Mỗi cột được chuyển đổi bằng cách nhân nó với một đa thức d(x), được định nghĩa như sau:

$$('03'x^3 + '01'x^2 + '01'x + '02') \times d(x) = '01'$$

Kết quả nhận được là:

$$\mathrm{d}(x) = '0B'x^3 + '0D'x^2 + '09'x + '0E'.$$

Kết hợp với khoá tròn AddRoundKey được thực hiện như một thao tác XOR theo chiều dọc của khoá tròn với trạng thái hiện tại. Chiều dài của khoá vòng bằng chiều dài của khối N_b.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

AddRoundKey là một sự đảo ngược của chính nó.

Các khoá vòng có nguồn gốc từ khóa mã hóa bằng cách sử dụng phép biến đổi bao gồm hai thành phần khoá chính (Key Expansion) và lựa chọn khoá vòng (Round Key Selection). Các nguyên tắc cơ bản của thuật toán như sau:

1. Tổng số bit của khoá vòng bằng chiều dài khối nhân với số chu kỳ + 1 (ví dụ, độ dài khối 128 bit và 10 chu kỳ cần 1408 bit của khóa chu kỳ).
2. Khóa mã hóa được mở rộng thành Khóa mở rộng (Expanded Key.
3. Các khóa tuần hoàn được lấy từ Khóa mở rộng như sau: khóa tuần hoàn đầu tiên chứa N_b từ đầu tiên, khóa thứ hai chứa N_b từ sau, v.v.

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Khóa mở rộng là một mảng tuyến tính gồm các từ bốn byte và được ký hiệu là:

$$W[N_b \times (N_r + 1)]$$

$\boldsymbol{N_k}$ từ đầu tiên bao gồm khóa mã hóa. Các từ còn lại được định nghĩa một cách đệ quy. Hàm mở rộng khóa phụ thuộc vào giá trị của $\boldsymbol{N_k}$: có một phiên bản của hàm cho $\boldsymbol{N_k}$ bằng hoặc nhỏ hơn 6 và một phiên bản cho $\boldsymbol{N_k}$ lớn hơn 6. Với $\boldsymbol{N_k}$ ≤6, chúng ta có:

```
KeyExpansion(byte Key[4* Nk], word W[Nb *(Nr +1)])
{
 for(i=0;i< Nk;i++)
  W[i]=(Key[4*i],Key[4*i+1],Key[4*i+2],Key [4*i+3]);
 for(i= Nk;i< Nb *(Nr +1);i++) {
  temp=W[i-1];
  if (i% Nk ==0)
   temp=ByteSub(RotByte(temp))^Rcon[i/ Nk];
  W[i]=W[i- Nk]^temp;
                                                    }
}
```

Trong trường hợp này, ByteSub là một hàm trả về một từ bốn byte, trong đó mỗi byte là kết quả của việc áp dụng S-box Rijndael cho byte tại vị trí tương ứng trong từ đầu vào. Hàm RotByte (W) trả về một từ, trong đó các byte được sắp xếp lại theo chu kỳ sao cho từ đầu vào (a, b, c, d) tạo ra từ đầu ra (b, c, d, a).

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Có thể thấy rằng N_k từ đầu tiên được điền bằng khóa mã hóa. Mỗi từ $W\ [i]$ tiếp theo bằng XOR của từ $W\ [i-1]$ trước đó và các vị trí của các từ, từ N_k đến $W\ [i\ -N_k]$.

Đối với các từ ở vị trí là bội số của N_k, việc chuyển đổi được áp dụng với XOR đến $W\ [i-1]$ và hằng số vòng. Việc chuyển đổi này bao gồm sự thay đổi theo chu kỳ của các byte trong các từ RotByte, tiếp theo là việc áp dụng thay thế bảng cho tất cả bốn byte trong một từ (ByteSub).

Với $N_k > 6$

```
KeyExpansion(byte Key [4* Nk ],word W[Nb *(Nr +1)])
{for (i=0;i< Nk;i++)
    W[i]=(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3]);
 for (i= Nk;i< Nb *(Nr +1); i++) {
   temp=W[i-1];
   if(i % Nk ==0)
     temp=SubByte(RotByte(temp))^Rcon[i/ Nk ];
   else if (i % Nk ==4)
     temp=SubByte(temp);
   W[i]=W[i- Nk ]^temp;
}}
```

MÃ HOÁ HIỆN ĐẠI

MÃ HOÁ KHOÁ ĐỐI XỨNG – THUẬT TOÁN AES

Sự khác biệt trong sơ đồ cho $N_k \leq 6$ là đối với bội số $i-4$ của N_k, SubByte được sử dụng cho $W\ [i-1]$ trước XOR. Các hằng số vòng độc lập với N_k và được định nghĩa như sau:

$Rcon[i] = (RC[i]\text{`00`}, \text{`00`}, \text{`00`})$

$RC[i]$, là các thành phần trong GF (2^8) $với\ giá\ trị\ x(i-1)$ ta có:

$$RC[1] = 1\ (tức\ là \text{`01`})$$

$$RC[i] = x(tức\ là \text{`02`}) \cdot (RC[i-1]) = x(i-1)$$

Khoá vòng và xuất phát từ các từ của bộ đệm khoá tròn từ $W[N_b * i]đến\ W[N_b * (i+1)]$

Ưu điểm của thuật toán Rijndael là tốc độ tốt, tính thay đổi khi chọn độ dài của khối và khóa, khả năng song song hóa các quy trình mã hóa / giải mã. Nhược điểm chính của thuật toán là sự bất đối xứng của nó theo quy trình mã hóa / giải mã.