# **Chuẩn Mật Mã Hóa Nâng Cao**

# **Advanced Encryption Standard**

# **(AES)**

1. **Khái Niệm và Lịch sử hình thành**

* Năm 1997: NIST (National Institute of Standards and Technology) của Hoa Kỳ khởi động quá trình chọn lọc một tiêu chuẩn mã hóa mới để thay thế hệ mật DES (Data Encryption Standard). Các ứng cử viên được đề xuất bao gồm Rijndael, MARS, Twofish, Serpent và RC6.
* Năm 2000: Rijndael, một thuật toán mã hóa đối xứng do hai nhà mật mã học người Bỉ Joan Daemen và Vincent Rijmen thiết kế, được chọn làm ứng viên chiến thắng. Rijndael có cấu trúc đơn giản, hiệu suất cao và khả năng mở rộng dễ dàng.
* Năm 2001: NIST chính thức công bố Rijndael là tiêu chuẩn mã hóa mới và đổi tên thành AES. AES được chấp nhận là một tiêu chuẩn toàn cầu và trở thành một phần quan trọng của nhiều ứng dụng mật mã trên thế giới.
* AES có ba biến thể khác nhau dựa trên kích thước khóa: AES-128, AES-192 và AES-256. Mỗi biến thể sử dụng Rijndael nhưng có các khóa và số vòng lặp khác nhau.
* AES đã được sử dụng rộng rãi trong nhiều lĩnh vực bảo mật, bao gồm mạng không dây, ứng dụng di động, giao dịch tài chính trực tuyến và nhiều hệ thống mã hóa dữ liệu khác.
* Trong suốt thời gian sử dụng, AES đã chứng tỏ độ bảo mật cao, khả năng chống lại các cuộc tấn công phổ biến và hiệu suất tốt.

1. **Đặc Điểm**

* AES- Là một thuật toán mã hóa khối đối xứng ( sử dụng 128 bits dữ liệu)
* Đọ dài khóa là 128,192 hoặc 256 bits tương ứng AES-128, AES-192, AES-256
* Vòng lặp chính của AES có các hàm sau:

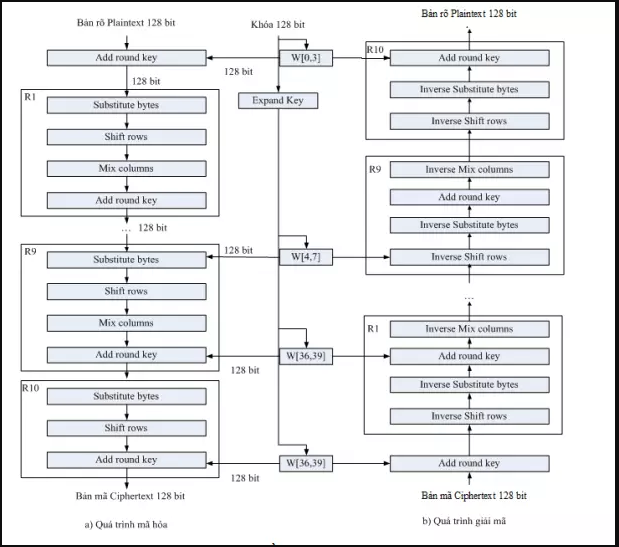
SubBytes() : thực hiện thay thế các byte dữ liệu( trạng thái)

ShiftRows() : dịch vòng dữ liệu( trạng thái)

MixColumns() : Trộn cột dữ liệu vào( trạng thái)

AddRoundKey(): Chèn khóa vòng

1. **Cấu trúc hệ mật**

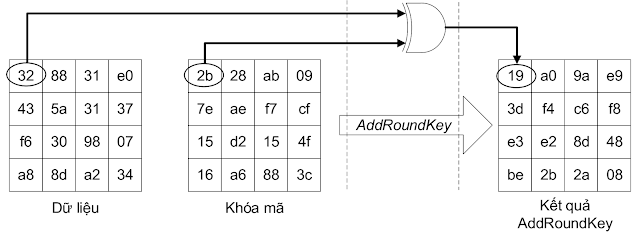


* Bước 1. Bước khởi tạo: dữ liệu cần được mã hóa plain\_text[127:0] kết hợp với key[127:0] bằng chức năng AddRoundKey
* Bước 2. Bước lặp mã hóa: kết quả bước 1 được sử dụng để thực hiện tuần tự các chức năng SubBytes, ShiftRows, MixColumns và AddRoundKey. Bước này được lặp lại 9 lần. Chú ý, KeyExpansion thực hiện song song với bước AddRoundKey để tạo khóa vòng cho chức năng này.
* Bước 3. Bước tạo ngõ ra: Sau 9 lần lặp ở bước 2, kết quả được sử dụng để thực hiện tuần tự các chức năng SubBytes, ShiftRows và AddRoundKey để tạo ngõ ra cipher\_text[127:0].

1. **Chức năng AddRoundKey( chèn khóa vòng)**

Chức năng AddRoundKey thực hiện ở:

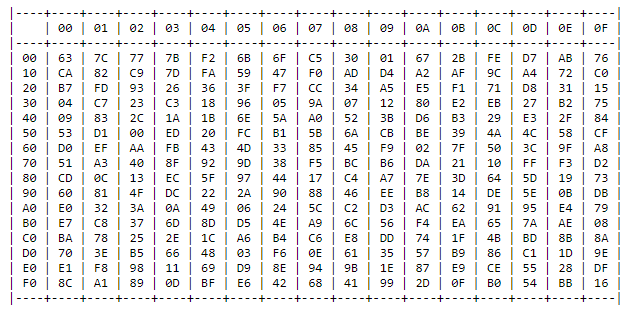
* Bước khởi tạo: XOR khóa mã với ma trận dữ liệu
* Bước lặp mã hóa và bước tạo ngõ ra: XOR khóa vòng (round key) với ma trận trạng thái.



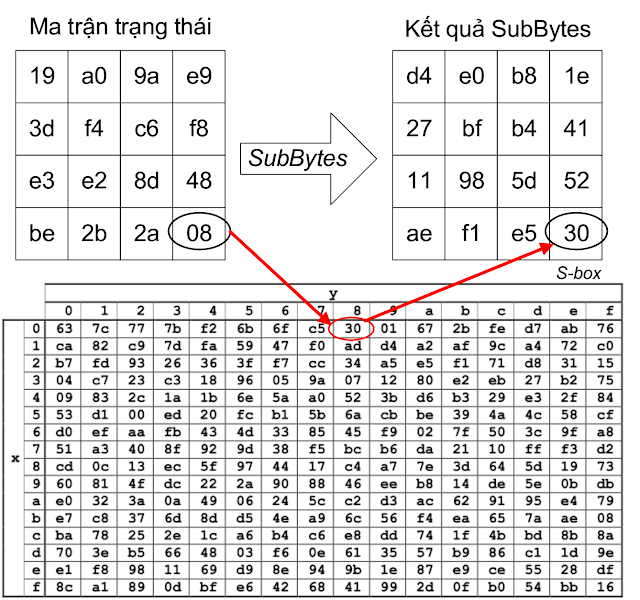
Đối với bước lặp mã hóa và bước tạo ngõ ra, vị trí "khóa mã" là các "khóa vòng" còn dữ liệu là của lần tính trước đó.

1. **Chức năng SubBytes(Subtitute Byte)**

Chức năng SubBytes là thực hiện thay thế từng byte của ma trận trạng thái, ngõ ra của AddRoundKey, bằng một giá trị đã quy định trong chuẩn AES. Bảng quy định giá trị thay thế gọi là S-box.

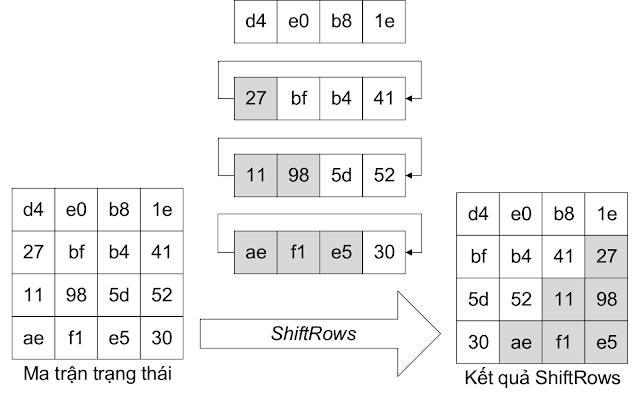


Ví dụ: byte cần thay thế là H08 thì dò ở hàng số 0 và cột số 8 trong bảng S-box sẽ được kết quả là 30.



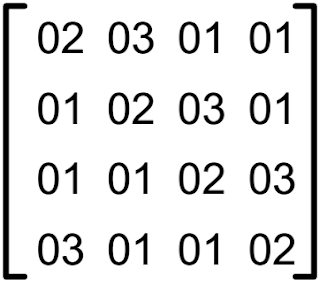
1. **Chức năng ShiftRows**

Chức năng ShiftRows thực hiện quay trái từng hàng của ma trận trạng thái, ngõ ra của SubBytes, theo byte với hệ số quay tăng dần từ 0 đến 3. Hàng đầu tiên có hệ số quay là 0 thì các byte được giữ nguyên vị trí. Hàng thứ hai có hệ số quay là 1 thì các byte được quay một byte. Hàng thứ ba quay hai byte và hàng thứ tư quay ba byte.



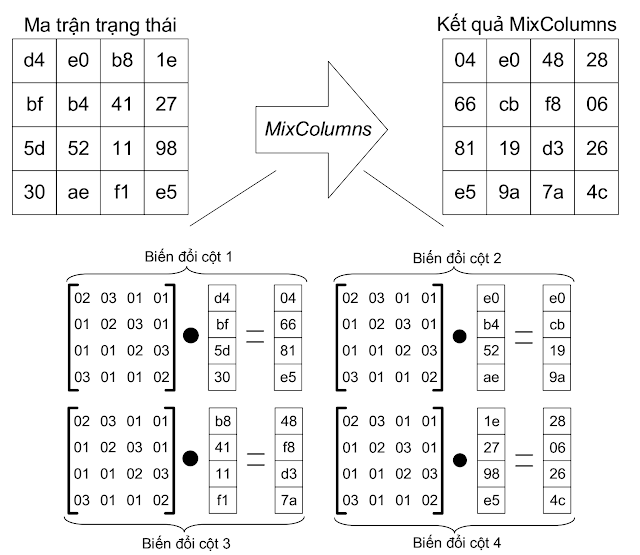
1. **Chức năng MixColumns**

Chức năng MixColumns thực hiện nhân từng cột của ma trận trạng thái, ngõ ra của ShiftRows, với một ma trận chuyển đổi quy định bởi chuẩn AES.



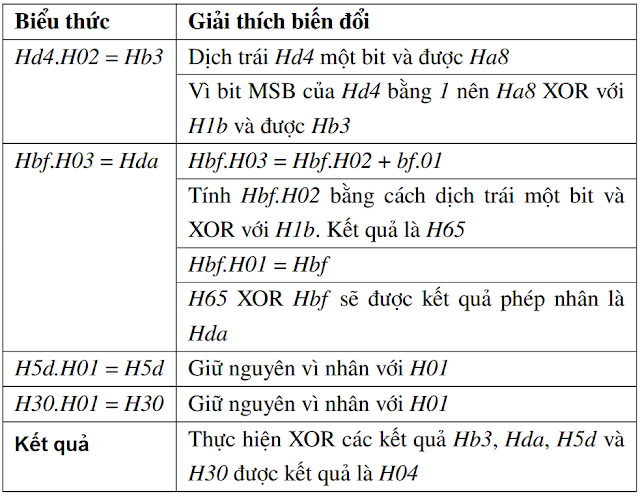
Việc biến đổi một cột của ma trận trạng thái được thực hiện bởi hai phép toán là nhân (.) và XOR (+).  
Biểu thức sau tạo ra phần tử *H04,* H là ký hiệu của số Hex, ở cột 1 trong hình minh họa "chức năng MixColumns".  
ví dụ:

H04 =Hd4.H02 + Hbf.H03 + H5d.H01 + H30.H01=Hd4.H02 + (Hbf.H02 + Hbf.H01) + H5d.H01 + H30.H01



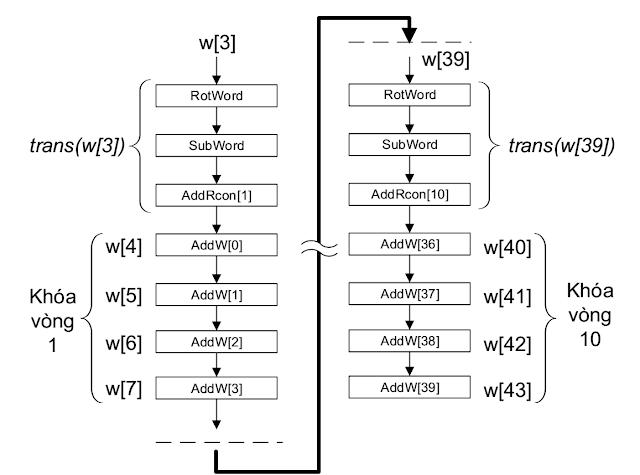
Phép nhân với H01 thì giữ nguyên giá trị. Phép nhân với H02 tương đương với việc dịch trái một bit và XOR có điều kiện như sau:

* Nếu bit MSB của giá trị được dịch bằng 1 thì giá trị sau khi dịch được XOR với H1b
* Nếu bit MSB của giá trị được dịch bằng 0 thì giữ giá trị saukhi dịch.



1. **Chức năng KeyExpansion**

Chức năng KeyExpansion thực hiện tính toán khóa vòng cho bước lặp mã hóa và bước tạo ngõ ra. Kết quả của một lần thực thi KeyExpansion là một khóa vòng sử dụng cho chức năng AddRoundKey. Với mã hóa AES-128, số khóa vòng là 10 tương ứng với 9 lần AddRoundKey ở bước lặp mã hóa và 1 lần AddRoundKey ở bước tạo ngõ ra.  
Chức năng KeyExpansion được thực hiện thông qua 4 chức năng là RotWord, SubWord, AddRcon và AddW.



Mỗi khóa vòng có 128 bit được chia làm 4 word, mỗi word là 4 byte và ký hiệu là w[j] với j là số nguyên. Mã hóa AES-128 có 1 khóa mã và 10 khóa vòng nên tổng số từ là 44 và được đánh số từ 0 đến 43. Khóa mã có 4 từ là w[0], w[1], w[2] và w[3]. Khóa vòng 1 có 4 từ là w[4], w[5], w[6] và w[7]. Tương tự, khóa vòng 10 có 4 từ là w[40], w[41], w[42] và w[43].  
Từ w[j] tính theo công thức sau, với 3 < j < 44.

**w[j] = AddW[j - 4] = w[j - 1] + w[j - 4]**

**w[j = 4∗ n] = AddW[j - 4] = trans(w[j - 1])+ w[j - 4]**

Chú ý, khi tính các từ ở vị trí j là bội số của 4, như w[4], w[8],... và w[40], thì w[j-1] phải được biến đổi qua 3 chức năng RotWord, SubWord và AddRcon, gọi là trans(w[j-1]), trước khi XOR với w[j-4].  
Khóa mã key ở mục 1 được sử dụng để minh họa việc tính toán khóa vòng. Khóa mã key[127:0] được chia làm 4 từ như biểu thức sau:

**w[0] = 2b7e1516 w[1] = 28aed2a6**

**w[2] = abf71588 w[3] = 09cf4f3c**

Việc tính toán khóa vòng 1 là thực hiện tính 4 từ w[4], w[5], w[6] và w[7]. Để tính khóa vòng 1, trans(w[3]) phải được tính trước thông qua 3 chức năng RotWord, SubWord và AddRcon.

**w[4] = AddW[0] = trans(w[3])+ w[0]w[5] = AddW[1] = w[4]+ w[1]w[6] = AddW[2] = w[5]+ w[2]w[7] = AddW[3] = w[6]+ w[3]**

Chức năng RotWord Chức năng RotWord thực hiện quay trái từ w[j] một byte.

|  |
| --- |
|  |
| Hình 11:  Thực thi RotWord cho từ w[3] |

Chức năng SubWord thực hiện thay thế các phi tuyến từng byte của kết quả RotWord theo bảng S-box.

|  |
| --- |
|  |
| Hình 12:  Thực thi SubWord khi chuyển đổi từ w[3] |

Chức năng AddRcon thực hiện XOR kết quả SubWord và giá trị Rcon[j/4] với j là bội số của 4. Số lượng giá trị Rcon[j/4] là 10 tương ứng với 10 lần tính khóa vòng. Chức năng AddRcon sẽ tạo ra kết quả cuối cùng của biến đổi trans(w[j-1]).  
  
**Rcon[j/4]** **Giá trị  HEX** **Vị trí sử dụng**  
Rcon[1]   01000000        sử dụng cho trans(w[3]) khi tính w[4]  
Rcon[2]   02000000        sử dụng cho trans(w[7]) khi tính w[8]  
Rcon[3]   04000000        sử dụng cho trans(w[11]) khi tính w[12]  
Rcon[4]   08000000        sử dụng cho trans(w[15]) khi tính w[16]  
Rcon[5]   10000000        sử dụng cho trans(w[19]) khi tính w[20]  
Rcon[6]   20000000        sử dụng cho trans(w[23]) khi tính w[24]  
Rcon[7]   40000000        sử dụng cho trans(w[27]) khi tính w[28]  
Rcon[8]   82000000        sử dụng cho trans(w[31]) khi tính w[32]  
Rcon[9]   1b000000        sử dụng cho trans(w[35]) khi tính w[36]  
Rcon[10]  36000000       sử dụng cho trans(w[39]) khi tính w[40]

|  |
| --- |
|  |
| Hình 13:  Thực thi AddRcon khi chuyển đổi từ w[3] |

Chức năng AddW thực hiện XOR w[j-4] với w[j-1] hoặc trans(w[j-1]) như công thức 4.8 để tạo ra khóa vòng.

|  |
| --- |
|  |
| Hình 14:  Thực thi AddW để tạo khóa vòng 1 |

Như vậy, bài viết đã trình bày các bước tính toán cần thiết để mã hóa một chuỗi dữ liệu 128 bit với key 128 bit theo giải thuật AES.