BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP.HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

---o0o---

HỌC PHẦN: BẢO MẬT MÁY TÍNH

TÊN ĐỀ TÀI: GIẢI THUẬT MÃ HOÁ AES

NHÓM 6

Thành phố Hồ Chí Minh, tháng 05 năm 2024

BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP.HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

---o0o---

HỌC PHẦN: BẢO MẬT MÁY TÍNH

TÊN ĐỀ TÀI: GIẢI THUẬT MÃ HOÁ AES

NHÓM 6

Thành viên:

1. Nguyễn Văn Thành
2. Trần Trọng Quí
3. Nguyễn Minh Sang

Thành phố Hồ Chí Minh, tháng 05 năm 2024

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh phúc

BIÊN BẢN PHÂN CÔNG NHÓM

Nhóm trưởng phân công công việc cho các thành viên như sau:

STT	MSSV	HỌ VÀ TÊN	NHIỆM VỤ PHÂN CÔNG	Phần trăm đóng góp
1	2001223968	Trần Trọng Quí	Mixcolumns, invmixcolumn, Code demo (phụ), lý thuyết.	33.33%
2	2001224715	Nguyễn Văn Thành	Shiftrows, invShiftrows, Keyexpansion, Code demo (chính), lý thuyết.	33.33%
3	2001224160	Nguyễn Minh Sang	Addroundkey, subbytes, invsubbytes, Code demo (phụ), lý thuyết.	33.33%

Mục lục

Phân tích bài toán

Mục đích:

Tìm hiểu và áp dụng mã hoá AES vào thực tế.

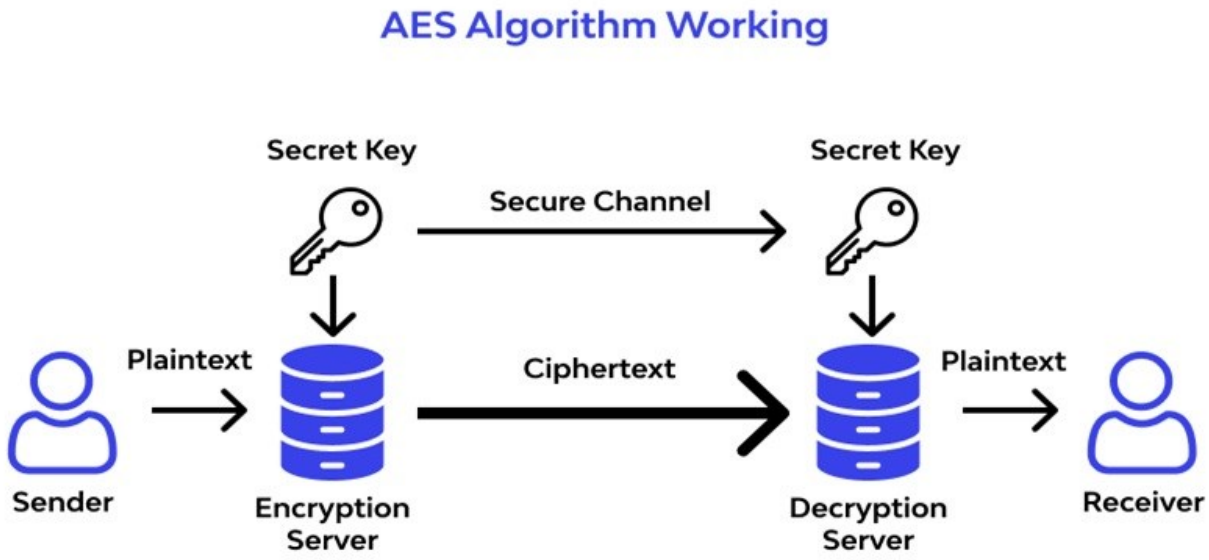
Mục đích của mã hoá và giải mã của thuật toán AES nhằm bảo vệ dữ liệu, thông tin, tránh bị đánh cắp, và bảo đảm tính toàn vẹn của thông tin trong quá trình di chuyển trên đường truyền mạng.

Ngôn ngữ và công cụ sử dụng

- Ngôn ngữ C#
- Phần mềm VS Code

Phần 1. Giới thiệu

Tiêu chuẩn **Advanced Encryption Standard** (AES) - Tiêu chuẩn mã hóa nâng cao là một thuật toán tiêu chuẩn của chính phủ Hoa Kỳ nhằm mã hóa và giải mã dữ liệu do Viện Tiêu chuẩn và Công nghệ quốc gia Hoa Kỳ (National Institute Standards and Technology– NIST) phát hành ngày 26/11/2001 và được đặc tả trong Tiêu chuẩn Xử lý thông tin Liên bang 197 (Federal Information Processing Standard – FIPS 197) sau quá trình kéo dài 5 năm trình phê duyệt, AES tuân theo mục 5131 trong Luật Cải cách quản lý công nghệ thông tin năm 1996 và Luật An toàn máy tính năm 1997.

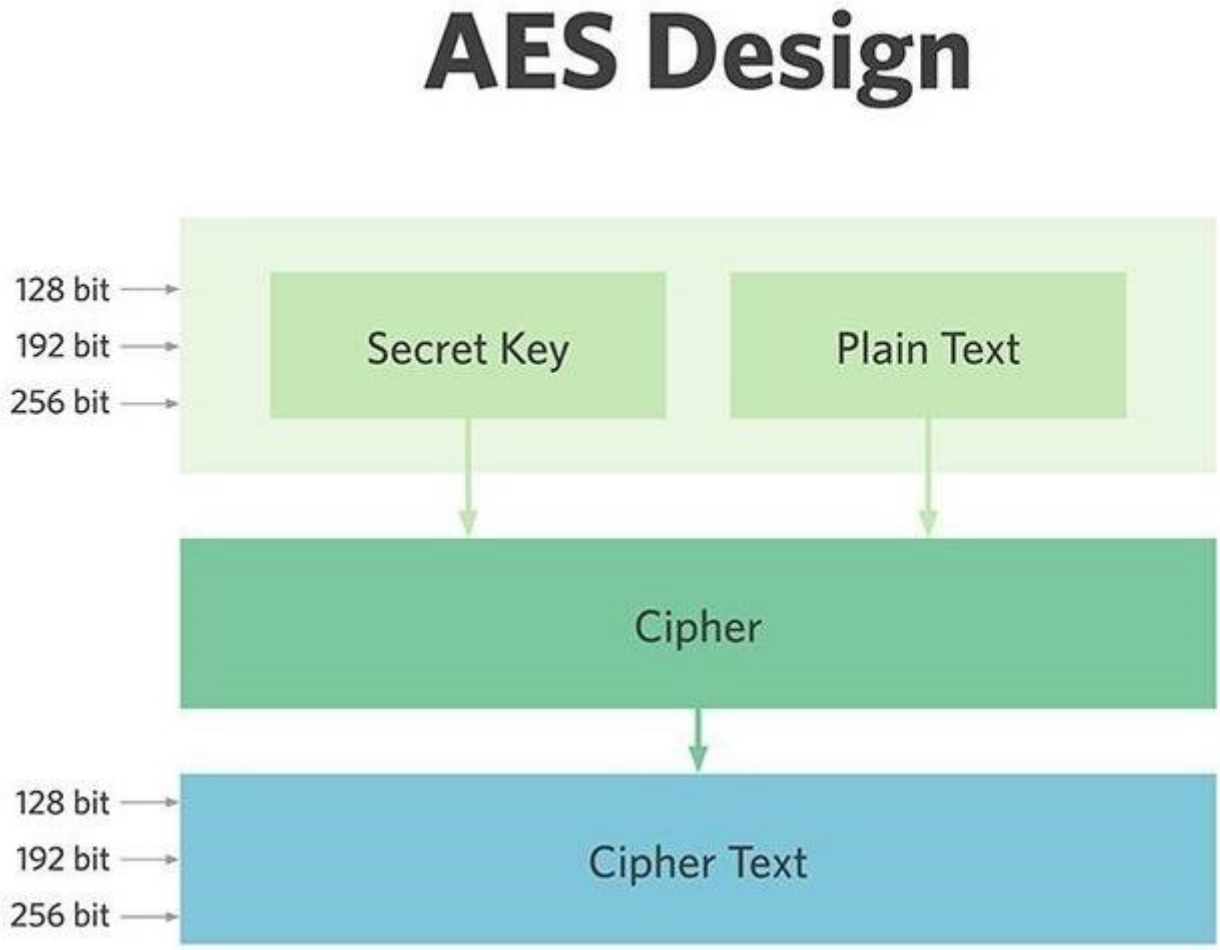


1. Advanced Encryption Standard là một thuật toán mã hóa đối xứng

AES là một thuật toán "**mã hóa khối**" (block cipher) **đối xứng** ban đầu được tạo ra bởi hai nhà mật mã học người Bỉ là Joan Daemen và Vincent Rijmen. Kể từ khi được công bố là một tiêu chuẩn, AES trở thành một trong những thuật toán mã hóa phổ biến nhất sử dụng khóa mã đối xứng để mã hóa và giải mã (một số được giữ bí mật dùng cho quy trình mở rộng khóa nhằm tạo ra một tập các khóa vòng). Ở Việt Nam, thuật toán AES đã được công bố thành tiêu chuẩn quốc gia TCVN 7816:2007 năm 2007 về Thuật toán mã hóa dữ liệu AES.

1.1. Nguyên lí hoạt động

AES là một thuật toán mã hóa khối đối xứng với độ dài khóa là **128 bit** (một chữ số nhị phân có giá trị 0 hoặc 1), **192 bit** và **256 bit** tương ứng được gọi là AES-128, AES-192 và AES-256. AES-128 sử dụng 10 vòng (round), AES-192 sử dụng 12 vòng và AES-256 sử dụng 14 vòng. Mỗi vòng đều thực hiện ba bước thay thế, biến đổi và hòa trộn khối plain text (bản rõ) đầu vào để biến nó thành Ciphertext (bản mã).



2 AES design

Thông tin được chính phủ phân loại theo ba cấp độ: bảo mật, bí mật, tối mật. Tất cả các độ dài của key từ 128, 192 và 256 bit đều được dùng ở cấp độ bảo mật, bí mật. Riêng với những thông tin tối mật để đảm bảo không xảy ra bất cứ sai sót nào phải cần đến key 192 hoặc 256 bit. Mật mã sẽ dùng một key riêng tư để mã hóa và giải mã dữ liệu và tất nhiên cả người gửi và người nhận đều phải nhận biết và sử dụng được key này.

Dữ liệu cần được chia thành các khối 16 byte và được lưu trữ trong một ma trận 4x4 gọi là trạng thái. Nếu dữ liệu không đủ 16 byte, có thể sử dụng các kỹ thuật như đệm (padding) hay chuẩn hóa (normalization) để bổ sung.

Khóa bí mật cũng cần được chia thành các khối 16 byte và được lưu trữ trong một ma trận 4x4 gọi là khóa vòng. Tùy vào độ dài của khóa ban đầu, có thể sinh ra từ 10 đến 14 khóa vòng khác nhau bằng cách sử dụng các phép biến đổi như hoán vị, thay thế và XOR.

Quá trình mã hóa gồm có nhiều vòng lặp (round), tùy vào độ dài của khóa ban đầu.Vòng lặp chính của AES thực hiện các hàm sau: SubBytes(), ShiftRows(), MixColumns() và AddRoundKey(). Ba hàm đầu của một vòng AES được thiết kế để ngăn chặn phân tích mã bằng phương thức "mập mờ" (confusion) và phương thức "khuếch tán" (diffusion), còn hàm thứ tư mới thực sự được thiết kế để mã hóa dữ liệu. Trong đó "khuếch tán" có nghĩa là các kiểu mẫu trong bản rõ (Dữ liệu đầu vào của phép mã hóa hoặc dữ liệu đầu ra của phép giải mã) được phân tán trong các bản mã (Dữ liệu đầu ra của phép mã hóa hoặc dữ liệu đầu vào của phép giải mã), "mập mờ" nghĩa là mối quan hệ giữa bản rõ và bản mã bị che khuất. Một cách đơn giản hơn để xem thứ tự hàm AES là: Trộn từng byte (SubBytes), trộn từng hàng (ShiftRows), trộn từng cột (MixColumns) và mã hóa (AddRoundKey).

Tính bảo mật và toàn vẹn

Với mã hoá AES 128/192/256 bit, dữ liệu được bảo vệ một cách tuyệt đối khỏi các cuộc tấn công mã hoá thô và phá vỡ. Kẻ xấu sẽ gặp rất nhiều khó khăn khi cố gắng giải mã những dữ liệu được mã hoá bằng AES.

Các phép toán chặt chẽ trong AES như SubBytes, ShiftRows, MixColumns và AddRoundKey đảm bảo rằng dữ liệu không thể bị thay đổi hoặc can thiệp trái phép mà không bị phát hiện. Điều này rất quan trọng trong các ứng dụng yêu cầu tính toàn vẹn cao.

1.2. Ứng dụng của AES

Bên cạnh ứng dụng trong việc đảm đảm an toàn cho các tài liệu chính phủ, AES cũng được áp dụng nhiều trong các lĩnh vực khác nhau. Cụ thể:

- Ứng dụng cho tất cả người dùng phổ thông trong quá trình giải mã dữ liệu bằng cách truy cập vào trang web AES Encryption, nhập dữ liệu và áp mã khoá. Tuy nhiên, đây chỉ là phương pháp áp dụng cho các tác vụ thông thường và tính bảo mật thường không cao.

- Mã hoá thông tin trên phần mềm với các ngôn ngữ lập trình như C/C++, Java hay Assembler. AES hỗ trợ rất nhiều cho các hệ điều hành như Linux hay Windows.

- AES áp dụng cho các thiết bị phần cứng bao gồm dòng thiết bị dựa trên hoạt động của hệ vi xử lý và dòng thiết bị cắm qua cổng USB hoặc thẻ thông minh Smart Card.

- Ứng dụng trong truyền thông tin qua Internet thông qua kết nối HTTPS: Dữ liệu sẽ được mã hoá và giải mã thông qua thuật toán AES, thông tin được bảo mật tốt hơn khi so sánh với kết nối HTTP. Bên cạnh đó, wifi hiện nay cũng được ứng dụng thuật toán AES, khi kết hợp với giao thức WPA2, giao tiếp này trở nên an toàn, hiệu quả hơn nhiều và ngăn chặn tấn công trung gian. Bên cạnh đó, AES cũng được sử dụng để mã hoá wifi trên router, kết hợp với giao thức phổ biến WPA2 được gọi là AES/WPA2. AES còn được sử dụng nhằm hỗ trợ mã hoá SSL.

1.3. So sánh AES và DES

DES được xem như tiêu chuẩn mã hóa lâu đời được Hoa Kỳ phát triển cách đây hơn 40 năm. Chức năng chính của DES là mang tới tiêu chuẩn chung, an toàn cho các hệ thống của Chính phủ, đồng thời tạo điều kiện để các hệ thống liên kết với nhau một cách nhanh chóng.

Trong nhiều thập niên, DES đã và đang trở thành trụ cột để bảo vệ an ninh quốc gia của Hoa Kỳ. Mãi đến năm 1999, key 56 bit của DES bị phá vỡ bởi các nhà nghiên cứu. Do đó đến năm 2000, AES đã được nghiên cứu thành công và trở thành giải pháp thay thế cho DES. Tuy DES hiện đã trở nên ít thông dụng hơn nhưng vẫn được sử dụng trong một số trường hợp cụ thể.

Cả AES và DES đều là mật mã khối đối xứng nhưng AES vẫn nhỉnh hơn về hiệu quả bởi độ dài key của nó. Có thể thấy rằng, các key 128 bit, 192 bit và 256 bit của AES mạnh gấp nhiều lần so với 56 bit của DES. Không chỉ vậy, mã hóa AES cũng nhanh hơn so với DES.

Cơ sở để so sánh	DES (Tiêu chuẩn mã hóa dữ liệu)	AES (Tiêu chuẩn mã hóa nâng cao)
Căn bản	Trong DES, khối dữ liệu được chia thành hai nửa.	Trong AES, toàn bộ khối dữ liệu được xử lý dưới dạng một ma trận.
Nguyên tắc	DES hoạt động trên cấu trúc mật mã Feistel.	AES hoạt động trên Nguyên tắc thay thế và hoán vị.
Văn bản thô	Bản rõ là 64 bit	Bản rõ có thể là 128,192 hoặc 256 bit
Kích thước khóa	DES so với AES có kích thước khóa nhỏ hơn.	AES có kích thước khóa lớn hơn so với DES.
Vòng	16 vòng	10 vòng cho thuật toán 128 bit 12 vòng cho thuật toán 192 bit 14 vòng cho thuật toán 256-bit
Tên vòng	Extended License, Xor, S-box, P-box, Xor và Swap.	Subbyte, Shiftbow, MixColumns, Addroundkeys.
Bảo vệ	DES có khóa nhỏ hơn, kém an toàn hơn.	AES có khóa bí mật lớn tương đối do đó, an toàn hơn.
Tốc độ	DES tương đối chậm hơn.	AES nhanh hơn.

3 Biểu đồ so sánh DES và AES

Phần 2. Mã hoá và giải mã AES

2.1. Mã hoá AES

Plaintext của mã hoá AES:

- Mã hoá AES mã hoá một *Plaintext (bản rõ)* có kích thước khối bằng 128-bit thành một *Ciphertext (bản mờ).*
- Plaintext có thể là bất cứ dạng dữ liệu nào, chẳng hạn như văn bản, hình ảnh, âm thanh,…

Key của mã hoá AES:

- *Key là một giá trị bí mật dùng để mã hoá và giải mã AES*
- *Trong đó Key có kích thước khác nhau gồm: 128-bit, 192-bit và 256-bit.* → *Số vòng lặp của thuật toán lần lượt là 10, 12 và 14.*
- *Độ bảo mật của quá trình mã hoá phụ thuộc vào độ dài, ngẫu nhiên và tính khó đoán của khoá.*

❖ Quá trình mã hóa trong thuật toán AES được thực hiện thông qua một loạt các Round. Mỗi Round bao gồm các bước nhất định để biến đổi dữ liệu đầu vào.

Khoá ban đầu (bit)	128	192	256
Plaintext (bit)	128	128	128
Số vòng lặp	10	12	14
Khoá vòng lặp (bit)	128	128	128
Khoá mở rộng (Byte)	176	208	240

4: Mã hoá AES

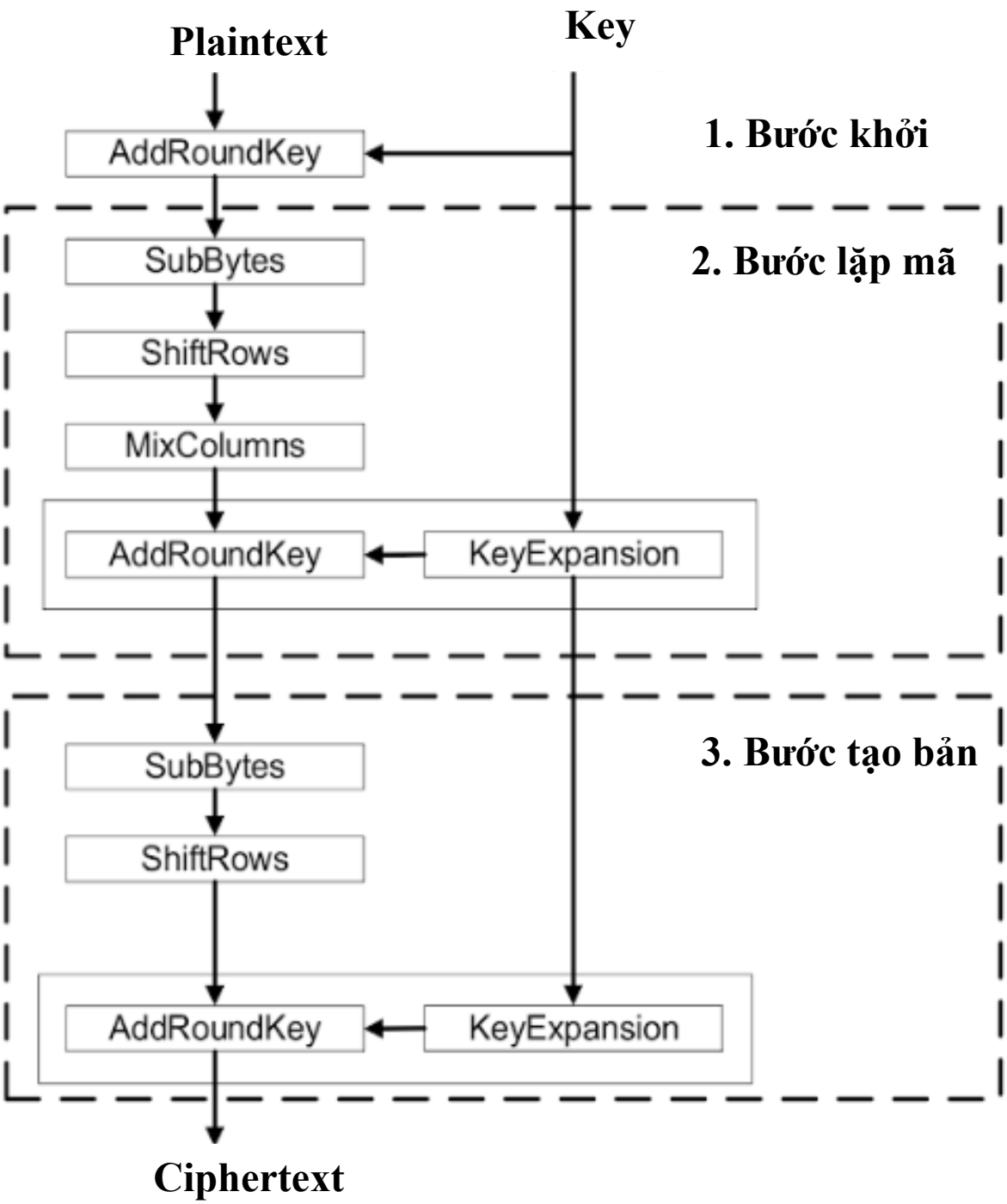
2.1.a. Các bước mã hoá AES:

❖ Quá trình mã hoá được thực hiện thông qua 5 chức năng: **AddRoundKey, SubBytes, ShiftRows, MixColumns và KeyExpansion**.

- Bước 1: Khởi tạo: Khối dữ liệu bản rõ cần mã hoá kết hợp với khoá thông qua *AddRoundKey*.

- Bước 2: Lặp mã hoá: Kết quả của bước 1 được thực thi tuần tự các chức năng *SubBytes, ShiftRows, MixColumns và AddRoundKey*. Bước này được lặp lại (n – 1) lần, trong đó n là số vòng lặp. *KeyExpansion* được thực hiện song song với bước *AddRoundKey* để tạo khoá cho vòng kế tiếp.

- Bước 3: Tạo bản mờ: Ở vòng lặp cuối, kết quả của bước 2 được sử dụng để thực thi tuần tự *SubBytes, ShiftRows và AddRoundKey* để tạo ra bản mờ.



5: Sơ đồ mã hoá AES

2.1.b. ADDROUNDKEY

Trong thuật toán AES (Advanced Encryption Standard), AddRoundKey là một trong các bước chính trong quá trình mã hóa. Trong bước này, mỗi byte của ma trận dữ liệu đầu vào được kết hợp với một byte tương ứng từ khóa con (subkey). Khóa con được tạo ra từ khóa chính dựa trên vòng mã hóa hiện tại.

Chức năng:

+ 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.

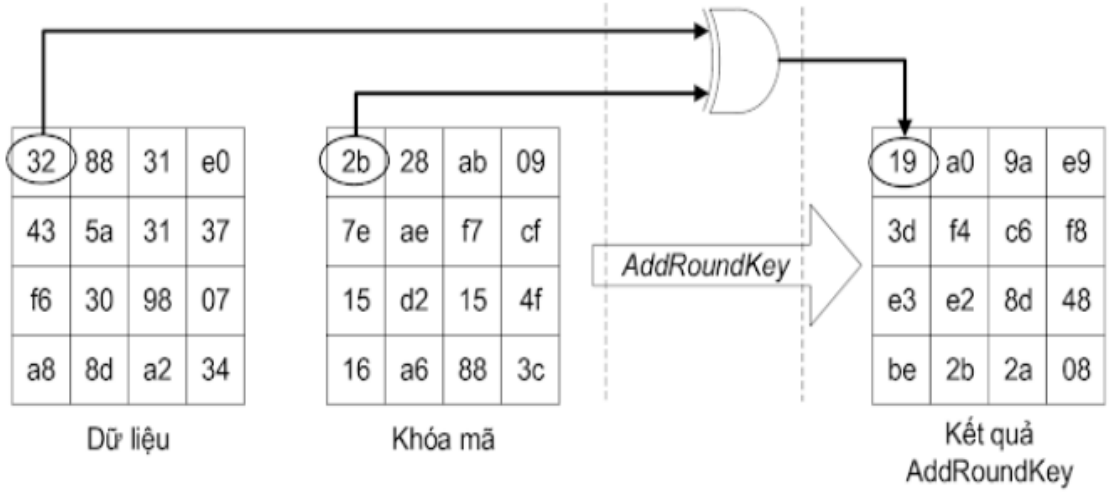
Đối với bước lặp mã hóa và bước tạo ngõ ra, vị trí "khóa mã" là các "khóa vòng" còn dữ liệu là của lần tính trước đó.

Quá trình thực hiện:

+ Mỗi byte của ma trận dữ liệu đầu vào được thực hiện một phép XOR với byte tương ứng trong khóa con.

+ Phép XOR này là một phép toán tạo ra sự kết hợp (hoặc "thêm") giữa dữ liệu và khóa con, từ đó làm cho dữ liệu trở nên không dễ dàng dự đoán và làm tăng độ an toàn của quá trình mã hóa.

Bước này là một trong những bước quan trọng nhất trong thuật toán AES, vì nó đóng góp vào việc tạo ra tính ngẫu nhiên và phức tạp trong quá trình mã hóa, làm cho việc phá mã trở nên khó khăn hơn.



6 AddRoundKey

2.1.c. SUBBYTES

Trong thuật toán AES (Advanced Encryption Standard), SubBytes là một phần của quá trình mã hóa. Trong bước này, các byte trong ma trận dữ liệu đầu vào được thay thế bằng các byte khác sử dụng một hộp thay thế cố định gọi là hộp S-box.

Hộp S-box chứa một bảng 16x16 byte, trong đó mỗi byte được biểu diễn bởi hai hệ số hexa. Quá trình SubBytes thay thế mỗi byte trong ma trận dữ liệu đầu vào bằng byte tương ứng trong hộp S-box.

Quá trình này giúp tăng tính phân tán và phi tuyến trong dữ liệu, làm tăng độ phức tạp của việc phá mã. Nó cũng giúp đảm bảo rằng mỗi byte trong ma trận dữ liệu đầu vào không giữ nguyên giá trị của nó sau mỗi vòng mã hóa, làm cho mật mã trở nên an toàn hơn.

Chức năng: Có các mục tiêu chính sau:

Tính Phi Tuyến: Bằng cách thay thế mỗi byte trong ma trận dữ liệu đầu vào bằng một byte tương ứng trong hộp S-box, SubBytes tạo ra sự phi tuyến trong dữ liệu, làm cho mỗi byte sau mỗi vòng mã hóa trở nên khác biệt và khó khăn hơn cho bất kỳ kẻ tấn công nào để dự đoán.

Phân Tán Dữ Liệu: Quá trình thay thế của SubBytes làm cho mỗi byte trong dữ liệu đầu vào được "phân tán" thành một byte khác, làm tăng sự phức tạp và đa dạng trong dữ liệu sau mỗi vòng mã hóa.

Tăng Độ An Toàn: Bằng cách tạo ra tính phi tuyến và phân tán, SubBytes đóng góp vào việc tăng cường độ an toàn của quá trình mã hóa AES, làm cho nó trở nên khó khăn hơn cho các kẻ tấn công để phá vỡ mã hóa và thu thập thông tin từ dữ liệu đã được mã hóa.

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

7 S-box

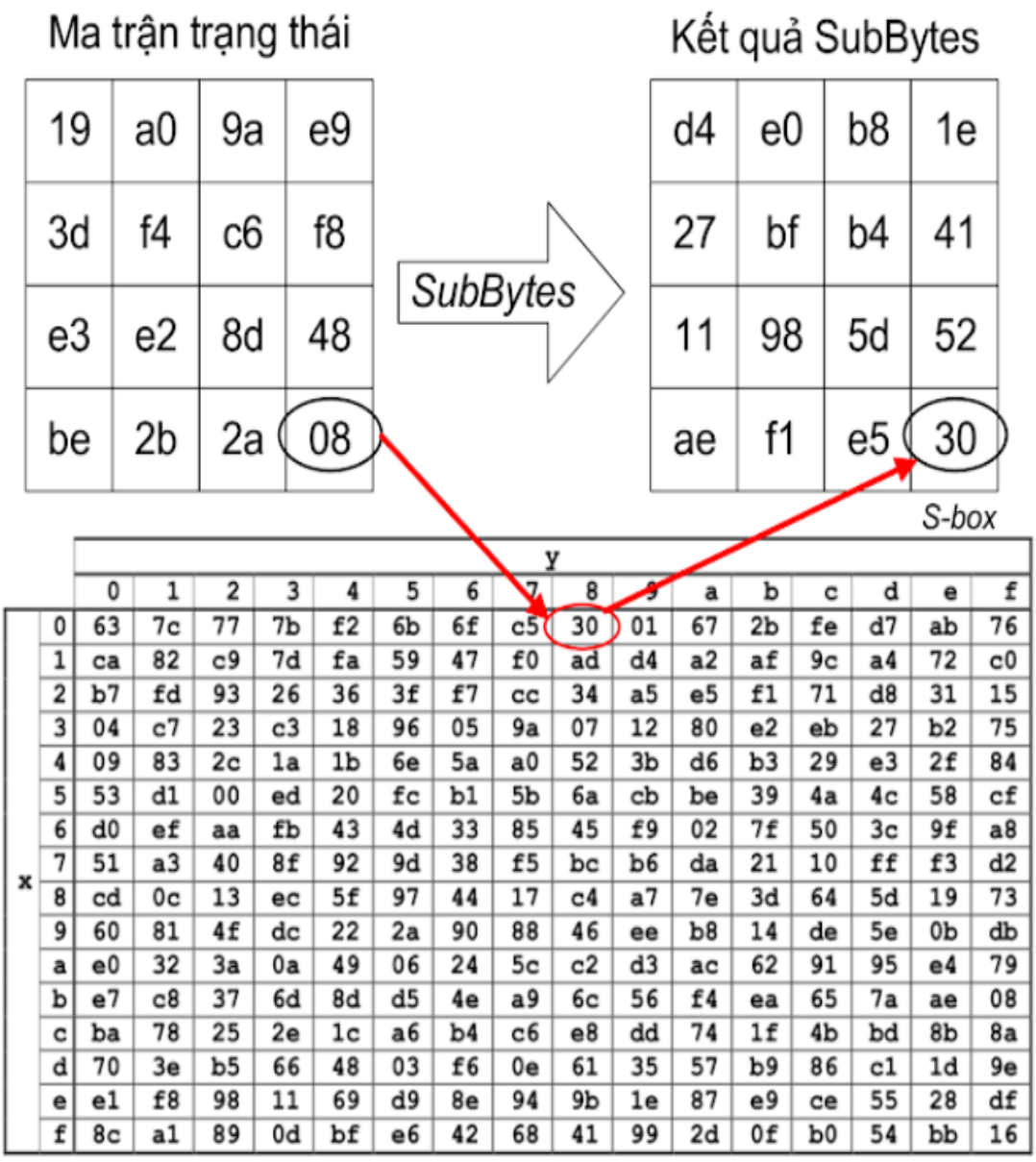
Quá trình thực hiện:

Nhận đầu vào: SubBytes nhận ma trận dữ liệu đầu vào từ bước trước đó trong quá trình mã hóa AES. Đầu vào này thường là một ma trận 4 x 4 byte.

Thay thế byte: Mỗi byte trong ma trận đầu vào được thay thế bằng một byte tương ứng trong hộp S-box. Cụ thể, byte tại hàng i, cột j của ma trận đầu vào được thay thế bằng byte tại hàng i, cột j trong hộp S-box.

Xuất ra: Sau khi tất cả các byte đã được thay thế, ma trận kết quả mới được tạo thành. Đây là đầu vào cho bước tiếp theo của quá trình mã hóa AES.

Quá trình này được lặp lại trong mỗi vòng của quá trình mã hóa AES, trừ vòng cuối cùng. SubBytes đóng vai trò quan trọng trong việc tạo ra tính phi tuyến và phân tán trong dữ liệu, làm tăng tính an toàn của thuật toán mã hóa.



8 SubBytes

2.1.d. MIXCOLUMNS

Phép biến đổi MixColumns thực hiện biến đổi độc lập từng cột trong ma trận state bằng một phép nhân đa thức. Mỗi cột của state đươc coi là biểu diễn của một đa thức trong GF(2^8) như vậy phép biến đổi MixColumns chính là phép nhân theo modulo với $\mathbf{x^4 + 1}$ với một đa thức cố định định nghĩa như sau:

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$$

- Có thể biểu diễn bằng 1 phép nhân ma trận như sau:

$$s'(x) = a(x) \otimes s(x)$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix}$$

- Kết quả là 4 byte trong mỗi cột sẽ được thay thế theo công thức như sau:

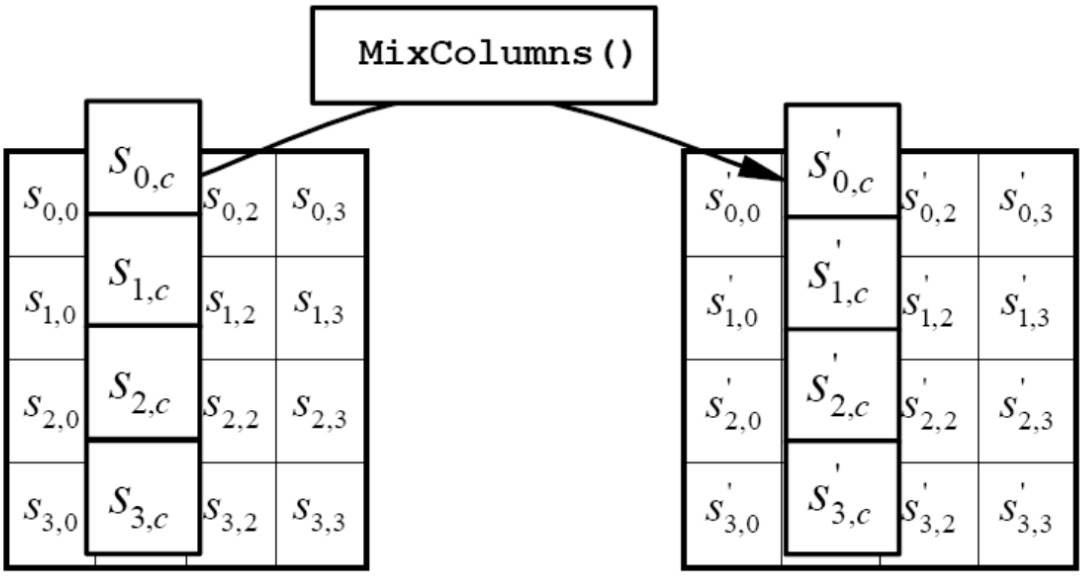
$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c})$$

- Có thể minh họa việc thực hiện của hàm này bằng hình vẽ sau:



9 MixColumns

Ví dụ:

Ma trận trạng thái

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

•

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

=

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

Kết quả cho cột 1

({02}.{87}) ⊕ ({03}.{6E}) ⊕ {46} ⊕ {A6} = {47}

{87} ⊕ ({02} .{6E}) ⊕ ({03}.{46}) ⊕ {A6} = {37}

{87} ⊕ {6E} ⊕ ({02}.{46}) ⊕ ({03}.{A6}) = {94}

({03} .{87}) ⊕ {6E} ⊕ {46} ⊕ ({02}.{A6}) = {ED}

❖ Tương tự ta sẽ thực hiện với các cột còn lại

2.1.e. SHIFTROWS

❖ Phép ShiftRows là một trong các bước của quá trình mã hóa và giải mã trong thuật toán AES.

❖ Với mục tiêu nhằm phân tán dữ liệu trong khối dữ liệu để ngăn chặn các mẫu lặp lại và cải thiện tính bảo mật của thuật toán.

❖ Phép ShiftRows lấy kết quả và thực thi sau phép SubBytes

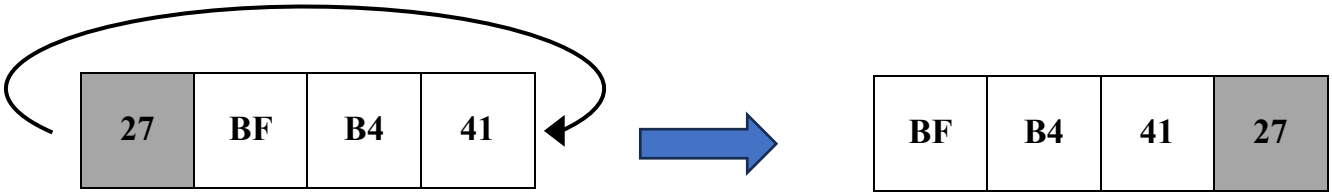
D4	**E0**	**B8**	**1E**
27	**BF**	**B4**	**41**
11	**98**	**5D**	**52**
AE	**F1**	**E5**	**30**

10: Kết quả của phép SubBytes

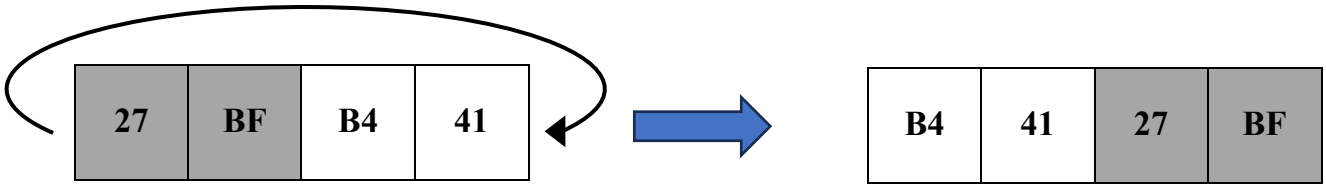
- **Các bước thực hiện phép ShiftRows :** được áp dụng trên mỗi hàng của khối dữ liệu

Theo các quy tắc cụ thể:

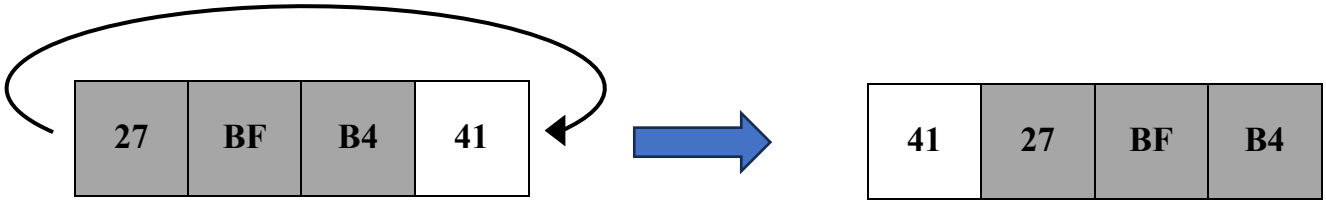
- Hàng đầu tiên không thay đổi.
- Hàng thứ hai được dịch chuyển sang trái một byte.
- Hàng thứ ba được dịch chuyển sang trái hai byte.
- Hàng thứ tư được dịch chuyển sang trái ba byte.



11: Dịch 1 byte sang trái



12: Dịch 2 byte sang trái



13: Dịch 3 byte sang trái

Thực hiện phép ShiftRows:

D4	E0	B8	1E
27	BF	B4	41
11	98	5D	52
AE	F1	E5	30

Kết quả

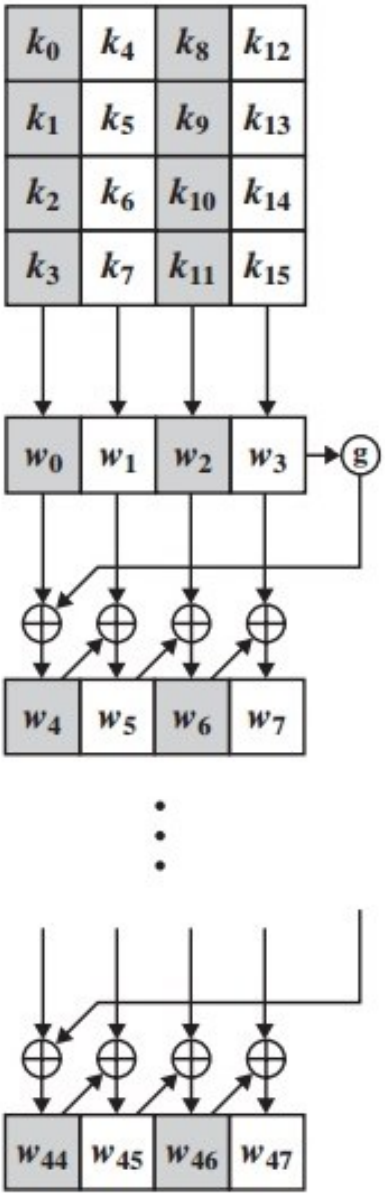
D4	E0	B8	1E
BF	B4	41	27
5D	52	11	98
30	AE	F1	E5

14: Kết quả sau khi thực thi hàm ShiftRows

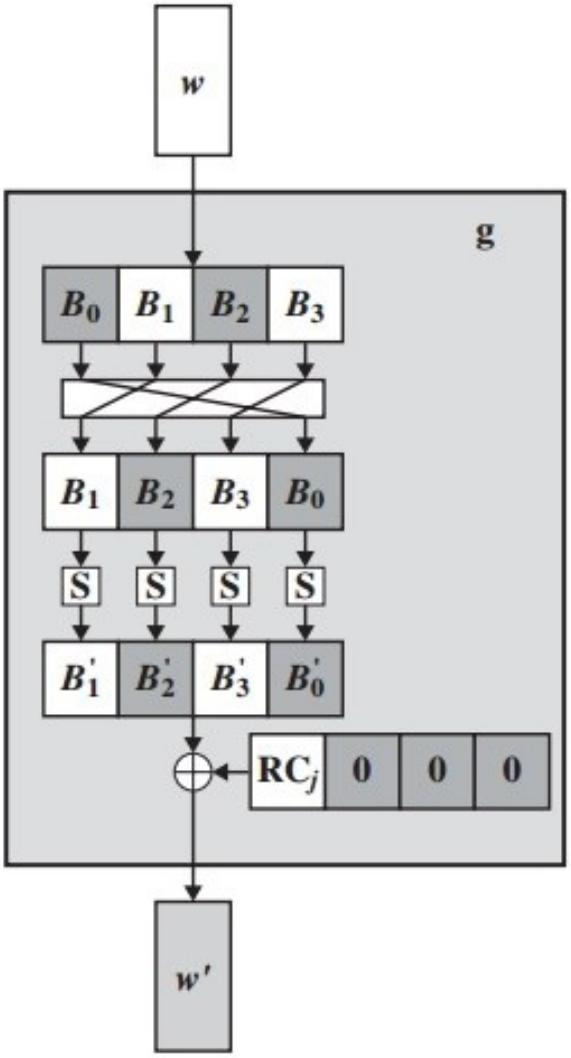
2.1.f. KEYEXPANSION

• Chức năng KeyExpansion là chức năng mở rộng khoá chính sinh ra những khoá con (SubKeys) đây là một chức năng quan trọng để mã hoá và giải mã AES và đảm bảo tính bảo mật của thuật toán.

• Số vòng lập mã hoá cũng thay đổi tuỳ theo độ dài của khoá dẫn đến độ an toàn cũng tăng. Với độ dài khoá là 128-bit, 192-bit, 256-bit thì mỗi lần sử dụng phép KeyExpansion có số vòng lặp lần lượt là 10, 12, 14.



16: Qui trình tạo ra khoá con



15: Hàm g

❖ Giả sử: *key-128* [1]= 2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C

Hàm g

o Hàm g xử lý một word 4 byte cuối trong khoá:

09	*CF*	*4F*	*3C*

o **Rcon (RC)** là một word gồm 4 byte được định nghĩa **Rcon[j] = (RC[j],0,0,0)**.

o Với *RC[1] = 1*, *RC[j] = 2 * RC[j - 1]*, phép nhân được định nghĩa trên trường GF(2^8). Các giá trị của *RC[j]* trong cơ số 16.

j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36

Các bước thực hiện của hàm g:

- Bước 1: Dịch sang trái 1 byte.

CF	*4F*	*3C*	*09*

- Bước 2: Từng byte một được qua một phép **SubBytes.**

CF → 8A	*4F → 84*	*3C → EB*	09 → *01*

- Bước 3: Sử dụng hàm *XOR* giữa kết quả của bước 2 và giá trị *Rcon[j].* Trong đó, j là số thứ tự vòng lặp của KeyExpainsion.

o Ví dụ ở vòng lặp thứ 3: RC[3] = 04 → *Rcon[3] = 04 00 00 00*

[1] Key-128: Là một khoá có 128-bit

Thực thi sinh khoá con

Bước 1: Chia *key-128* thành 4 word, mỗi word 4 byte.

2B 7E 15 16	28 AE D2 A6	AB F7 15 88	09 CF 4F 3C
W0	W1	W2	W3

Bước 2: Đưa W3 đi qua *hàm g.* Ở vòng lặp thứ 1: *Rcon[1] = 01 00 00 00.*

Ta sẽ có kết quả của hàm g ở vòng lặp KeyExpasion thứ 1:

9A	84	EB	01

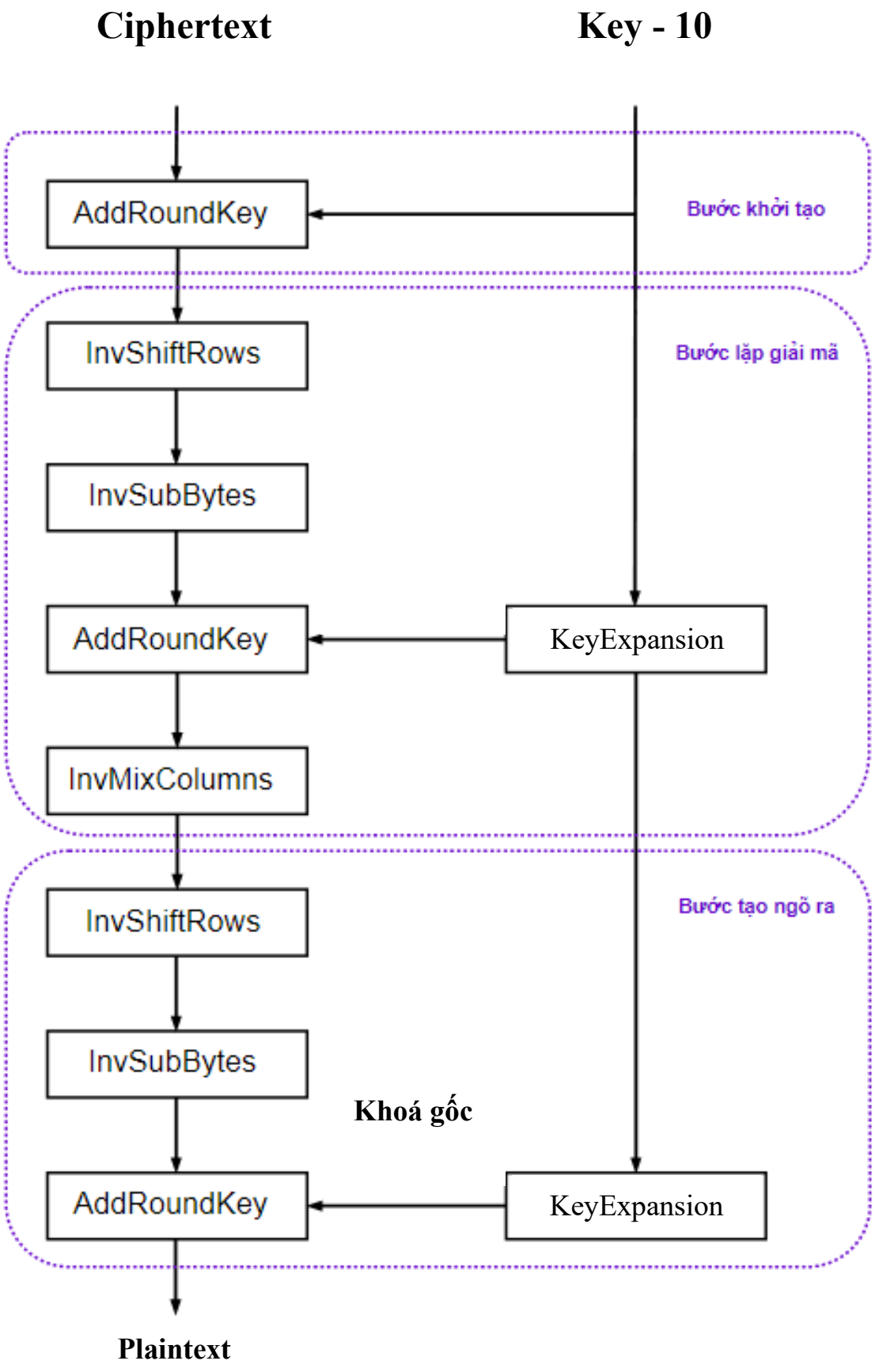
Bước 3: Sử dụng hàm *XOR* giữa W0 và kết quả *hàm g để ra W4.*

W5 = W1 *xor W4 ;* W6 = W2 *xor W5 ;* W7 = W3 *xor W6.*

B1 FA FE 17	99 54 2C B1	32 A3 39 39	3B 6C 76 05
W4	W5	W6	W7

- W[4] W[5] W[6] W[7] sẽ là khoá K_1.
- Những khoá con phía sau sẽ dùng khoá trước nó để biến đổi.

2.2. Giải mã AES



17. Giải mã AES

Cũng tương tự như quá trình mã hoá, giải mã bao gồm 3 bước:

[1]. Bước khởi tạo: AddRoundKey(ciphertext, key[n]).

[2]. Bước lặp giải mã: số vòng lặp ở bước này là n – 1 vòng, gồm các bước:

- InvShiftRows
- InvSubBytes
- AddRoundKey
- InvMixColumns

[3]. Bước tạo ngõ ra: vòng lặp cuối cùng của giải mã, và gồm các bước:

- InvShiftRows
- InvSubBytes
- AddRoundKey

2.2.a. INVSHIFTROWS

Cũng tương tự như hàm *ShiftRows,* chúng ta sử dụng phép dịch 1 byte sang phải.

Theo các quy tắc cụ thể:

- Hàng đầu tiên không thay đổi.
- Hàng thứ hai được dịch chuyển sang phải một byte.
- Hàng thứ ba được dịch chuyển sang phải hai byte.
- Hàng thứ tư được dịch chuyển sang phải ba byte.

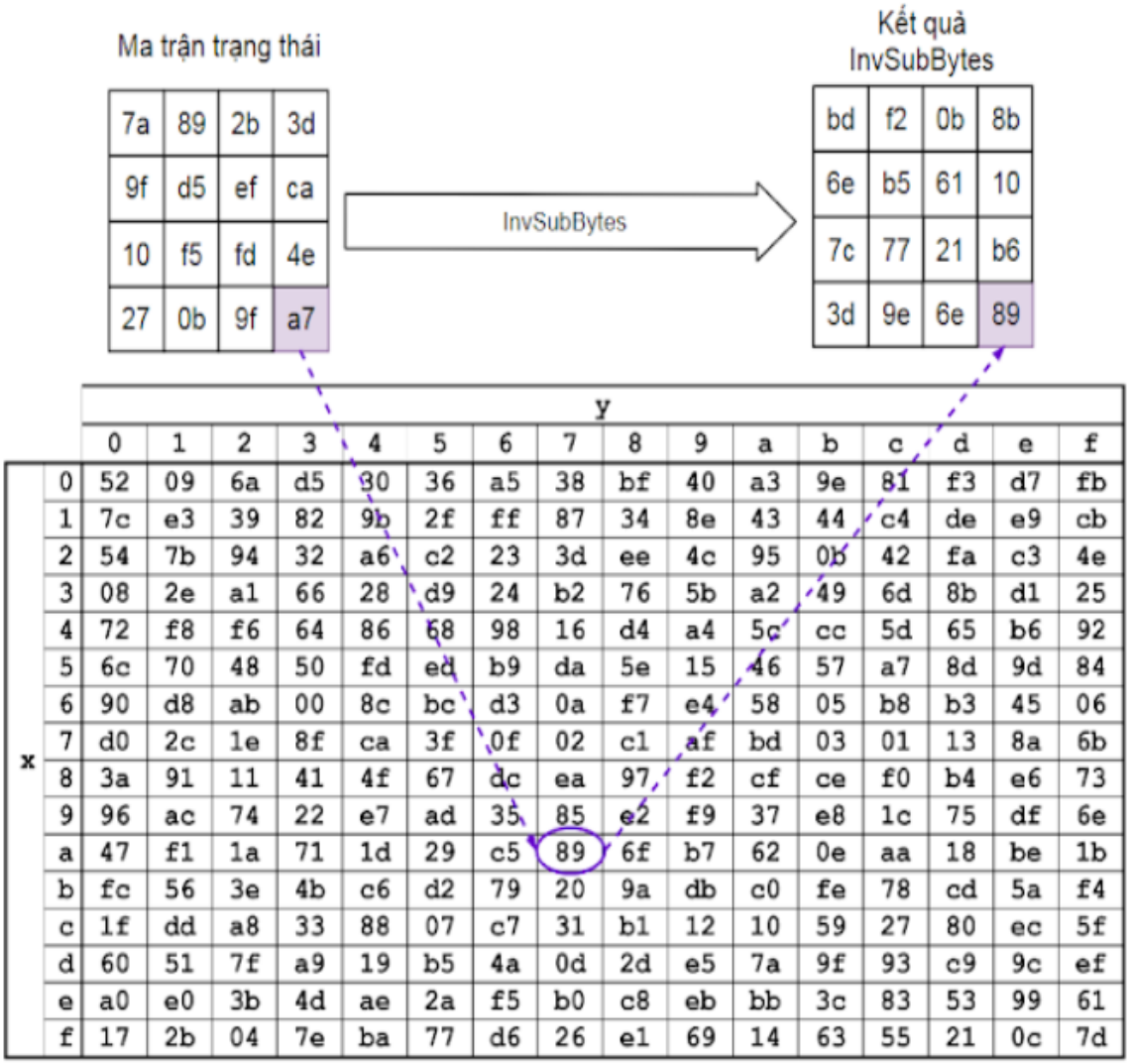
2.2.b. INVSUBBYTES

Chức năng InvSubBytes là thực hiện thay thế từng byte của ma trận trạng thái, bằng một giá trị đã quy định trong chuẩn AES. Bảng quy định giá trị thay thế cho InvSubBytes gọi là S-box đảo (Inverse S-box)

Hàm InvSubBytes làm như sau:

1. Nhận đầu vào là một chuỗi hex (ví dụ: "2B7E1516").
2. Tách chuỗi hex thành các cặp ký tự (ví dụ: "2B", "7E", "15", "16").
3. Đối với mỗi cặp ký tự, chuyển đổi chúng thành các chỉ số tương ứng trong bảng tra cứu invSbox (bảng tra cứu ngược) và lấy giá trị tại vị trí đó.
4. Ghép kết quả các giá trị được thay thế thành một chuỗi hex mới, chính là kết quả của việc giải mã.

Ví dụ, byte cần thay thế là Ha7 thì dò ở hàng "a" và cột số 7 trong bảng S-box đảo sẽ được kết quả là H89.



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

18 InvSubbytes

2.2.c. INVMIXCOLUMNS

Phép biến đổi ngược InvMixColumns: Là phép biến đổi ngược với phép biến đổi MixColumns. InvMixColumns cũng thực hiện thao tác theo từng cột của trạng thái, xem mỗi cột như một đa thức bậc 3 gồm 4 hạng tử trên trường GF(2^8). Các cột của phép InvMixColumns được nhân theo modulo $\mathbf{x^4 + 1}$ với đa thức nghịch đảo a(x) chính là đa thức $\mathbf{a^{-1}(x)}$ được định nghĩa:

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}$$

- Có thể biểu diễn bằng 1 phép nhân ma trận như sau:

$$s'(x) = a^{-1}(x) \otimes s(x)$$

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix}$$

- Kết quả là 4 byte trong mỗi cột sẽ được thay thế theo công thức như sau:

$$s'_{0,c} = (\{0e\} \bullet s_{0,c}) \oplus (\{0b\} \bullet s_{1,c}) \oplus (\{0d\} \bullet s_{2,c}) \oplus (\{09\} \bullet s_{3,c})$$

$$s'_{1,c} = (\{09\} \bullet s_{0,c}) \oplus (\{0e\} \bullet s_{1,c}) \oplus (\{0b\} \bullet s_{2,c}) \oplus (\{0d\} \bullet s_{3,c})$$

$$s'_{2,c} = (\{0d\} \bullet s_{0,c}) \oplus (\{09\} \bullet s_{1,c}) \oplus (\{0e\} \bullet s_{2,c}) \oplus (\{0b\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{0b\} \bullet s_{0,c}) \oplus (\{0d\} \bullet s_{1,c}) \oplus (\{09\} \bullet s_{2,c}) \oplus (\{0e\} \bullet s_{3,c})$$

Ví dụ:

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

•

0e	0b	0d	09
09	0e	0b	0d
0d	09	0e	0b
0b	0d	09	0e

=

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

❖ Cách xử lí tương tự ở MixColumns

Phần 3. Cơ sơ toán học và hiệu suất

3.1. Phép toán trường hữu hạn:

Phép toán trên trường hữu hạn GF(2^8) (hoặc GF(256)GF(256)) là một phần quan trọng của lĩnh vực mã hóa và mã hóa tin nhắn. Trong trường này, các phần tử là các số từ 0 đến 255, được biểu diễn bằng các số nhị phân 8-bit. Phép toán cơ bản trong GF(2^8) bao gồm cộng và nhân.

Phép Cộng : Tổng 2 đa thức f(x) và g(x), kí hiệu f(x) ⊕g(x), được đinhj nghĩa như sau:

f(x) ⊕g(x)= f(x) + g(x)

Phép Nhân: Tích 2 đa thức f(x) và g(x), kí hiệu f(x) ⊗ g(x), được đinhj nghĩa như sau:

f(x) ⊗ g(x)= [f(x) x g(x)] mod m(x)

Ví dụ : Thực hiện phép cộng và nhân hai số {57} và {83} GF(2^8).

*Phép cộng:

{57} = 0101.0111 tương đương f (x) = x^6 + x^4 + x^2 +x+1

{83} = 1000.0011 tương đương g(x) = x^7 + x+1.

f(x)+g(x)= x^6 + x^4 + x^2 +x+1+ x^7 + x+1=x^7+x^6+x^4+x^2=11010100

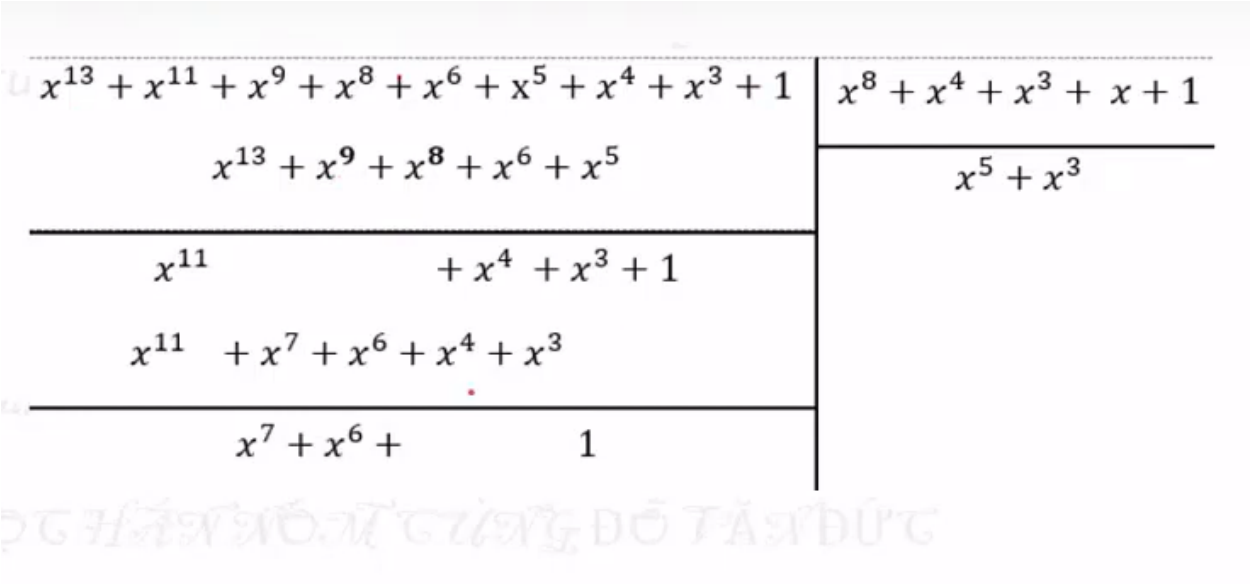
*Phép nhân:

f(x) x g(x) = x^13 + x^11 + x^9 + x^8 + x^7 + x^7 + x^5 + x^3 + x^2 + x + x^6 + x^2 + x^2 + x + 1

= x^13 + x^11 + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1

Thực hiện f(x) x g(x) mod (x)

(x^13 + x^11 + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1)mod(x^8+ x^4+ X^3+ x+1)=x^7+x^6+x^1=11000001



3.2. <u>Hiệu suất</u>

✓ **Tốc độ mã hóa và giải mã** AES là một thuật toán mã hóa đối xứng, nghĩa là cả quá trình mã hóa và giải mã đều cần thực hiện các bước giống nhau. Tuy nhiên, trong một số trường hợp, tốc độ mã hóa có thể được ưu tiên hơn tốc độ giải mã hoặc ngược lại, tùy thuộc vào yêu cầu cụ thể của ứng dụng.

✓ AES hỗ trợ ba kích thước khóa khác nhau. Kích thước khóa có thể ảnh hưởng đến hiệu suất của thuật toán, vì việc xử lý các khối lớn hơn hoặc sử dụng các khóa lớn hơn có thể đòi hỏi nhiều tài nguyên tính toán hơn.

✓ Trong một số trường hợp, việc tăng cường bảo mật có thể làm giảm hiệu suất và ngược lại. Các cải tiến bảo mật như việc thực hiện thêm vòng lặp hoặc sử dụng khóa lớn hơn có thể làm tăng thời gian xử lý.

Phần 4. Demo

```
+---------------MENU---------------+
| 1  . ma hoa du lieu               |
| 2  . giai ma du lieu              |
| 0  . Thoat                        |
+-----------------------------------+
Nhap lua chon: 1
Nhap chuoi can ma hoa: CongThuongTp.HCM
Nhap khoa: CongNgheThongTin
Chuoi da ma hoa: 5f85346a465348963ec4e7fb83ea6a12
```

Sử dụng khoá *CongNgheThongTin* để mã hoá chuỗi *CongThuongTp.HCM*.

Sau khi thực thi mã hoá xong chúng ta sẽ nhận được một chuỗi đã mã hoá:

- 5f85346a465348963ec4e7fb83ea6a12

❖ Sau khi nhận được chuỗi dữ liệu đã mã hoá ta sẽ sử dụng thuật toán giải mã với khoá sử dụng khi mã hoá để giải mã.

```
+---------------MENU---------------+
| 1  . ma hoa du lieu               |
| 2  . giai ma du lieu              |
| 0  . Thoat                        |
+-----------------------------------+
Nhap lua chon: 2
Nhap chuoi can giai ma: 5f85346a465348963ec4e7fb83ea6a12
Nhap khoa: CongNgheThongTin
Chuoi da giai hoa: CongThuongTp.HCM
```

- Kết quả giải mã ta sẽ nhận được: *CongThuongTp.HCM*

Phụ lục

Tài liệu tham khảo

[1]. [AES] Bài 1 - Lý thuyết về mã hóa AES-128 ~ VLSI TECHNOLOGY (nguyenquanicd.blogspot.com)

[2]. AES (Advanced Encryption Standard) Lý thuyết và bài tập - YouTube

[3]. AES là gì? Ứng dụng và nguyên lý hoạt động của AES (bizflycloud.vn)

[4]. Mã hóa AES là gì? Đặc điểm, tính năng và cách hoạt động (cmccloud.vn)

[5]. Tiêu chuẩn mã hóa dữ liệu AES là gì và các chế độ hoạt động của AES phần 1 (viettelidc.com.vn)

[6]. Sự khác biệt giữa DES (Tiêu chuẩn mã hóa dữ liệu) và AES (Tiêu chuẩn mã hóa nâng cao) (gadget-info.com)

[7]. AES là gì? Đặc điểm & cách hoạt động của mã hóa AES | BKHOST

[8]. Tiêu chuẩn mã hóa dữ liệu AES là gì và các chế độ hoạt động (hoanghamobile.com)

[9]. Cấu trúc và thuật toán Advanced Encryption Standard (Chuẩn mã hóa nâng cao) (viblo.asia)