TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN
****

BÁO CÁO THỰC NGHIỆM
HỌC PHẦN: AN NINH MẠNG

Đề tài:

CHUẨN AES VÀ ỨNG DỤNG TRONG MÃ HÓA DỮ LIỆU

GVHD:	TS. Phạm Văn Hiệp		
Lớp:	20241IT6070001		
Nhóm:	10		
Sinh viên:	Nguyễn Văn Hậu Giang	-	2022604193
	Khổng Thị Linh	-	2022603748
	Phạm Việt Long	-	2022600752
	Lê Tuấn Minh	-	2022605662
	Tống Đăng Quang	-	2022603783

Hà Nội, năm 2024

LỜI CẢM ƠN

Lời đầu tiên, nhóm 10 xin được gửi lời cảm ơn chân thành nhất đến thầy Phạm Văn Hiệp - người đã truyền tải nội dung bộ môn An ninh mạng trong kỳ học này cho chúng em. Trong quá trình học tập, thầy không chỉ đơn thuần giúp sinh viên tiếp cận được nội dung môn học, mà còn chia sẻ những kinh nghiệm hết sức quý báu với sinh viên. Hình thức truyền tải nội dung đặc biệt của thầy đã giúp tất cả sinh viên trong lớp ai cũng có cơ hội được đưa ra quan điểm về nội dung bài học. Điều đó đã giúp sinh viên trở nên chủ động hơn trong việc tìm hiểu kiến thức để trình bày trước lớp nội dung bài học đã chuẩn bị, từ đó giúp sinh viên hình thành và biết bảo vệ quan điểm đúng đắn của mình, đồng thời qua những góp ý của thầy và các sinh viên khác trong lớp giúp rút ra kinh nghiệm, sửa chữa những quan điểm sai về nội dung bài học. Thầy đã thành công trong việc định hướng sinh viên tự khám phá nội dung bài học, cùng với đó là sinh viên đã có cả một học kỳ được rèn giũa kỹ năng mềm cho bản thân mình.

Trong quá trình thực hiện đề tài "Chuẩn AES và ứng dụng trong mã hóa dữ liệu", nhóm đã vận dụng kiến thức được học trong những tuần lên lớp tìm hiểu cùng với thầy. Hơn thế nữa, sự tận tình chỉ dẫn của thầy đã giúp nhóm đi đúng hướng ban đầu đề ra và hoàn thành báo cáo thực nghiệm này.

Một lần nữa, chúng em xin chân thành cảm ơn!

Nhóm thực hiện

Nhóm 10

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

TT	Thuật ngữ	Ý nghĩa
1	Add round key	Cộng với khóa vòng
2	AES (Advanced Encryption Standard)	Chuẩn mã hóa nâng cao
3	CIA (Confidentiality - Integrity - Availability)	Tính bí mật - Tính toàn vẹn - Tính sẵn sàng
4	DDoS	Tấn công từ chối dịch vụ
5	DES (Data Encryption Standard)	Tiêu chuẩn mã hóa dữ liệu
6	GUI (Graphical User Interface)	Giao diện người dùng đồ họa
7	IDE (Integrated Development Environment)	Môi trường phát triển tích hợp
8	JVM (Java Virtual Machine)	Môi trường chạy ảo
9	Key expansion	Mở rộng khóa
10	Malware	Phần mềm độc hại
11	Mix columns	Trộn cột
12	NIST (National Institute of Standards and Technology)	Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ
13	OOP (Object-Oriented Programming)	Lập trình hướng đối tượng
14	Phishing	Lừa đảo
15	Ransomware	Tấn công mã hóa tống tiền
16	Shift rows	Dịch dòng
17	Substitute bytes	Thay thế byte

LỜI NÓI ĐẦU

Trong thời đại công nghệ bùng nổ, thông tin đã trở thành một trong những tài sản quý giá nhất của con người. Tuy nhiên, sự phát triển nhanh chóng của hệ thống mạng và các nền tảng số cũng kéo theo những nguy cơ mất an toàn thông tin ngày càng phức tạp và khó lường. Việc bảo mật dữ liệu không chỉ là yêu cầu bắt buộc đối với các tổ chức lớn như ngân hàng, cơ quan chính phủ, mà còn là mối quan tâm của mỗi cá nhân khi giao dịch và lưu trữ thông tin trên không gian mạng.

Trong số các phương pháp bảo mật, mã hóa dữ liệu được xem là một giải pháp hiệu quả và đáng tin cậy nhất. Trong đó, Advanced Encryption Standard (AES) đã chứng minh vai trò vượt trội của mình như một chuẩn mã hóa quốc tế, được sử dụng rộng rãi trong nhiều lĩnh vực nhờ tính bảo mật cao, hiệu năng tốt và khả năng ứng dụng linh hoạt.

Đề tài "Chuẩn AES và ứng dụng trong mã hóa dữ liệu" tập trung vào việc xây dựng chương trình mã hóa và giải mã mật mã hóa công khai AES. Đề tài được xây dựng thành ba chương với nội dung chính như sau:

- Chương 1: Tổng quan về đề tài nghiên cứu
- Chương 2: Kết quả nghiên cứu
- Chương 3: Kết luận và bài học kinh nghiệm

Qua quá trình thực hiện đề tài, chúng em sẽ có cơ hội tìm hiểu sâu về chuẩn mã hóa AES và vai trò của nó trong lĩnh vực bảo mật dữ liệu. Từ việc phân tích chi tiết và cài đặt thuật toán, chúng em hiểu rõ hơn về cách AES đảm bảo tính an toàn và hiệu quả trong việc bảo vệ thông tin. Đề tài không chỉ giúp chúng em nâng cao kiến thức chuyên môn mà còn rèn luyện kỹ năng nghiên cứu và tư duy phản biện, tạo nền tảng cho những nghiên cứu và ứng dụng trong tương lai.

CHƯƠNG 1. TỔNG QUAN

1.1. Tổng quan về an ninh mạng

1.1.1. Khái niệm về an ninh mạng

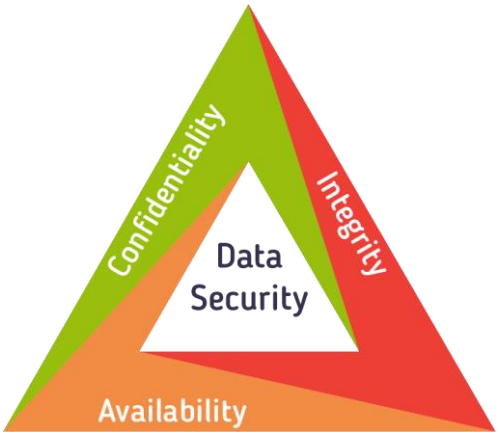
An ninh mạng là tập hợp các biện pháp, công nghệ và quy trình được thiết kế để bảo vệ an toàn cho máy tính, mạng, ứng dụng phần mềm, hệ thống và dữ liệu khỏi các mối đe dọa tiềm ẩn từ không gian mạng.

Trong kỷ nguyên kỹ thuật số hiện nay, an ninh mạng không chỉ là một bộ phận hỗ trợ, mà còn là nền tảng thiết yếu để duy trì sự an toàn và bền vững trong mọi hoạt động.

Vì dữ liệu ngày càng phổ biến và ngày càng có nhiều người làm việc cũng như kết nối từ mọi nơi, nên những kẻ xấu đã đối phó bằng cách phát triển các phương pháp tinh vi để có được quyền truy nhập vào các tài nguyên cũng như lấy cắp dữ liệu, phá hoại doanh nghiệp hoặc tống tiền. Mỗi năm số cuộc tấn công tăng lên và những kẻ tấn công tiếp tục phát triển các phương pháp tránh bị phát hiện mới. Một chương trình an ninh mạng hiệu quả thường bao gồm con người, quy trình và giải pháp công nghệ cùng nhau giảm rủi ro gây gián đoạn kinh doanh, tổn thất tài chính và tổn thất uy tín từ các cuộc tấn công.

1.1.2. Mục tiêu của an ninh mạng

An ninh mạng là sự bảo vệ cho các hệ thống nhằm đạt được được các mục tiêu đó là duy trì tính bí mật, tính toàn vẹn và tính sẵn sàng của tài nguyên hệ thống.



Hình 1.1: Tam giác bảo mật CIA

Tính bí mật (Confidentiality) là sự ngăn chặn việc thông tin bị truy cập hoặc tiết lộ trái phép. Đảm bảo chỉ những người hoặc hệ thống được ủy quyền mới có thể truy cập vào dữ liệu. Mọi truy cập trái phép đều phải bị ngăn chặn.

Tính toàn vẹn (Integrity) là sự phát hiện và ngăn chặn việc sửa đổi trái phép về dữ liệu, thông tin và hệ thống. Dữ liệu cần phải chính xác, đầy đủ và không bị chỉnh sửa bởi các yếu tố không được phép. Việc thay đổi hoặc xóa dữ liệu có thể gây ra những tổn hại nghiêm trọng.

Tính sẵn sàng (Availability) là sự đảm bảo dữ liệu và hệ thống luôn sẵn sàng cho người dùng hợp pháp truy cập khi cần thiết. Nếu hệ thống không thể truy cập được, các hoạt động sẽ bị gián đoạn và ảnh hưởng nghiêm trọng.

1.1.3. Tầm quan trọng của an ninh mạng

An ninh mạng là yếu tố vô cùng quan trọng trong thời đại số hiện nay. Với sự phát triển nhanh chóng của công nghệ thông tin và việc kết nối toàn cầu qua Internet, an ninh mạng đang dần trở nên quan trọng hơn bao giờ hết.

Bảo vệ dữ liệu cá nhân: Thông tin cá nhân, như tên, địa chỉ, số thẻ tín dụng và mật khẩu, có thể bị lộ nếu không có hệ thống bảo mật thích hợp. Các cuộc tấn công mạng có thể dẫn đến việc đánh cắp thông tin cá nhân, gây tổn thất cho người dùng. Bảo vệ dữ liệu cá nhân không chỉ giúp tránh các cuộc tấn công mà còn giữ gìn sự riêng tư của mỗi cá nhân trong không gian mạng.

Bảo vệ tài sản doanh nghiệp: Các tổ chức và công ty sở hữu nhiều dữ liệu nhạy cảm như thông tin khách hàng, dữ liệu tài chính, nghiên cứu và phát triển sản phẩm. Nếu bị xâm nhập, dữ liệu này có thể bị rò rỉ hoặc đánh cắp, gây ảnh hưởng nghiêm trọng đến uy tín và tài chính của doanh nghiệp. Việc bảo vệ an ninh mạng giúp ngăn chặn các mối đe dọa này và duy trì hoạt động ổn định của doanh nghiệp.

Ngăn chặn các cuộc tấn công mạng: Các cuộc tấn công mạng như virus, ransomware, tấn công từ chối dịch vụ (DDoS), … có thể làm gián đoạn hoạt động của các hệ thống công nghệ, gây thiệt hại nặng nề cho cá nhân và tổ chức. Những cuộc tấn công này có thể dẫn đến mất mát dữ liệu, thiệt hại về tài chính và làm gián đoạn các dịch vụ quan trọng. Việc đảm bảo an ninh mạng giúp bảo vệ hệ thống khỏi các tấn công này.

Bảo vệ quyền lợi quốc gia: Các quốc gia có thể là mục tiêu của các cuộc tấn công mạng nhắm vào cơ quan chính phủ, hệ thống hạ tầng quan trọng hoặc các tổ chức chiến lược. Việc bảo vệ an ninh mạng quốc gia là rất quan trọng để duy trì sự ổn định và bảo vệ an ninh quốc gia, đồng thời ngăn ngừa các mối đe dọa từ các đối tượng tấn công có ý đồ xấu.

Đảm bảo tính toàn vẹn và sẵn sàng của hệ thống: An ninh mạng giúp đảm bảo rằng các hệ thống công nghệ hoạt động ổn định, không bị xâm nhập hay thay đổi trái phép, từ đó duy trì sự liên tục trong hoạt động của cá nhân, tổ chức và xã hội. Điều này đặc biệt quan trọng đối với các hệ thống thiết yếu như điện lực, giao thông và chăm sóc sức khỏe.

1.1.4. Các mối đe dọa an ninh mạng

Sự gia tăng của các cuộc tấn công mạng là kết quả tất yếu của quá trình chuyển đổi số nhanh chóng, nhưng nhiều tổ chức và doanh nghiệp lại chưa có sự đầu tư đầy đủ cho an toàn thông tin. Trong quá trình cải tiến công nghệ, các tổ chức này vô tình để lộ những điểm yếu do không chú trọng đến an ninh mạng ngay từ giai đoạn thiết kế. Chính những điểm yếu này đã biến họ thành mục tiêu dễ dàng cho tin tặc, khiến các cuộc tấn công mạng ngày càng trở nên tinh vi hơn về cả quy mô và phương thức.

Phần mềm độc hại (Malware): Phần mềm độc hại bao gồm virus, worm, trojan, ransomware, spyware và adware. Các phần mềm này xâm nhập vào hệ thống máy tính mà không được phép, có thể đánh cắp thông tin, phá hoại hệ thống, hoặc chiếm quyền kiểm soát máy tính. Trong đó, malware thường xâm nhập qua các tệp đính kèm email, website không an toàn, hoặc phần mềm bị nhiễm độc. Một số loại malware có thể tự nhân bản và lan truyền qua mạng nội bộ hoặc Internet.

Tấn công từ chối dịch vụ (DDos): DDoS là một trong những hình thức tấn công phổ biến nhất. Kẻ tấn công sử dụng nhiều máy tính bị lây nhiễm để gửi lượng lớn yêu cầu đến một máy chủ hoặc hệ thống, làm cho nó quá tải và không thể hoạt động bình thường. Mục tiêu của DDoS là làm gián đoạn hoặc ngưng trệ hoạt động của dịch vụ, khiến người dùng hợp pháp không thể truy cập vào dịch vụ hoặc thông tin họ cần.

Lừa đảo (Phishing): Phishing là hành vi lừa đảo qua email hoặc trang web giả mạo nhằm đánh cắp thông tin cá nhân, thông tin đăng nhập, hoặc dữ liệu tài chính của người dùng. Kẻ tấn công thường giả mạo các tổ chức hoặc cá nhân tin cậy để lừa người dùng cung cấp thông tin nhạy cảm. Các email hoặc website phishing thường rất tinh vi, khiến người dùng khó phân biệt với email hoặc trang web hợp pháp. Chúng có thể chứa liên kết đến các trang web giả mạo hoặc yêu cầu người dùng cung cấp thông tin cá nhân.

Tấn công khai thác lỗ hổng: Kẻ tấn công có thể tận dụng các lỗ hổng bảo mật trong phần mềm, hệ điều hành, hoặc ứng dụng để xâm nhập vào hệ thống, chiếm quyền kiểm soát hoặc đánh cắp dữ liệu.

Tấn công qua mạng xã hội: Tấn công qua mạng xã hội là hình thức tấn công mà kẻ tấn công thao túng tâm lý con người để lừa họ tiết lộ thông tin mật hoặc thực hiện các hành động không an toàn. Một ví dụ phổ biến là việc giả làm đồng nghiệp hoặc đối tác kinh doanh để yêu cầu người dùng cung cấp mật khẩu hoặc thông tin đăng nhập vào hệ thống.

Tấn công mã hóa tống tiền (Ransomware): Ransomware là một loại malware mã hóa các tệp dữ liệu quan trọng trên máy tính hoặc hệ thống mạng và yêu cầu một khoản tiền chuộc để giải mã chúng. Các doanh nghiệp và cá nhân bị tấn công ransomware có thể mất hoàn toàn quyền truy cập vào dữ liệu của mình, gây ra thiệt hại nghiêm trọng về tài chính và danh tiếng.

Tấn công xem giữa: Bên ngoài cố gắng truy cập trái phép vào mạng trong khi trao đổi dữ liệu.

Mối đe dọa nội bộ: Một rủi ro an ninh do nhân viên có ý định xấu trong một tổ chức gây ra, làm mất ổn định an ninh hệ thống từ bên trong.

1.1.5. Các biện pháp bảo vệ an ninh mạng

Theo khoản 1 Điều 5 Luật An ninh mạng 2018 có 13 biện pháp bảo vệ an ninh mạng bao gồm:

- Thẩm định an ninh mạng.
- Đánh giá điều kiện an ninh mạng.
- Kiểm tra an ninh mạng.
- Giám sát an ninh mạng.
- Ứng phó, khắc phục sự cố an ninh mạng.
- Đấu tranh bảo vệ an ninh mạng.
- Sử dụng mật mã để bảo vệ thông tin mạng.
- Ngăn chặn, yêu cầu tạm ngừng, ngừng cung cấp thông tin mạng; đình chỉ, tạm đình chỉ các hoạt động thiết lập, cung cấp và sử dụng mạng viễn thông, mạng Internet, sản xuất và sử dụng thiết bị phát, thu phát sóng vô tuyến theo quy định của pháp luật.
- Yêu cầu xóa bỏ, truy cập xóa bỏ thông tin trái pháp luật hoặc thông tin sai sự thật trên không gian mạng xâm phạm an ninh quốc gia, trật

tự, an toàn xã hội, quyền và lợi ích hợp pháp của cơ quan, tổ chức, cá nhân.

- Thu thập dữ liệu điện tử liên quan đến hoạt động xâm phạm an ninh quốc gia, trật tự, an toàn xã hội, quyền và lợi ích hợp pháp của cơ quan, tổ chức, cá nhân trên không gian mạng.
- Phong tỏa, hạn chế hoạt động của hệ thống thông tin; đình chỉ, tạm đình chỉ hoặc yêu cầu ngừng hoạt động của hệ thống thông tin, thu hồi tên miền theo quy định của pháp luật.
- Khởi tố, điều tra, truy tố, xét xử theo quy định của Bộ luật Tố tụng hình sự.
- Biện pháp khác theo quy định của pháp luật về an ninh quốc gia, pháp luật về xử lý vi phạm hành chính.

1.2. Lý do chọn đề tài

Trong thời đại công nghệ số, lượng dữ liệu được lưu trữ, truyền tải và xử lý ngày càng lớn, kéo theo đó là nguy cơ mất an toàn thông tin. Việc bảo vệ dữ liệu trước các mối đe dọa như tấn công mạng, đánh cắp dữ liệu và xâm phạm quyền riêng tư là một yêu cầu cấp thiết. Trong bối cảnh đó, các phương pháp mã hóa đóng vai trò trọng yếu trong việc đảm bảo tính bí mật, toàn vẹn và sẵn sàng của dữ liệu và hệ thống.

AES (Advanced Encryption Standard), được phê chuẩn bởi Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (NIST), là một trong những thuật toán mã hóa đối xứng phổ biến và hiệu quả nhất hiện nay. Với khả năng bảo mật mạnh mẽ, tốc độ xử lý cao và hiệu quả trong nhiều ứng dụng, AES đã được công nhận và sử dụng rộng rãi trên toàn cầu, trở thành lựa chọn hàng đầu trong nhiều hệ thống bảo mật. AES không chỉ được sử dụng trong các hệ thống lớn như ngân hàng, thương mại điện tử hay các cơ quan chính phủ mà còn xuất hiện trong các ứng dụng đời thường như mã hóa dữ liệu cá nhân trên điện thoại di động, đĩa cứng và truyền thông qua Internet. Tìm hiểu và áp dụng AES sẽ giúp nâng cao kỹ năng thực hành trong bảo mật thông tin, đồng thời mở ra cơ hội phát triển các hệ thống an toàn và tối ưu hơn.

Nghiên cứu chuẩn AES mang lại cơ hội hiểu sâu về cơ chế hoạt động của các thuật toán mã hóa hiện đại. Đề tài cũng là cơ hội để triển khai các ứng dụng thực tế, như phát triển chương trình mã hóa, qua đó kiểm chứng lý thuyết và nâng cao năng lực ứng dụng công nghệ.

Bên cạnh đó, việc xây dựng chương trình mã hóa và giải mã AES cũng là một cơ hội cho việc phát triển, nâng cao kỹ năng lập trình và hiểu sâu hơn về bảo mật thông tin.

Đề tài không chỉ mang tính học thuật mà còn góp phần nâng cao nhận thức của cộng đồng về bảo mật dữ liệu, khuyến khích sử dụng các biện pháp an toàn trong lưu trữ và truyền thông thông tin. Điều này đặc biệt quan trọng trong bối cảnh các mối đe dọa an ninh mạng ngày càng gia tăng.

1.3. Nội dung nghiên cứu

Hệ mã hóa khóa bí mật: Các khái niệm cơ bản, cách hoạt động, mô hình và các yêu cầu khi sử dụng.

Chuẩn mã nâng cao AES: Giới thiệu, cách hoạt động, độ an toàn, ưu nhược điểm và ứng dụng.

Ngôn ngữ lập trình sử dụng cho việc xây dựng chương trình mã hóa và giải mã AES: Java, C#.

Kỹ thuật để xây dựng ra một chương trình mã hóa và giải mã: Cách xây dựng thuật toán, cách xây dựng giao diện phù hợp và dễ thao tác.

Triển khai, kiểm thử và sửa lỗi chương trình: Thiết kế kịch bản cho chương trình, chạy thử chương trình, kiểm tra tất cả các tính năng đã xây dựng và sửa lỗi nếu có.

1.4. Các kiến thức cơ sở

1.4.1. Cơ sở toán học

1.4.1.1. Phép XOR

Phép XOR (viết tắt của Exclusive OR) là một phép toán logic được sử dụng trong nhiều lĩnh vực, đặc biệt là trong khoa học máy tính, điện tử và mật mã học.

Phép XOR lấy hai dãy bit có cùng độ dài và thực hiện XOR trên mỗi cặp bit tương ứng. Phép XOR trả về đúng (True) nếu hai toán hạng khác nhau và sai (False) nếu hai toán hạng giống nhau.

Bảng 1.1: Bảng chân lý cho phép XOR

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Ví dụ, thực hiện phép XOR đối với số hệ nhị phân: 0101 XOR 0011.

Bảng 1.2: Ví dụ phép XOR đối với số hệ nhị phân

A	0	1	0	1
B	0	0	1	1
A XOR B	0	1	1	0

Ví dụ, thực hiện phép XOR đối với số hệ 16: A5 XOR 3F.

Bảng 1.3: Ví dụ phép XOR đối với số hệ 16

Chuyển đổi sang hệ nhị phân sau đó thực hiện phép XOR.								
$A5_{16}$	1	0	1	0	0	1	0	1
$3F_{16}$	0	0	1	1	1	1	1	1
A5 XOR 3F	1	0	0	1	1	0	1	0
Chuyển đổi kết quả về lại hệ 16: 1001 1010 = 9A								

1.4.1.2. Phép nhân ma trận

Phép nhân hai ma trận chỉ thực hiện được khi số lượng cột trong ma trận thứ nhất phải bằng số lượng hàng trong ma trận thứ hai. Ma trận kết quả, được gọi là ma trận tích, có số lượng hàng của ma trận đầu tiên và số cột của ma trận thứ hai.

Nếu ma trận *A* có kích thước $(m \times n)$ và ma trận *B* có kích thước $(n \times p)$ thì ma trận tích $C = A \times B$ có kích thước $(m \times p)$. Phần tử đứng ở hàng thứ *i*, cột thứ *j* được xác định bởi công thức:

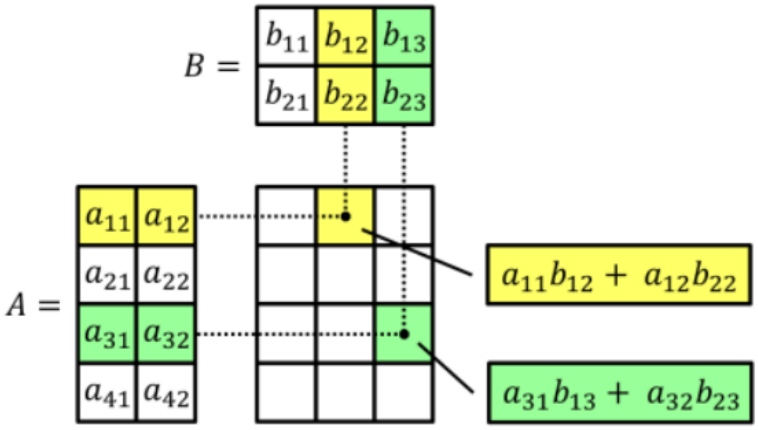
$$C_{ij} = A_{i1}B_{1j} + A_{i2}B_{2j} + \ldots + A_{in}B_{nj} = \sum_{k=1}^{n} A_{ik}B_{kj} \text{ (Với } 1 \leq i \leq m; 1 \leq j \leq p)$$

Hay có thể viết:

$$C_{ij} = \begin{bmatrix} a_{i1} & a_{i2} & \ldots & a_{in} \end{bmatrix} \begin{bmatrix} b_{1j} \\ b_{2j} \\ \vdots \\ b_{nj} \end{bmatrix}$$

Tức là phần tử ở hàng thứ *i*, cột thứ *j* của *C* là tích của vector hàng thứ *i* của ma trận *A* với vector cột thứ *j* của ma trận *B*.

Minh họa tích ma trận *AB* của hai ma trận *A* và *B*:



Hình 1.2: Minh họa tích ma trận AB của hai ma trận A và B

Ví dụ, tính $C = AB$, biết: $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 2 & 0 \\ 1 & 3 \end{bmatrix}$

- $C[1][1] = 1 \times 2 + 2 \times 1 = 4$
- $C[1][2] = 1 \times 0 + 2 \times 3 = 6$
- $C[2][1] = 3 \times 2 + 4 \times 1 = 10$
- $C[2][2] = 3 \times 0 + 4 \times 3 = 12$

Ta được ma trận tích: $C = \begin{bmatrix} 4 & 6 \\ 10 & 12 \end{bmatrix}$

1.4.2. Ngôn ngữ lập trình

1.4.2.1. Ngôn ngữ Java

Hình 1.3: Ngôn ngữ lập trình Java

Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sản phẩm gia dụng, và có tên là Oak. Java được chính thức phát hành năm 1994, đến năm 2010 được Oracle mua lại từ Sun MicroSystem.

Java là ngôn ngữ lập trình đa nền tảng (cross-platform), được phát triển bởi James Gosling tại Sun Microsystems (nay là Oracle Corporation). Ngôn ngữ lập trình này ra đời vào năm 1995 và được thiết kế để có thể chạy trên các nền tảng khác nhau, từ máy tính cá nhân đến thiết bị di động, các máy chủ và thiết bị nhúng.

Java sử dụng cấu trúc lập trình hướng đối tượng (Object-Oriented Programming - OOP) và được xây dựng trên cơ sở của ngôn ngữ lập trình C++. Nó cung cấp một môi trường chạy ảo (virtual machine) gọi là Java Virtual Machine (JVM), giúp các chương trình Java có thể chạy trên nhiều nền tảng khác nhau mà không cần phải biên dịch lại.

Đặc điểm của Java:

- Hướng đối tượng hoàn toàn.
- Có thể chạy trên nhiều nền tảng khác nhau.
- Quản lý bộ nhớ tự động.
- Hỗ trợ đa luồng.
- Tính bảo mật cao.
- Hỗ trợ thư viện và các công cụ mạnh mẽ.

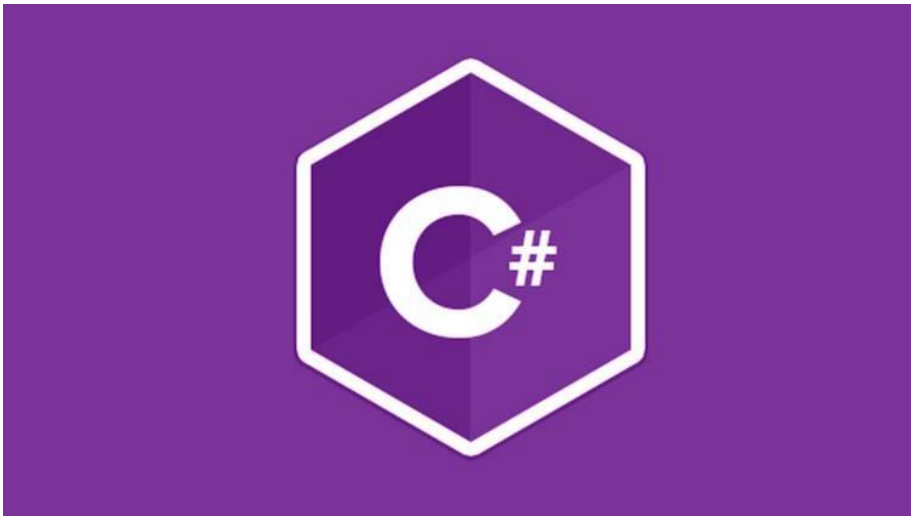
Ưu điểm của Java:

- Độ tin cậy cao.
- Tính đa nền tảng.
- Quản lý bộ nhớ tự động.
- Công cụ phát triển phong phú.
- Hỗ trợ đa luồng.

Nhược điểm của Java:

- Tốc độ chậm hơn so với các ngôn ngữ lập trình gần sát với phần cứng, chẳng hạn như C hoặc C++.
- Java có thể chạy trên nhiều nền tảng khác nhau, nhưng ứng dụng này có thể cần đến một trình biên dịch hoặc máy ảo Java riêng biệt để có thể chạy trên các thiết bị di động.
- Sử dụng bộ nhớ lớn hơn so với một số ngôn ngữ lập trình khác.
- Cú pháp phức tạp hơn so với một số ngôn ngữ lập trình khác

1.4.2.2. Ngôn ngữ C#



Hình 1.4: Ngôn ngữ lập trình C#

C# (hay C sharp) là một ngôn ngữ lập trình đơn giản, được phát triển bởi đội ngũ kỹ sư của Microsoft vào năm 2000. C# là ngôn ngữ lập trình hiện đại, hướng đối tượng và được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java.

Trong các ứng dụng Windows truyền thống, mã nguồn chương trình được biên dịch trực tiếp thành mã thực thi của hệ điều hành.

Trong các ứng dụng sử dụng .NET Framework, mã nguồn chương trình (C#, VB.NET) được biên dịch thành mã ngôn ngữ trung gian MSIL (Microsoft Intermediate Language). Sau đó mã này được biên dịch bởi Common Language Runtime (CLR) để trở thành mã thực thi của hệ điều hành.

C# với sự hỗ trợ mạnh mẽ của .NET Framework giúp cho việc tạo một ứng dụng Windows Forms hay WPF (Windows Presentation Foundation), phát triển game, ứng dụng Web, ứng dụng Mobile trở nên rất dễ dàng.

Đặc điểm của C#:

- Là một ngôn ngữ thuần hướng đối tượng.
- Là ngôn ngữ khá đơn giản, chỉ có khoảng 80 từ khóa và hơn mười mấy kiểu dữ liệu được dựng sẵn.
- Cung cấp những đặc tính hướng thành phần (component-oriented) như Property, Event, …
- C# không khuyến khích sử dụng con trỏ như trong C++ nhưng khi thực sự muốn sử dụng thì phải đánh dấu đây là mã không an toàn (unsafe).

- C# có bộ Garbage Collector sẽ tự động thu gom vùng nhớ khi không còn sử dụng nữa.
- C# đã loại bỏ đa kế thừa trong C++ mà thay vào đó C# sẽ hỗ trợ thực thi giao diện interface.

Ưu điểm của C#:

- Gần gũi với các ngôn ngữ lập trình thông dụng (C++, Java, Pascal).
- Xây dựng dựa trên nền tảng của các ngôn ngữ lập trình mạnh nên thừa hưởng những ưu điểm của những ngôn ngữ đó.
- Cải tiến các khuyết điểm của C/C++.
- Dễ tiếp cận, dễ phát triển.
- Được sự chống lưng của .NET Framework.

Nhược điểm của C#:

- Nhược điểm lớn nhất của C# là chỉ chạy trên nền Windows và có cài .NET Framework.
- Thao tác đối với phần cứng yếu hơn so với ngôn ngữ khác. Hầu hết phải dựa vào Windows.

Tiểu kết chương 1

Chương 1 đã trình bày tổng quan về an ninh mạng, bao gồm khái niệm, mục tiêu, tầm quan trọng, các mối đe dọa, và các biện pháp bảo vệ. Nội dung này nhấn mạnh sự cần thiết của việc đảm bảo an toàn thông tin, đặc biệt trong bối cảnh các mối nguy cơ ngày càng gia tăng và đa dạng trong môi trường số.

Lý do chọn đề tài "Chuẩn AES và ứng dụng trong mã hóa dữ liệu" đã được đề cập, xuất phát từ vai trò quan trọng của AES (Advanced Encryption Standard) trong việc bảo vệ dữ liệu hiện nay. Đây là một thuật toán mã hóa được ứng dụng rộng rãi nhờ tính bảo mật cao và hiệu suất tốt, phù hợp với nhiều hệ thống và nền tảng khác nhau.

Nội dung nghiên cứu được định hướng rõ ràng, bao gồm việc tìm hiểu cấu trúc và hoạt động của chuẩn AES, cùng với ứng dụng thực tiễn trong việc bảo mật dữ liệu.

Bên cạnh đó, chương đã cung cấp các kiến thức cơ sở làm nền tảng cho nghiên cứu. Về cơ sở toán học: Các phép toán như XOR và phép nhân ma trận, đều là các thành phần quan trọng trong hoạt động của AES. Về ngôn ngữ lập trình: Java và C# được giới thiệu như các công cụ hỗ trợ triển khai thuật toán AES và xây dựng các ứng dụng bảo mật.

Những nội dung trên tạo cơ sở lý thuyết và thực tiễn vững chắc cho việc nghiên cứu chuyên sâu về chuẩn AES và ứng dụng trong mã hóa dữ liệu ở các chương tiếp theo.

CHƯƠNG 2. KẾT QUẢ NGHIÊN CỨU

2.1. Nghiên cứu và tìm hiểu về hệ mã hóa khóa bí mật

2.1.1. Các khái niệm cơ bản

Mã hóa khóa bí mật là một phương pháp mã hóa, trong đó cùng một khóa được sử dụng cho cả việc mã hóa và giải mã dữ liệu. Đây là một trong những hình thức mã hóa phổ biến nhất, được sử dụng rộng rãi để bảo vệ thông tin trong các hệ thống truyền thông và lưu trữ dữ liệu.

Người gửi và người nhận nhờ chia sẻ khóa chung K, mà họ có thể trao đổi bí mật với nhau. Ta xét hai hàm ngược nhau: E là hàm biến đổi bản rõ thành bản mã và D là hàm biến đổi bản mã trở về bản rõ. Giả sử X là văn bản cần mã hóa và Y là văn bản đã được biến đổi qua việc mã hóa. Khi đó ta kí hiệu:

$$Y = E_K(X)$$

$$X = D_K(Y)$$

Các hệ mã hóa khóa bí mật được phát minh ra trước các hệ mã hóa khóa công khai. Hiện nay các hệ mã hóa khóa bí mật và công khai vẫn tiếp tục phát triển và hoàn thiện. Mã công khai ra đời hỗ trợ mã bí mật chứ không thay thế nó, do đó mã bí mật đến nay vẫn được sử dụng.

Một số khái niệm cơ bản về mã hóa:

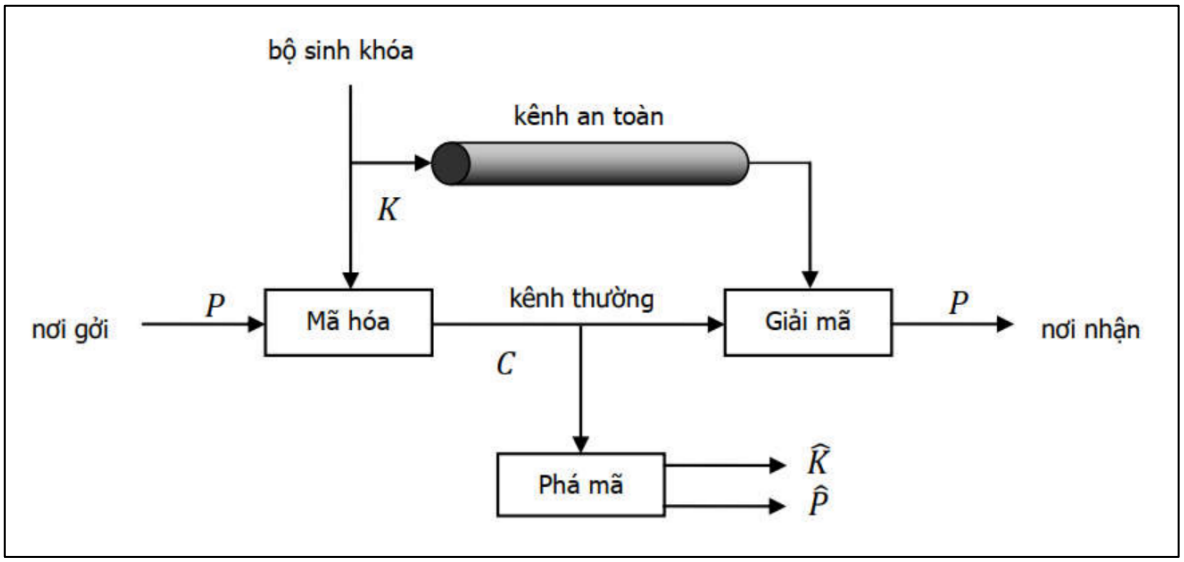
- Bản rõ X: Là bản tin gốc. Bản rõ có thể được chia nhỏ để có kích thước phù hợp.
- Bản mã Y: Là bản tin gốc đã được mã hóa. Ta thường xét phương pháp mã hóa mà không làm thay đổi kích thước của bản rõ, tức là chúng có cùng độ dài.
- Khóa K: Là thông tin tham số dùng để mã hóa, chỉ có người gửi và người nhận biết. Khóa độc lập với bản rõ và có độ dài phù hợp với yêu cầu bảo mật.
- Mã hóa E: Là quá trình chuyển bản rõ thành bản mã. Thông thường bao gồm việc áp dụng thuật toán mã hóa và một số quá trình xử lý thông tin kèm theo.
- Giải mã D: Là quá trình chuyển bản mã về bản rõ. Đây là quá trình ngược lại của mã hóa.
- Mật mã: Là chuyên nghành khoa học của Khoa học máy tính nghiên cứu về các nguyên lý và phương pháp mã hóa. Hiện nay người ta đưa ra nhiều chuẩn an toàn cho các lĩnh vực khác nhau.

- Thám mã: Nghiên cứu các nguyên lý và phương pháp giải mã mà không biết khóa. Thông thường khi đưa các mã mạnh ra làm chuẩn dùng chung, các mã đó được những kẻ thám mã cũng như những người phát triển mã tìm hiểu và nghiên cứu các phương pháp giải mã một phần bản mã với các thông tin không đầy đủ.
- Lý thuyết mã: Bao gồm cả mật mã và thám mã. Nó là một thể thống nhất để đánh giá một mã mạnh hay không.

2.1.2. Cơ chế hoạt động của hệ mã hóa khóa bí mật

Người gửi và người nhận nhờ chia sẻ khóa chung K, mà họ có thể trao đổi bí mật với nhau. Cơ chế hoạt động của hệ mã hóa khóa bí mật như sau:

- Người gửi sử dụng khóa bí mật chung K để mã hóa thông tin rồi gửi cho người nhận.
- Người nhận khi nhận được thông tin sẽ dùng chính khóa bí mật chung K để giải mã.



Hình 2.1: Mô hình của hệ mã hóa khóa bí mật

Đặc tính quan trọng của mã hóa khóa bí mật là khóa phải được giữ bí mật giữa người gửi và người nhận, hay nói cách khác khóa phải được chuyển một cách an toàn từ người gửi đến người nhận. Có thể đặt ra câu hỏi là nếu đã có một kênh an toàn để chuyển khóa như vậy thì tại sao không dùng kênh đó để chuyển bản tin, tại sao cần đến chuyện mã hóa? Câu trả lời là nội dung bản tin thì có thể rất dài, còn khóa thì thường là ngắn. Ngoài ra một khóa còn có thể áp dụng để truyền tin nhiều lần. Do đó nếu chỉ chuyển khóa trên kênh an toàn thì đỡ tốn kém chi phí.

2.1.3. Phân loại hệ mã hóa khóa bí mật

Mã hóa khóa bí mật có thể phân thành hai nhóm phụ đó là mã hóa theo khối và má hóa dòng.

Mã hóa hóa theo khối: Từng khối dữ liệu trong văn bản gốc ban đầu được thay thế bằng một khối dữ liệu khác có cùng độ dài. Độ dài mỗi khối gọi là kích thước khối và thường được tính bằng đơn vị bit. Một số thuật toán mã hóa khóa thông dụng đó là: DES, 3-DES, AES, RC5, RC6, 3-Way, CAST, …

Mã hóa dòng: Dữ liệu đầu vào được mã hóa từng bit một. Các thuật toán mã hóa dòng có tốc độ nhanh hơn các thuật toán mã hóa khối, được dùng khi khối lượng dữ liệu cần mã hóa chưa biết trước, ví dụ trong kết nối không dây. Có thể coi thuật toán mã hóa dòng là thuật toán mã hóa khối với kích thước mỗi khối là 1 bit. Một số thuật toán mã hóa dòng thông dụng đó là: RC4, A5/1, A5/2, Chameleon, …

2.1.4. Các yêu cầu đối với hệ mã hóa khóa bí mật

Một hệ mã hóa khóa bí mật có các đặc trưng là cách xử lý thông tin của thuật toán mã hóa, giải mã, tác động của khóa vào bản mã, độ dài của khóa. Mối liên hệ giữa bản rõ, khóa và bản mã càng phức tạp càng tốt, nếu tốc độ tính toán là chấp nhận được. Cụ thể hai yêu cầu để sử dụng an toàn mã khoá đối xứng là:

- Thuật toán mã hoá mạnh. Có cơ sở toán học vững chắc đảm bảo rằng mặc dù công khai thuật toán, mọi người đều biết, nhưng việc thám mã là rất khó khăn và phức tạp nếu không biết khóa.
- Khoá mật chỉ có người gửi và người nhận biết. Có kênh an toàn để phân phối khoá giữa các người sử dụng chia sẻ khóa. Mối liên hệ giữa khóa và bản mã là không nhận biết được.

2.1.5. Nhận xét về hệ mã hóa khóa bí mật

2.1.5.1. Ưu điểm

Ưu điểm nổi bật của mã hóa khóa bí mật là tốc độ lập mã, giải mã khá nhanh chóng. Hiện nay có nhiều phần mềm thương mại hỗ trợ thuật toán mã hóa khóa bí mật hữu hiệu và rất phổ dụng.

Ưu điểm thứ hai là tuy có nhiều nghiên cứu thám mã đã thực hiện nhưng với các thuật toán được cải tiến gần đây như 3-DES và nhất là AES thì độ bảo mật khá cao, trong thực tế việc phá mã cũng không dễ dàng.

Bên cạnh đó, việc sử dụng và triển khai các thuật toán mã hóa khóa bí mật cũng khá dễ dàng. Các thuật toán mã hóa khóa bí mật cũng tạo được độ bảo mật cao nếu khóa được chọn đủ mạnh và được quản lý tốt.

2.1.5.2. Nhược điểm

Nhược điểm lớn nhất của thuật toán mã hóa khóa bí mật là vấn đề chuyển giao chìa khóa giữa các đối tác, đặc biệt là trong môi trường mở. Quá trình phân phối và bảo mật khóa cho các thành viên rất khó khăn, đặc biệt trong các môi trường lớn.

Mã hóa khóa bí mật bộc lộ hạn chế khi thông tin mật cần được chia sẻ với một bên thứ hai vì khi đó cần phải chuyển giao chìa khóa cho đối tác mà việc chuyển giao chìa khóa trong môi trường mở có nhiều nguy cơ bị lộ và như vậy việc mã hóa về sau trở thành vô nghĩa! Do đó cần phản có kênh an toàn cho việc chia sẻ khóa bí mật giữa các bên.

Vì phải có một cặp khóa riêng cho mỗi cặp người sử dụng nên cần phải có số lượng lớn sự kết hợp các cặp khóa. Với n người sẽ cần $n(n-1)/2$ cặp khóa.

Một hạn chế nữa đó là mã hóa khóa bí mật không thể dùng cho mục đích xác thực giống như mã hóa khóa công khai.

2.2. Nghiên cứu và tìm hiểu về chuẩn mã nâng cao AES

2.2.1. Giới thiệu về AES

Từ cuối thập niên 1980, đầu thập niên 1990, xuất phát từ những lo ngại về độ an toàn và tốc độ thấp khi áp dụng bằng phần mềm, giới nghiên cứu đã đề xuất khá nhiều thuật toán mã hóa khối để thay thế DES. Những ví dụ tiêu biểu bao như: RC5, Blowwfish, IDEA, NewDES, SAFER và FEAL. Hầu hết những thuật toán này có thể sử dụng từ khóa 64 bit của DES mặc dù chúng thường được thiết kế hoạt động với từ khóa 64 bit hay 128 bit. Bản thân DES cũng cải tiến để có thể được sử dụng an toàn hơn.

Vào năm 1999, Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (NIST) đã ban hành một phiên bản mới của tiêu chuẩn DES chỉ ra rằng DES chỉ nên được sử dụng cho các hệ thống cũ và 3-DES được sử dụng. 3-DES có hai ưu điểm đảm bảo cho việc sử dụng rộng rãi trong vài năm tới. Đầu tiên, với độ dài khóa 168 bit, nó khắc phục được lỗ hổng đối với cuộc tấn công vét cạn của DES. Thứ hai, thuật toán mã hóa cơ bản trong 3-DES cũng giống như trong DES. Thuật toán này đã được giám sát kỹ lưỡng hơn bất kỳ thuật toán mã hóa

nào khác trong một khoảng thời gian dài và không có cuộc tấn công phá mã hiệu quả nào dựa trên thuật toán thay vì vét cạn được tìm thấy. Do đó, 3-DES có khả năng chống phá mã rất tốt. Nếu bảo mật là yếu tố duy nhất được xem xét, thì 3-DES sẽ là lựa chọn thích hợp cho thuật toán mã hóa tiêu chuẩn trong nhiều thập kỷ tới.

Hạn chế chính của 3-DES là thuật toán tương đối chậm trong phần mềm. DES ban đầu được thiết kế để triển khai bằng phần cứng giữa những năm 1970 và không tạo ra mã phần mềm hiệu quả. 3-DES có số vòng gấp ba lần DES, do đó thực hiện chậm hơn DES. Một nhược điểm phụ là cả DES và 3-DES đều sử dụng kích thước khối 64 bit. Vì lý do hiệu quả và bảo mật, kích thước khối lớn hơn là cần thiết.

Vì những nhược điểm này, 3-DES không phải là ứng cử viên thích hợp để sử dụng lâu dài. Để thay thế, vào năm 1997 NIST đã đưa ra lời kêu gọi đề xuất Tiêu chuẩn mã hóa nâng cao (AES) mới, tiêu chuẩn này phải có sức mạnh bảo mật bằng hoặc tốt hơn 3-DES. Ngoài các yêu cầu chung này, NIST quy định rằng AES phải là mật mã khối đối xứng với độ dài khối 128 bit và hỗ trợ độ dài khóa có thể là 128, 192 và 256 bit.

Trong vòng đánh giá đầu tiên, 15 thuật toán được đề xuất đã được chấp nhận. Vòng thứ hai thu hẹp còn 5 thuật toán mạnh mẽ và tiềm năng. Sau thời gian dài phân tích, đánh giá toàn diện và tham khảo ý kiến của nhiều chuyên gia trên thế giới, NIST đã hoàn thành quá trình đánh giá của mình và chính thức xuất bản tiêu chuẩn cuối cùng vào tháng 11 năm 2001. NIST đã chọn Rijndael làm thuật toán AES được đề xuất. Hai nhà nghiên cứu đã phát triển và gửi Rijndael cho AES đều là những nhà mật mã học tài năng đến từ Bỉ: Tiến sĩ Joan Daemen và Tiến sĩ Vincent Rijmen. Khác với DES sử dụng mạng Feistel, Rijndael sử dụng mạng thay thế-chuyển vị.

Cuối cùng, AES được thiết kế để thay thế 3-DES. Tuy nhiên, quá trình thay thế này không thể diễn ra ngay lập tức mà sẽ mất một khoảng thời gian nhất định để các hệ thống và tổ chức chuyển đổi hoàn toàn. NIST đã dự đoán rằng 3-DES vẫn sẽ được sử dụng rộng rãi trong tương lai gần, đặc biệt là trong các ứng dụng không yêu cầu tốc độ xử lý cao hoặc mức độ bảo mật quá lớn.

AES có nhiều ưu điểm vượt trội khi so sánh với các thuật toán mã hóa khác. Nó có thể dễ dàng được thực hiện với tốc độ cao thông qua cả phần mềm và phần cứng, phù hợp cho các thiết bị hiện đại từ máy tính, điện thoại di động

đến các hệ thống nhúng. Một điểm mạnh khác của AES là không yêu cầu nhiều bộ nhớ, giúp nó trở thành lựa chọn lý tưởng cho các thiết bị có tài nguyên hạn chế. Với những ưu điểm này cùng với việc được công nhận là tiêu chuẩn mã hóa mới, AES đang được triển khai sử dụng rộng rãi trên toàn cầu trong nhiều lĩnh vực, từ bảo mật dữ liệu cá nhân, ngân hàng điện tử, đến các hệ thống bảo mật quốc gia.

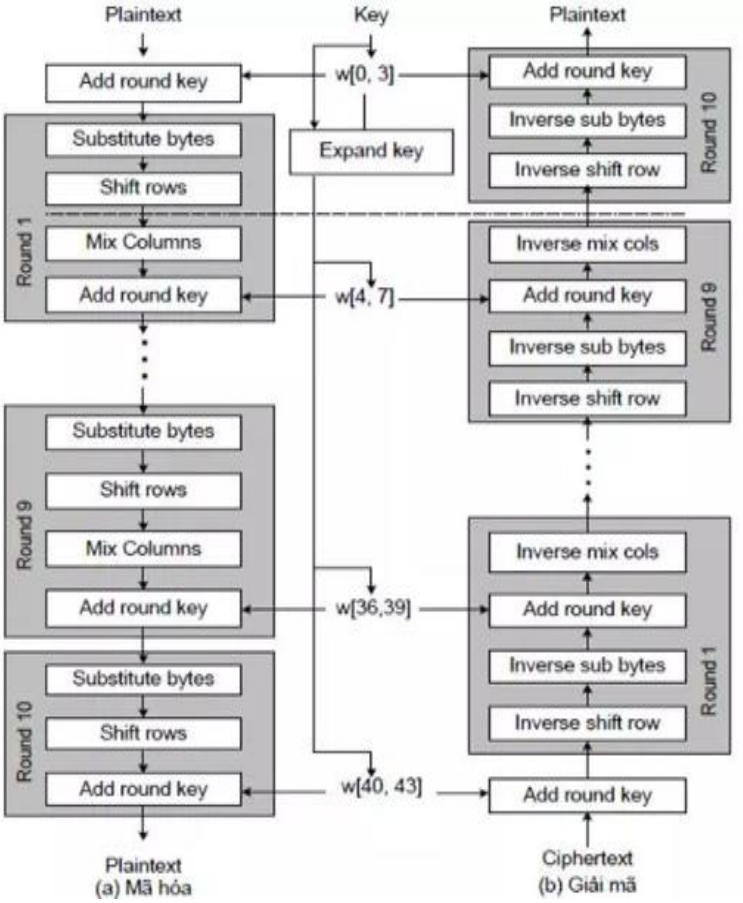
Các tham số của AES phụ thuộc vào kích thước của khóa. Chúng được thể hiện qua bảng dưới:

Bảng 2.1: Tham số của AES

Kích thước khóa	128 bit	192 bit	256 bit
Kích thước khối của bản rõ	128 bit	128 bit	128 bit
Số vòng	10	12	14
Kích thước khóa tại mỗi vòng	128 bit	128 bit	128 bit
Kích thước khóa mở rộng	1408 bit	1664 bit	1920 bit

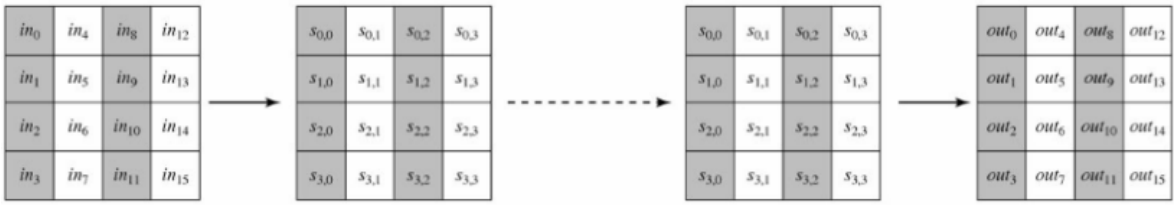
2.2.2. Mã hóa và giải mã AES

Sơ đồ tổng quát của quá trình mã hóa và giải mã AES:



Hình 2.2: Sơ đồ tổng quát mã hóa và giải mã AES

Đầu vào cho thuật toán mã hóa và giải mã là một khối 128 bit, khối bit này được mô tả là một ma trận vuông, mỗi ô là 1 byte. Khối này được sao chép vào một mảng trạng thái, được sửa đổi ở mỗi giai đoạn mã hóa hoặc giải mã. Sau giai đoạn cuối cùng, mảng trạng thái này được sao chép vào một ma trận đầu ra.



Hình 2.3: Đầu vào, mảng trạng thái và đầu ra

Tương tự, khóa 128 bit được mô tả như một ma trận vuông, mỗi phần tử là 1 byte. Khóa này sau đó được mở rộng thành một mảng các từ (word), mỗi từ là 4 byte và tổng chiều dài khóa là 44 từ cho khóa 128 bit.

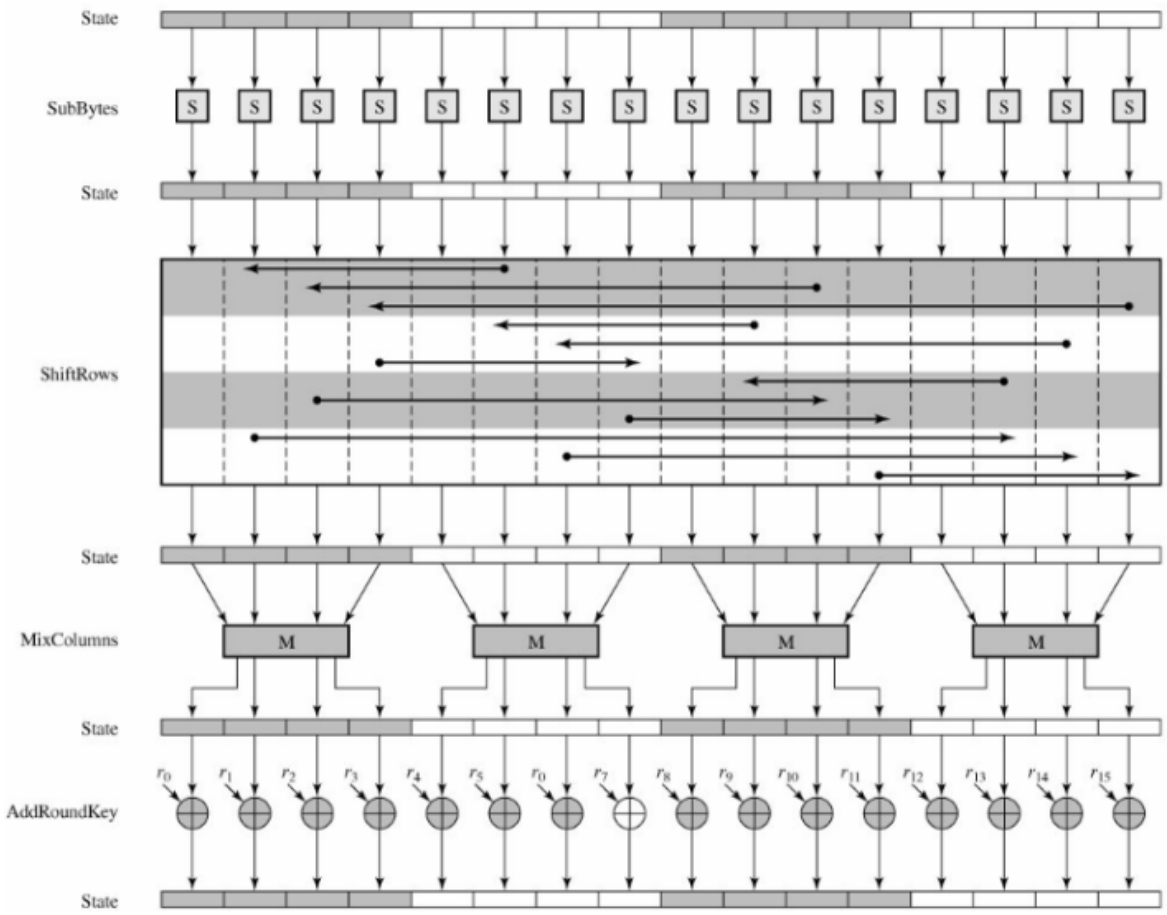
Hình 2.4: Khóa và mở rộng khóa

Thứ tự của các byte trong ma trận là theo cột. Vì vậy, 4 byte đầu tiên của bản rõ 128 bit đầu vào chiếm cột đầu tiên của ma trận, bốn byte thứ hai chiếm cột thứ hai, … Tương tự, 4 byte đầu tiên của khóa mở rộng tạo thành một từ, chiếm cột đầu tiên của ma trận.

Cấu trúc của thuật toán AES tương đối đơn giản. Cả thuật toán mã hóa và giải mã đều bắt đầu với giai đoạn Add round key, tiếp theo là 9 vòng, mỗi vòng đầy đủ 4 giai đoạn:

- Thay thế byte (Substitute bytes): Sử dụng hộp S để thực hiện việc thay thế từng byte của khối.
- Dịch dòng (Shift rows): Thực hiện hoán vị.
- Trộn cột (Mix columns): Phép thay thế sử dụng các phép toán số học trên Z_{256}.
- Cộng với khóa (Add round key): Phép XOR của khối hiện tại với một phần của khóa được mở rộng.

Vòng cuối cùng chỉ có 3 giai đoạn đó là: Substitute bytes, Shift rows và Add round key.



Hình 2.5: Minh họa một vòng mã AES

2.2.2.1. Thay thế byte (Substitute bytes)

Thay thế byte đơn giản chỉ là tra cứu trong bảng 16 x 16, mỗi ô là 1 byte và được gọi là hộp S (sử dụng trong Substitute bytes) và hộp S đảo (sử dụng trong Inverse Substitute bytes).

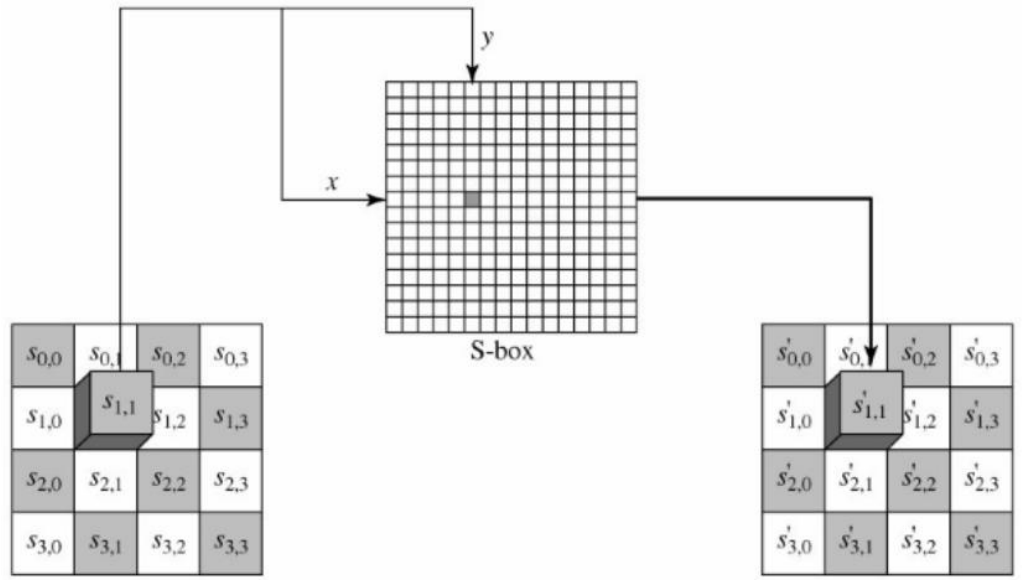
		y															
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
x	**0**	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	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

Hình 2.6: Hộp S

		y														
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	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

Hình 2.7: Hộp S đảo

Minh họa việc tra cứu hộp S:



Hình 2.8: Minh họa việc tra cứu hộp S

Ví dụ minh họa phép thay thế byte:

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

→

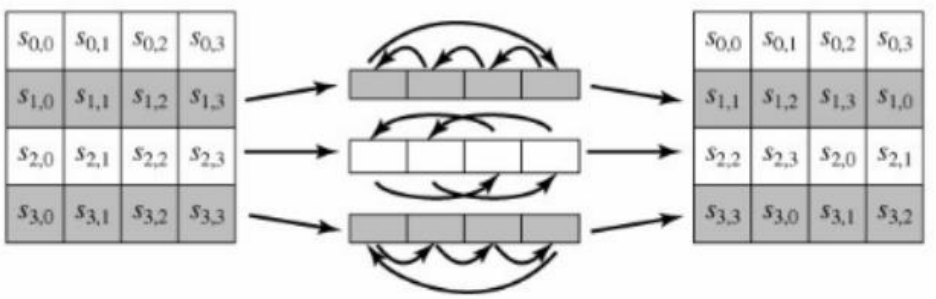
87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Hình 2.9: Minh họa phép thay thế byte

Để tìm byte thay thế của byte EA, ta tra dòng E và cột A trong hộp S thu được byte 87. Như vậy, byte EA sẽ được thay thế bằng byte 87. Tương tự, byte 04 ta tra dòng 0 và cột 4 thu được F2. Ta làm tương tự cho các byte còn lại sẽ thu được ma trận kết quả sau khi thực hiện phép thay thế.

2.2.2.2. Dịch dòng (Shift rows)

Minh họa phép dịch dòng. Dòng đầu tiên của ma trận trạng thái được giữ nguyên, dòng thứ hai dịch trái 1 byte, dòng thứ 3 dịch trái 2 byte và dòng cuối cùng dịch trái 3 byte.



Hình 2.10: Minh họa phép dịch dòng

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

→

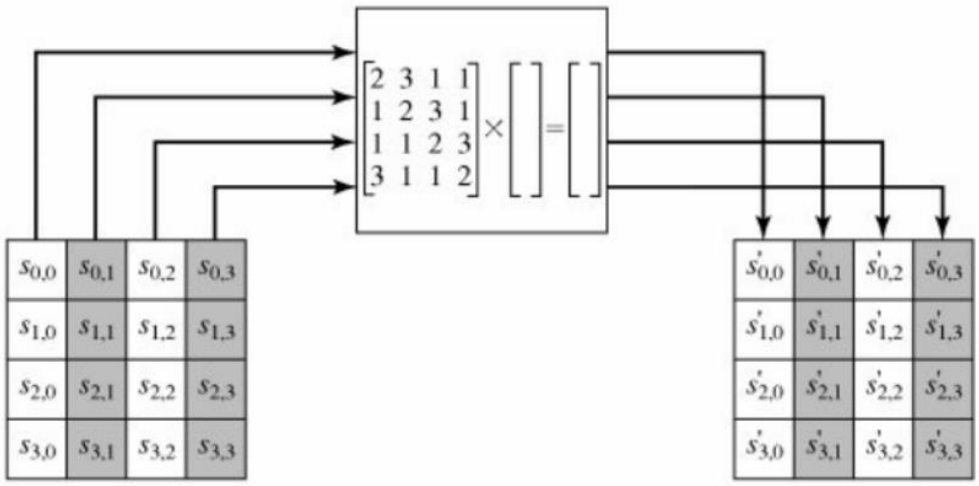
87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Hình 2.11: Ví dụ minh họa phép dịch dòng

Đối với thuật toán giải mã ta sử dụng phép dịch dòng ngược. Tức là, dòng đầu tiên của ma trận trạng thái giữ nguyên, dòng thứ 2 dịch phải 1 byte, dòng thứ 3 dịch phải 2 byte và dòng cuối cùng dịch phải 3 byte.

2.2.2.3. Trộn cột (Mix columns)

Minh họa phép trộn cột:



Hình 2.12: Minh họa phép trộn cột

Như vậy, kết quả của phép trộn cột sẽ được xác định theo công thức sau:

$$\begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \times \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

Áp dụng phép nhân hai ma trận ta thu được:

$$s'_{0,j} = (2 \cdot s_{0,j}) + (3 \cdot s_{1,j}) + s_{2,j} + s_{3,j}$$
$$s'_{1,j} = s_{0,j} + (2 \cdot s_{1,j}) + (3 \cdot s_{2,j}) + s_{3,j}$$
$$s'_{2,j} = s_{0,j} + s_{1,j} + (2 \cdot s_{2,j}) + (3 \cdot s_{3,j})$$
$$s'_{3,j} = (3 \cdot s_{0,j}) + s_{1,j} + s_{2,j} + (3 \cdot s_{3,j})$$

Trong đó, phép nhân (.) được thực hiện theo luật:

- Giả sử $s_{i,j}$ được biểu diễn dưới dạng 8 bit $b_7b_6b_5b_4b_3b_2b_1b_0$.
- Khi nhân với 2 sẽ được thực hiện theo công thức sau:

$$2.s_{i,j} = \begin{cases} b_6b_5b_4b_3b_2b_1 0 \text{ nếu } b_7 = 0 \\ b_6b_5b_4b_3b_2b_1 0 + 00011011 \text{ nếu } b_7 = 1 \end{cases}$$

- Khi nhân với 3 sẽ được thực hiện theo công thức sau:

$$3 \cdot s_{i,j} = s_{i,j} + 2 \cdot s_{i,j}$$

Phép cộng (+) trong các công thức trên là phép XOR bit.

Ví dụ minh họa phép trộn cột:

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

→

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

Hình 2.13: Ví dụ minh họa phép trộn cột

Diễn giải cách xác định phần tử đầu tiên trong ma trận sau khi thực hiện phép trộn cột:

s'_{00} = 2.(87) + 3.(6E) + 46 + A6

Chuyển các số từ hệ 16 sang hệ 2 để thực hiện các phép tính:

87h = 10000111

(b_7 = 1 nên 2.(87) = 00001110 XOR 00011011 = 00010101)

6Eh = 01101110

(b_7 của 6E là 0 nên 2.(6E) = 11011100.

Do đó, 3.(6E) = 01101110 XOR 11011100 = 10110010)

46h = 01000110

A6h = 10100110

2.(87)	=	0	0	0	1	0	1	0	1
3.(6E)	=	1	0	1	1	0	0	1	0
46	=	0	1	0	0	0	1	1	0
A6	=	1	0	1	0	0	1	1	0
XOR		0	1	0	0	0	1	1	1 = 47h

Tính toán tương tự cho các phần tử còn lại ta thu được trạng thái sau khi thực hiện phép trộn cột.

Phép trộn cột đảo (Inverse mix columns) trong thuật toán giải mã được thực hiện như sau:

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

Thay thế công thức của phép trộn cột vào thì ta thu được công thức sau:

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

$$= \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \times \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

Như vậy thì công thức sau phải được thỏa mãn:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Chứng minh phần tử đầu tiên thỏa mãn yêu cầu:

2.(0E) + 0B + 0D + 3.(09)

= 00011100 + 00001011 + 00001101 + 3.(09)

2.(0E)	=	0	0	0	1	1	1	0	0
0B	=	0	0	0	0	1	0	1	1
0D	=	0	0	0	0	1	1	0	1
3.(09)	=	0	0	0	1	1	0	1	1
XOR		0	0	0	0	0	0	0	1 = 1

Tính toán tương tự cho các phần tử còn lại.

2.2.2.4. Cộng với khóa (Add round key)

Phép cộng với khóa là thực hiện phép XOR bit của 128 bit của ma trận trạng thái và 128 bit của khóa vòng tương ứng.

Ví dụ minh họa phép cộng khóa, ma trận đầu tiên là trạng thái và ma trận thứ hai là khóa của vòng:

47	40	A3	4C	⊕	AC	19	28	57	=	EB	59	8B	1B
37	D4	70	9F		77	FA	D1	5C		40	2E	A1	C3
94	E4	3A	42		66	DC	29	00		F2	38	13	42
ED	A5	A6	BC		F3	21	41	6A		1E	84	E7	D2

Hình 2.14: Ví dụ minh họa phép cộng khóa

2.2.2.5. Mở rộng khóa (Key expansion)

Thuật toán mở rộng khóa có đầu vào là 4 từ (16 byte) khóa và tạo ra một mảng đầu ra 44 từ (176 byte). Mã giả của thuật toán được mô tả như sau:

```
KeyExpansion (byte key[16], word w[44])
{
word temp
for(i=0;i<4;i++)
  w[i] = (key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
for(i=4,i<44;i++)
{
  temp = w[i-1]
  if(i mod 4 = 0)
    temp = SubWord(RotWord(temp)) XOR Rcon[i/4]
  w[i] = w[i-4] XOR temp
}
}
```

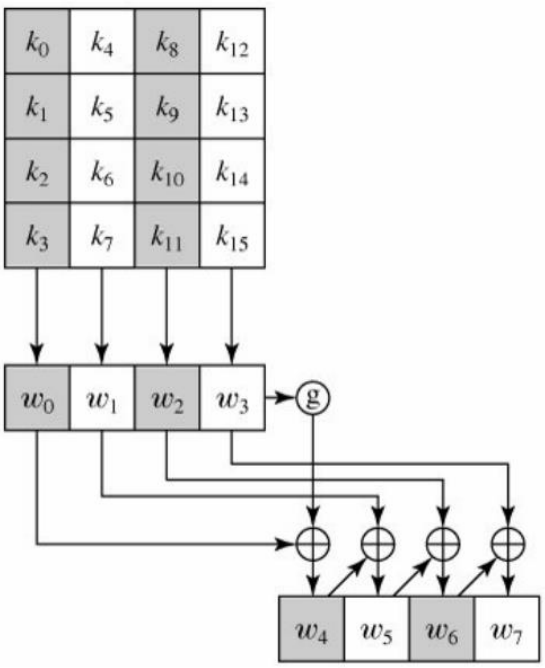
Phép toán RotWord là thực hiện phép dịch trái 1 byte, tức là đầu vào l từ có 4 byte [b_0, b_1, b_2, b_3] thì kết quả sau khi thực hiện phép dịch trái 1 byte sẽ là [b_1, b_2, b_3, b_0].

Phép toán SubWord là phép thay thế byte sử dụng bảng S.

Hằng số cho mỗi vòng khóa Rcon[j] = (RC[j], 0, 0, 0), với RC[1] = 1, RC[j] = 2.RC[j-1] và phép nhân (.) được thực hiện theo quy luật như trong thuật toán trộn cột. Giá trị của RC[j] được xác định như bảng dưới ở hệ thập lục phân (Hexadecimal).

Bảng 2.2: Giá trị của RC[j]

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



Hình 2.15: Minh họa cách xác định khóa của vòng 1

Ví dụ minh họa cách xác định khóa cho vòng thứ 9 khi khóa tại vòng 8 là EA D2 73 21 B5 8D BA D2 31 2B F5 60 7F 8D 29 2F, tương ứng:

- w[32] = [EA, D2, 73, 21]
- w[33] = [B5, 8D, BA, D2]
- w[34] = [31, 2B, F5, 60]
- w[35] = [7F, 8D, 29, 2F]

Giá trị của khóa tại vòng 9 được xác định như bảng sau:

Bảng 2.3: Xác định khóa cho vòng 9 khi khóa tại vòng 8

i	Temp	RotWord	SubWord	Rcon(9)	XOR Rcon	w[i-4]	w[i] = Temp XOR w[i-4]
36	7F8D292F	8D292F7F	5DA515D2	1B000000	46A515D2	EAD27321	AC7766F3
37	AC7766F3	AC7766F3	AC7766F3	1B000000	AC7766F3	B58DBAD2	19FABC21
38	19FABC21	19FABC21	19FABC21	1B000000	19FABC21	312BF560	28B14941
39	28B14941	28B14941	28B14941	1B000000	28B14941	7F8D292F	575C606E

Như vậy, khóa của vòng 9 sẽ là AC 77 66 F3 19 FA BC 21 28 B1 49 41 57 5C 60 6E.

2.2.3. Minh họa vòng đầu trong quá trình mã hóa AES-128

Dữ liệu đầu vào:

- Bản rõ: 32 88 31 E0 C0 C7 30 3C 98 07 AF 8D 7A 79 9D 6D
- Khóa ban đầu: 2B 7E 15 16 28 AE D2 A6 AB F7 97 75 46 7A 36 42
- Khóa vòng 1: F0 7B 39 4C D8 D5 EB EA 73 22 7C 9F 35 58 4A DD

- Từ bản rõ ta được ma trận trạng thái (State):

$$\begin{bmatrix} 32 & C0 & 98 & 7A \\ 88 & C7 & 07 & 79 \\ 31 & 30 & AF & 9D \\ E0 & 3C & 8D & 6D \end{bmatrix}$$

Vòng khởi tạo:

- AddRoundKey (Cộng với khóa vòng): Cộng State với khóa ban đầu.

$$\begin{bmatrix} 32 & C0 & 98 & 7A \\ 88 & C7 & 07 & 79 \\ 31 & 30 & AF & 9D \\ E0 & 3C & 8D & 6D \end{bmatrix} \text{XOR} \begin{bmatrix} 2B & 28 & AB & 46 \\ 7E & AE & F7 & 7A \\ 15 & D2 & 97 & 36 \\ 16 & A6 & 75 & 42 \end{bmatrix} = \begin{bmatrix} 19 & E8 & 33 & 3C \\ F6 & 69 & F0 & 03 \\ 24 & E2 & 38 & AB \\ F6 & 9A & F8 & 2F \end{bmatrix}$$

Vòng thứ nhất:

- SubBytes (Thay thế byte): Mỗi byte trong State sẽ được thay thế bởi byte khác từ bảng S-box (Ví dụ: 19 sẽ được thay thế bằng D4)

$$\begin{bmatrix} D4 & 9B & C3 & EB \\ 42 & F9 & 8C & 7B \\ 36 & 98 & 07 & 62 \\ 42 & B8 & 41 & 15 \end{bmatrix}$$

- ShiftRows (Dịch dòng): Dòng thứ nhất giữ nguyên, dòng thứ hai dịch trái 1 byte, dòng thứ ba dịch trái 2 byte, dòng thứ tư dịch trái 3 byte.

$$\begin{bmatrix} D4 & 9B & C3 & EB \\ F9 & 8C & 7B & 42 \\ 07 & 62 & 36 & 98 \\ 15 & 42 & B8 & 41 \end{bmatrix}$$

- MixColumns (Trộn cột):

$$\begin{bmatrix} B1 & 82 & 9E & D2 \\ 21 & 7C & D7 & 9D \\ 1C & 15 & 07 & 41 \\ B3 & DC & 78 & 7E \end{bmatrix}$$

- AddRoundKey (Cộng với khóa vòng): Cộng State với khóa vòng 1.

$$\begin{bmatrix} B1 & 82 & 9E & D2 \\ 21 & 7C & D7 & 9D \\ 1C & 15 & 07 & 41 \\ B3 & DC & 78 & 7E \end{bmatrix} \text{XOR} \begin{bmatrix} F0 & D8 & 73 & 35 \\ 7B & D5 & 22 & 58 \\ 39 & EB & 7C & 4A \\ 4C & EA & 9F & DD \end{bmatrix} = \begin{bmatrix} 41 & 5A & ED & E7 \\ 5A & A9 & F5 & C5 \\ 25 & FE & 7B & 0B \\ FF & 36 & E7 & A3 \end{bmatrix}$$

Như vậy qua vòng đầu tiên ta được ma trận trạng thái (State):

$$\begin{bmatrix} 41 & 5A & ED & E7 \\ 5A & A9 & F5 & C5 \\ 25 & FE & 7B & 0B \\ FF & 36 & E7 & A3 \end{bmatrix}$$

2.2.4. Độ an toàn của AES

Chuẩn mã hóa nâng cao AES đã được Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (NIST) tuyên bố là giải pháp tốt nhất để bảo vệ thông tin nhạy cảm cho Chính phủ trong thế kỷ 21. AES là người bạn đồng hành không thể thiếu của Chính phủ, cơ quan Nhà nước và các tổ chức tư nhân.

AES nếu được triển khai đúng quy trình thì sẽ đảm bảo an toàn tuyệt đối. Thế nhưng, một điều cần lưu ý đó là bất kỳ một hệ thống nào cũng có thể bị tấn công nếu hacker biết được khóa mã hóa. Do đó, các khóa cần được bảo vệ bằng nhiều cách khác nhau như dùng mật khẩu mạnh, xác thực tường lửa hay phần mềm chống độc hại.

Vào thời điểm năm 2006, dạng tấn công lên AES duy nhất thành công là tấn công kênh bên (side channel attack). Vào tháng 6 năm 2003, Chính phủ Hoa Kỳ tuyên bố AES có thể được sử dụng cho thông tin mật: "Thiết kế và độ dài khóa của thuật toán AES (128, 192 và 256 bit) là đủ an toàn để bảo vệ các thông tin được xếp vào loại tối mật. Các thông tin tuyệt mật sẽ phải dùng khóa 192 hoặc 256 bit. Các phiên bản thực hiện AES nhằm mục đích bảo vệ hệ thống an ninh hay thông tin quốc gia phải được NSA kiểm tra và chứng nhận trước khi sử dụng.".

Điều này đánh dấu lần đầu tiên công chúng có quyền tiếp xúc với thuật toán mật mã mà NSA phê chuẩn cho thông tin tuyệt mật. Nhiều phần mềm thương mại hiện nay sử dụng mặc định khóa có độ dài 128 bit.

Phương pháp thường dùng nhất để tấn công các dạng mã hóa khối là thử các kiểu tấn công lên phiên bản có số chu trình thu gọn. Đối với khóa 128 bit, 192 bit và 256 bit, AES có tương ứng 10, 12 và 14 chu trình. Tại thời điểm năm 2006, những tấn công thành công được biết đến là 7 chu trình đối với khóa 128 bit, 8 chu trình với khóa 192 bit và 9 chu trình với khóa 256 bit.

Một số nhà khoa học trong lĩnh vực mật mã lo ngại về an ninh của AES. Họ cho rằng ranh giới giữa số chu trình của thuật toán và số chu trình bị phá vỡ quá nhỏ. Nếu các kỹ thuật tấn công được cải thiện thì AES có thể bị phá vỡ. Ở đây, phá vỡ có nghĩa chỉ bất cứ phương pháp tấn công nào nhanh hơn tấn công kiểu duyệt toàn bộ (tấn công bạo lực). Vì thế một tấn công cần thực hiện 2^{120} plaintexts cũng được coi là thành công mặc dù tấn công này chưa thể thực hiện trong thực tế. Tại thời điểm hiện nay, nguy cơ này không thực sự nguy hiểm và có thể bỏ qua.

Tấn công kiểu duyệt toàn bộ quy mô nhất đã từng thực hiện là do distributed.net thực hiện lên hệ thống 64 bit RC5 vào năm 2002 (Theo định luật Moore thì nó tương đương với việc tấn công vào hệ thống 66 bit hiện nay).

Một vấn đề khác nữa là cấu trúc toán học của AES. Không giống với các thuật toán mã hóa khác, AES có mô tả toán học khá đơn giản. Tuy điều này chưa dẫn đến mối nguy hiểm nào nhưng một số nhà nghiên cứu sợ rằng sẽ có người lợi dụng được cấu trúc này trong tương lai.

Vào năm 2002, Nicolas Courtois và Josef Pieprzyk phát hiện một tấn công trên lý thuyết gọi là tấn công XSL và chỉ ra điểm yếu tiềm tàng của AES. Tuy nhiên, một vài chuyên gia về mật mã học khác cũng chỉ ra một số vấn đề chưa rõ ràng trong cơ sở toán học của tấn công này và cho rằng các tác giả đã có thể có sai lầm trong tính toán. Việc tấn công dạng này có thực sự trở thành hiện thực hay không vẫn còn để ngỏ và cho tới nay thì tấn công XSL vẫn chỉ là suy đoán.

2.2.5. Nhận xét về AES

2.2.5.1. Ưu điểm

Tính bảo mật cao: AES là một thuật toán mã hóa đối xứng mạnh, có khả năng đảm bảo tính bảo mật cao trong việc mã hóa và giải mã dữ liệu.

Tốc độ nhanh: Thuật toán này có tốc độ mã hóa và giải mã rất nhanh, phù hợp với các ứng dụng đòi hỏi tốc độ cao như truyền dữ liệu qua mạng.

Linh hoạt: AES cũng có thể sử dụng các khóa với độ dài khác nhau (128, 192 hoặc 256 bit) để tăng cường tính bảo mật. Ngoài ra, thuật toán này có thể sử dụng trên các nền tảng phần cứng và phần mềm khác nhau.

Chuẩn quốc tế: AES là một chuẩn mã hóa quốc tế, được chấp nhận rộng rãi trên toàn cầu.

2.2.5.2. Nhược điểm

AES phụ thuộc vào độ dài khóa để đảm bảo tính bảo mật. Nếu khóa ngắn hoặc dễ đoán, AES có thể bị tấn công và dữ liệu có thể bị tiết lộ. AES cần sử dụng khóa an toàn để đảm bảo tính bảo mật. Nếu khóa bị mất hoặc rơi vào tay của kẻ xấu, dữ liệu có thể bị tiết lộ.

AES không thể đảm bảo tính bảo mật nếu hệ thống bị tấn công bởi các phương thức khác như tấn công mạng, tấn công từ chối dịch vụ (DoS), tấn công đánh cắp thông tin đăng nhập, …

Ngoài ra việc triển khai AES có thể đòi hỏi chi phí phần cứng và phần mềm khá lớn để bảo vệ dữ liệu, đặc biệt là trong các hệ thống lớn và phức tạp.

2.2.6. Ứng dụng của AES

Bảo mật dữ liệu:

- Lưu trữ dữ liệu an toàn: AES được sử dụng để mã hóa dữ liệu nhạy cảm, như tài liệu, tệp tin, nhằm ngăn chặn truy cập trái phép.
- Bảo mật thiết bị di động: Dữ liệu trên điện thoại thông minh và máy tính bảng thường được mã hóa bằng AES, ví dụ: iOS của Apple và Android đều sử dụng AES để mã hóa bộ nhớ.

Bảo mật giao tiếp:

- Mạng Internet: AES là thành phần quan trọng trong các giao thức bảo mật như HTTPS, TLS/SSL để mã hóa thông tin truyền tải giữa trình duyệt và máy chủ.
- VPN (Mạng riêng ảo): AES được dùng để mã hóa dữ liệu truyền qua VPN, đảm bảo an toàn khi truy cập từ xa.
- Email và tin nhắn: Các ứng dụng mã hóa như ProtonMail hoặc Signal sử dụng AES để bảo mật thông tin liên lạc.

Ứng dụng trong ngân hàng và tài chính:

- Bảo vệ giao dịch trực tuyến: AES mã hóa thông tin thẻ tín dụng, thông tin tài khoản ngân hàng khi thực hiện giao dịch.
- ATM và POS: Các thiết bị này sử dụng AES để đảm bảo tính an toàn trong việc mã hóa dữ liệu.

Bảo mật lưu trữ đám mây:

- Các nhà cung cấp dịch vụ đám mây, như Google Drive, Dropbox, hoặc iCloud, sử dụng AES để mã hóa dữ liệu người dùng trên máy chủ, giúp bảo vệ dữ liệu khỏi các mối đe dọa.

Bảo mật thiết bị IoT:

- Trong các thiết bị IoT (Internet of Things), AES được sử dụng để mã hóa dữ liệu truyền tải giữa các thiết bị, đảm bảo dữ liệu không bị đánh cắp hoặc giả mạo.

Bảo mật thông tin quân sự và chính phủ:

- AES được tiêu chuẩn hóa bởi Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (NIST) và được nhiều tổ chức quân sự, chính phủ sử dụng để bảo mật thông tin nhạy cảm.

Bảo mật ứng dụng phần mềm:

- Quản lý mật khẩu: Các trình quản lý mật khẩu, như LastPass hoặc KeePass, sử dụng AES để mã hóa các tệp chứa mật khẩu.
- Ứng dụng bảo mật: Các ứng dụng bảo mật như BitLocker (Windows), FileVault (macOS) sử dụng AES để mã hóa toàn bộ ổ đĩa.

Ngành y tế:

- Trong y tế, AES được dùng để mã hóa hồ sơ sức khỏe điện tử (EHR) và bảo vệ dữ liệu bệnh nhân theo các tiêu chuẩn như HIPAA.

Phát triển phần mềm và trò chơi:

- AES giúp bảo vệ mã nguồn, dữ liệu người dùng, hoặc thông tin tài chính khi tích hợp thanh toán trong ứng dụng và trò chơi.

Bảo vệ quyền sở hữu trí tuệ:

- Dữ liệu số như phim, nhạc, sách điện tử được mã hóa bằng AES để ngăn chặn sao chép và phân phối trái phép.

2.3. Thiết kế chương trình và cài đặt thuật toán

2.3.1. Thiết kế kịch bản chương trình

2.3.1.1. Mục tiêu

Việc phát triển và xây dựng chương trình mã hóa và giải mã AES nhằm đáp ứng các mục tiêu sau:

- Xây dựng chương trình mã hóa và giải mã AES đầy đủ (AES-128, AES-192 và AES-256).
- Cho phép người dùng sử dụng khóa cho việc mã hóa và giải mã với kích thước tùy chọn, đáp ứng với từng yêu cầu bảo mật cụ thể.
- Cho phép người dùng có thể tùy chỉnh định dạng đầu vào hoặc đầu ra của tệp tin cần mã hóa hoặc giải mã, phù hợp với điều kiện thực tế.
- Cho phép người dùng mã hóa và giải mã văn bản trực tiếp trên giao diện chương trình.
- Cho phép người dùng mở các tệp tin txt để mã hóa hoặc giải mã, kết quả của việc mã hóa và giải mã có thể lưu lại nếu muốn.
- Cho phép người dùng mã hóa và giải mã các loại tệp tin khác như docx, pptx, xlsx, pdf, png, jpg, gif, mp3, mp4, … Sau quá trình mã hóa hoặc giải mã, tệp tin sẽ được lưu lại vào thiết bị của người dùng.

2.3.1.2. Kịch bản mã hóa văn bản

Sau khi truy cập vào chương trình, người dùng có thể thực hiện mã hóa văn bản qua các thao tác sau:

Bước 1: Nhập khóa

- Chọn kích thước khóa mong muốn (128, 192 hoặc 256 bit).
- Nhập khóa bí mật của mình tương ứng với kích thước khóa đã chọn.

Bước 2: Mã hóa

- Chọn định dạng đầu ra mong muốn của bản mã (Hex hoặc Base64).
- Nhập dữ liệu vào trường bản rõ (Tại bước này có thể mở tệp tin txt nếu muốn bằng cách nhấn nút "Mở file txt" và chọn tệp txt cần mã hóa).
- Nhấn nút "Mã hóa" (Tại bước này nếu khóa đã nhập không hợp lệ, chương trình sẽ thông báo lỗi "Khóa bị thay đổi! Vui lòng kiểm tra lại kịch thước và nội dung khóa!").

Bước 3: Nhận kết quả mã hóa

- Sau khi việc mã hóa hoàn tất, bản mã sẽ được hiển thị tại trường bản mã.
- Tại đây có thể lưu bản mã dưới dạng tệp txt nếu muốn bằng cách nhấn nút "Lưu file txt" sau đó chọn vị trí muốn lưu tệp.

2.3.1.3. Kịch bản giải mã văn bản

Sau khi truy cập vào chương trình, người dùng có thể thực hiện giải mã văn bản qua các thao tác sau:

Bước 1: Nhập khóa

- Chọn kích thước khóa mong muốn (128, 192 hoặc 256 bit).
- Nhập khóa bí mật của mình tương ứng với kích thước khóa đã chọn.

Bước 2: Giải mã

- Chọn định dạng đầu vào của bản mã (Hex hoặc Base64).
- Nhập dữ liệu vào trường bản mã (Tại bước này có thể mở tệp tin txt nếu muốn bằng cách nhấn nút "Mở file txt" và chọn tệp txt cần giải mã).
- Nhấn nút "Giải mã" (Tại bước này nếu khóa đã nhập không hợp lệ, chương trình sẽ thông báo lỗi "Khóa bị thay đổi! Vui lòng kiểm tra lại kịch thước và nội dung khóa!").

- Tại đây nếu văn bản mã đã bị thay đổi, chương trình sẽ hiển thị thông báo “Dữ liệu không toàn vẹn!”.

Bước 3: Nhận kết quả giải mã

- Sau khi việc giải mã hoàn tất, bản rõ tương ứng sẽ được hiển thị tại trường bản rõ.
- Tại đây có thể lưu bản rõ dưới dạng tệp txt nếu muốn bằng cách nhấn nút “Lưu file txt” sau đó chọn vị trí muốn lưu tệp.

2.3.1.4. Kịch bản mã hóa các loại tệp tin

Sau khi truy cập vào chương trình, người dùng có thể thực hiện mã hóa các loại tệp tin như docx, pptx, xlsx, pdf, png, jpg, gif, mp3, mp4, … qua các thao tác sau:

Bước 1: Nhập khóa

- Chọn kích thước khóa mong muốn (128, 192 hoặc 256 bit).
- Nhập khóa bí mật của mình tương ứng với kích thước khóa đã chọn.

Bước 2: Mở file cần mã hóa

- Nhập đường dẫn của tệp tin cần mã hóa hoặc có thể mở tệp tin bằng cách nhấn nút “Mở file”.

Bước 3: Mã hóa

- Nhấn nút “Mã hóa file” (Tại bước này nếu khóa đã nhập không hợp lệ, chương trình sẽ thông báo lỗi “Khóa bị thay đổi! Vui lòng kiểm tra lại kịch thước và nội dung khóa!”).

Bước 4: Lưu tệp tin sau khi mã hóa

- Sau khi quá trình mã hóa hoàn tất, chương trình sẽ hiển thị cửa sổ cho phép lưu tệp tin đã mã hóa.
- Đặt tên tệp tin và chọn đuôi tệp tương ứng với bản rõ.
- Chọn vị trí và lưu lại tệp tin.

2.3.1.5. Kịch bản giải mã các loại tệp tin

Sau khi truy cập vào chương trình, người dùng có thể thực hiện giải mã các loại tệp tin như docx, pptx, xlsx, pdf, png, jpg, gif, mp3, mp4, … qua các thao tác sau:

Bước 1: Nhập khóa

- Chọn kích thước khóa mong muốn (128, 192 hoặc 256 bit).
- Nhập khóa bí mật của mình tương ứng với kích thước khóa đã chọn.

Bước 2: Mở file cần giải mã

- Nhập đường dẫn của tệp tin cần giải mã hoặc có thể mở tệp tin bằng cách nhấn nút “Mở file”.

Bước 3: Giải mã

- Nhấn nút “Giải mã file” (Tại bước này nếu khóa đã nhập không hợp lệ, chương trình sẽ thông báo lỗi “Khóa bị thay đổi! Vui lòng kiểm tra lại kịch thước và nội dung khóa!”).
- Tại đây nếu đường dẫn tệp tin không hợp lên hoặc dữ liệu tệp tin đã bị thay đổi, chương trình sẽ hiển thị thông báo “Đường dẫn không hợp lệ hoặc file không tồn tại! Vui lòng kiểm tra lại!”.

Bước 4: Lưu tệp tin sau khi giải mã

- Sau khi quá trình giải mã hoàn tất, chương trình sẽ hiển thị cửa sổ cho phép lưu tệp tin đã giải mã.
- Đặt tên tệp tin và chọn đuôi tệp tương ứng với bản mã.
- Chọn vị trí và lưu lại tệp tin.

2.3.2. Cài đặt thuật toán và giao diện chương trình với Java

2.3.2.1. Công cụ và công nghệ triển khai

Chương trình được triển bằng công cụ NetBeans - một môi trường phát triển tích hợp (IDE) (tại thời điểm xây dựng chương trình sử dụng phiên bản Apache NetBeans IDE 22). NetBeans giúp cho việc phát triển chương trình trở lên dễ dàng hơn nhờ một số tính năng sau:

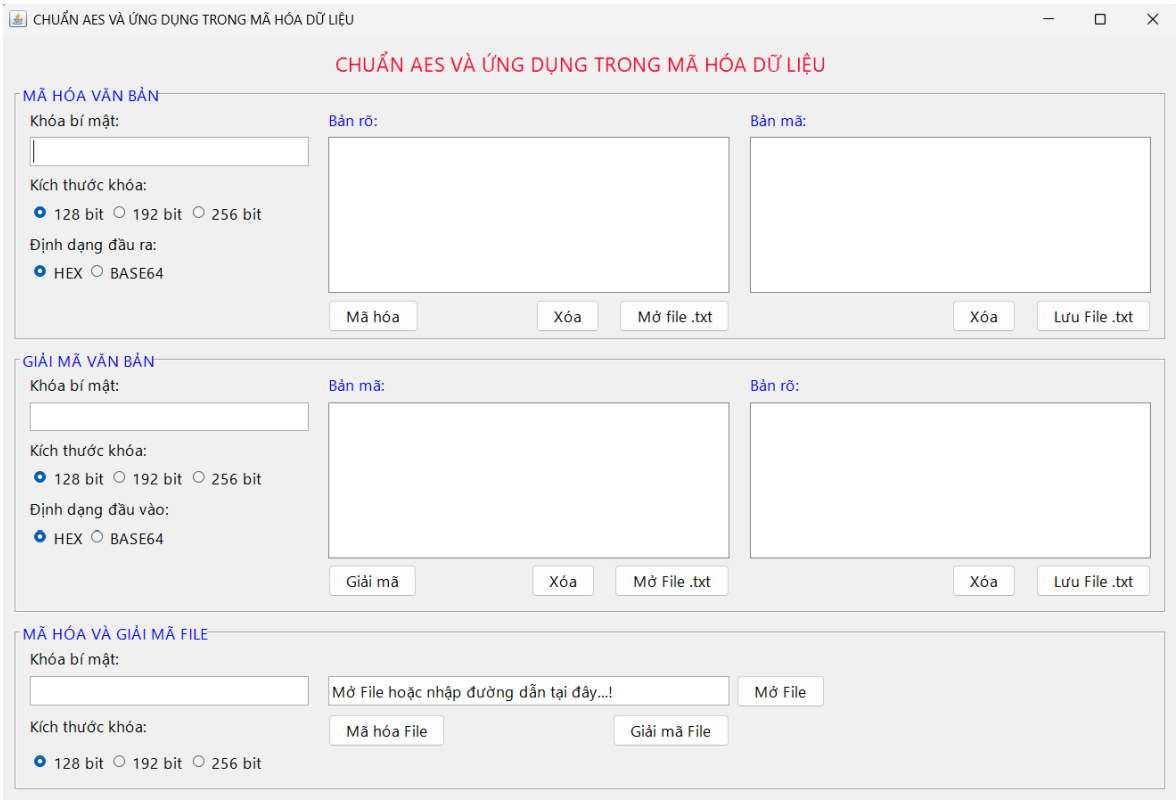
- Hỗ trợ biên dịch, gỡ lỗi và chạy chương trình trực tiếp trong môi trường.
- Hỗ trợ giao diện kéo thả, đặc biệt hữu ích để thiết kế giao diện người dùng (GUI) với công nghệ như Java Swing.
- Hỗ trợ quản lý mã nguồn và cấu trúc dự án.
- Có thể chạy trên nhiều nền tảng.

Ngôn ngữ triển khai khai thuật toán: Các thuật toán phục vụ cho việc mã hóa và giải mã AES được triển khai với ngôn ngữ Java (tại thời điểm xây dựng chương trình sử dụng phiên bản Java 22.0.1).

Công nghệ xây dựng giao diện: Ngoài các thuật toán, giao diện cũng được triển khai nhằm giúp người dùng dễ dàng thao tác và sử dụng. Giao diện của chương trình được xây dựng với công nghệ Java Swing - một thư viện hỗ trợ xây dựng giao diện người dùng trong Java.

2.3.2.2. Kết quả triển khai

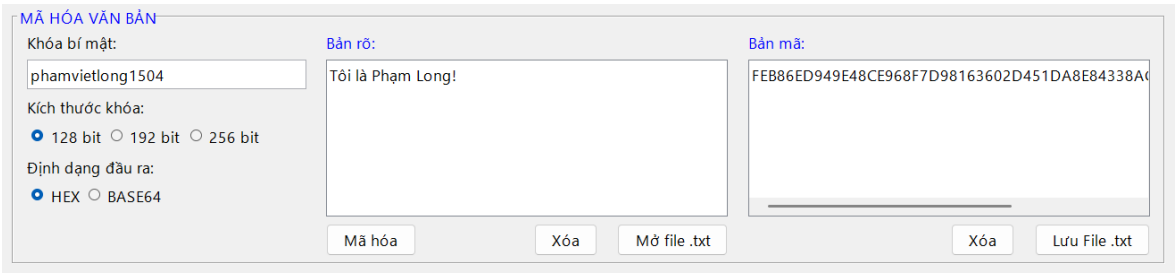
Qua quá trình triển khai, xây dựng và kiểm thử, chương trình Mã hóa và giải mã AES bằng ngôn ngữ Java đã được hoàn thành với các chức năng chính xác và dễ sử dụng. Giao diện chính của chương trình như hình dưới.



Hình 2.16: Giao diện chương trình mã hóa và giải mã AES bằng Java

Khu vực mã hóa văn bản: Tại đây người dùng có thể thực hiện việc mã hóa các đoạn văn bản hoặc các tệp txt khi cần thiết. Các thao tác tại khu vực này bao gồm:

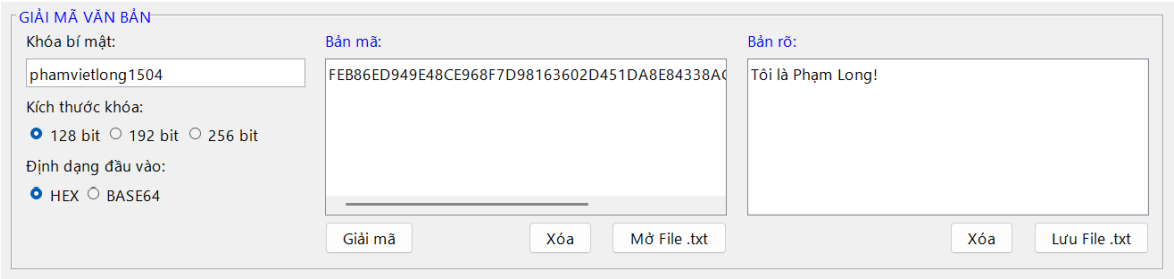
- Lựa chọn kích thước khóa và nhập khóa bí mật của mình với kích thước tương ứng đã chọn.
- Lựa chọn định dạng đầu ra mong muốn cho văn bản mã hóa.
- Nhập bản rõ từ bàn phím hoặc mở tệp txt cần mã hóa.
- Mã hóa để tạo ra bản mã tương ứng (có thể lưu lại bản mã sau khi mã hóa vào tệp txt nếu muốn).



Hình 2.17: Khu vực mã hóa văn bản AES với Java

Khu vực giải mã văn bản: Tại đây người dùng có thể thực hiện việc giải mã các đoạn văn bản hoặc các tệp txt khi cần thiết. Các thao tác tại khu vực này bao gồm:

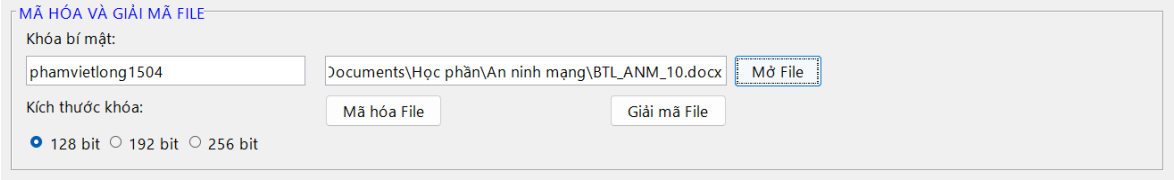
- Lựa chọn kích thước khóa và nhập khóa bí mật của mình với kích thước tương ứng đã chọn.
- Lựa chọn định dạng đầu vào của bản mã.
- Nhập bản mã từ bàn phím hoặc mở tệp txt cần giải mã.
- Giải mã để được bản rõ tương ứng (có thể lưu lại bản rõ sau khi giải mã vào tệp txt nếu muốn).



Hình 2.18: Khu vực giải mã văn bản AES với Java

Khu vực mã hóa và giải mã các loại tệp tin: Tại đây người dùng có thể thực hiện việc mã hóa và giải mã nhiều loại tệp tin khác nhau như docx, pptx, xlsx, pdf, png, jpg, gif, mp3, mp4, … Các thao tác tại khu vực này bao gồm:

- Lựa chọn kích thước khóa và nhập khóa bí mật của mình với kích thước tương ứng đã chọn.
- Mã hóa tệp tin:
  + Nhập đường dẫn hoặc mở tệp tin cần mã hóa.
  + Mã hóa và lưu lại lại tệp tin sau khi mã hóa.
- Giải mã tệp tin:
  + Nhập đường dẫn hoặc mở tệp tin cần giải mã.
  + Giải mã và lưu lại tệp tin sau khi giải mã.

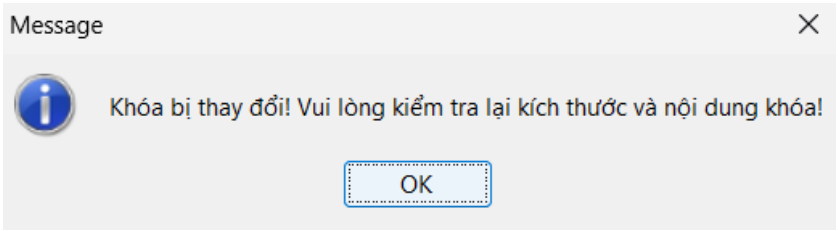


Hình 2.19: Khu vực mã hóa và giải mã các loại tệp tin AES với Java

Khi thực hiện các thao tác như nhập khóa, mã hóa và giải mã, việc nhập sai các thông tin có thể xảy ra. Biết được các trường hợp này và để tránh các lỗi không mong muốn trong qua trình mã hóa và giải mã, chương trình cũng

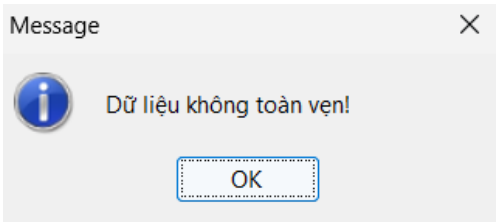
được thiết kế để bắt lỗi và xử lý lỗi. Khi có lỗi về mặt nhập liệu xảy ra, chương trình sẽ có các thông báo giúp người dùng nhận biết, từ đó giúp quá trình mã hóa và giải mã trở lên chính xác và hiệu quả hơn. Cụ thể các lỗi nhập liệu có thể xảy ra khi người dùng thao tác và cách chương trình thông báo lỗi được thể hiện như sau:

- Nhập sai kích thước hoặc nội dung khóa:



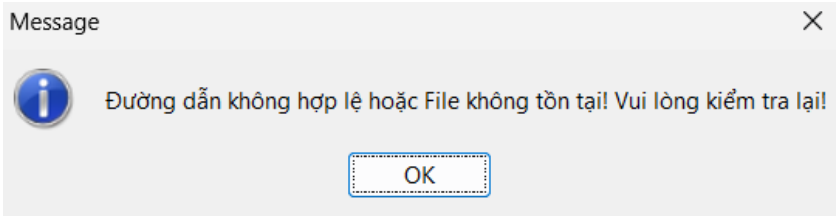
Hình 2.20: Thông báo khi người dùng nhập sai khóa

- Dữ bản mã đầu vào bị thay đổi:



Hình 2.21: Thông báo khi bản mã đầu vào bị thay đổi

- Đường dẫn tới tệp tin cần mã hóa không hợp lệ hoặc tệp tin không tồn tại:



Hình 2.22: Thông báo khi đường dẫn không hợp lệ hoặc tệp tin không tồn tại

2.3.3. Cài đặt thuật toán và giao diện chương trình với C#

2.3.3.1. Công cụ và công nghệ triển khai

Chương trình được triển khai bằng công cụ Visual Studio (phiên bản Visual Studio 2022). Visual Studio giúp cho việc phát triển chương trình trở nên dễ dàng hơn nhờ một số tính năng sau:

- Hỗ trợ biên dịch, gỡ lỗi và chạy chương trình trực tiếp trong môi trường.
- Cung cấp công cụ thiết kế giao diện kéo thả, hỗ trợ xây dựng giao diện người dùng (GUI) với công nghệ như WPF.
- Hỗ trợ quản lý mã nguồn và cấu trúc dự án một cách hiệu quả.

- Tích hợp với .NET Framework, cung cấp các thư viện và công cụ hữu ích để xử lý logic chương trình.

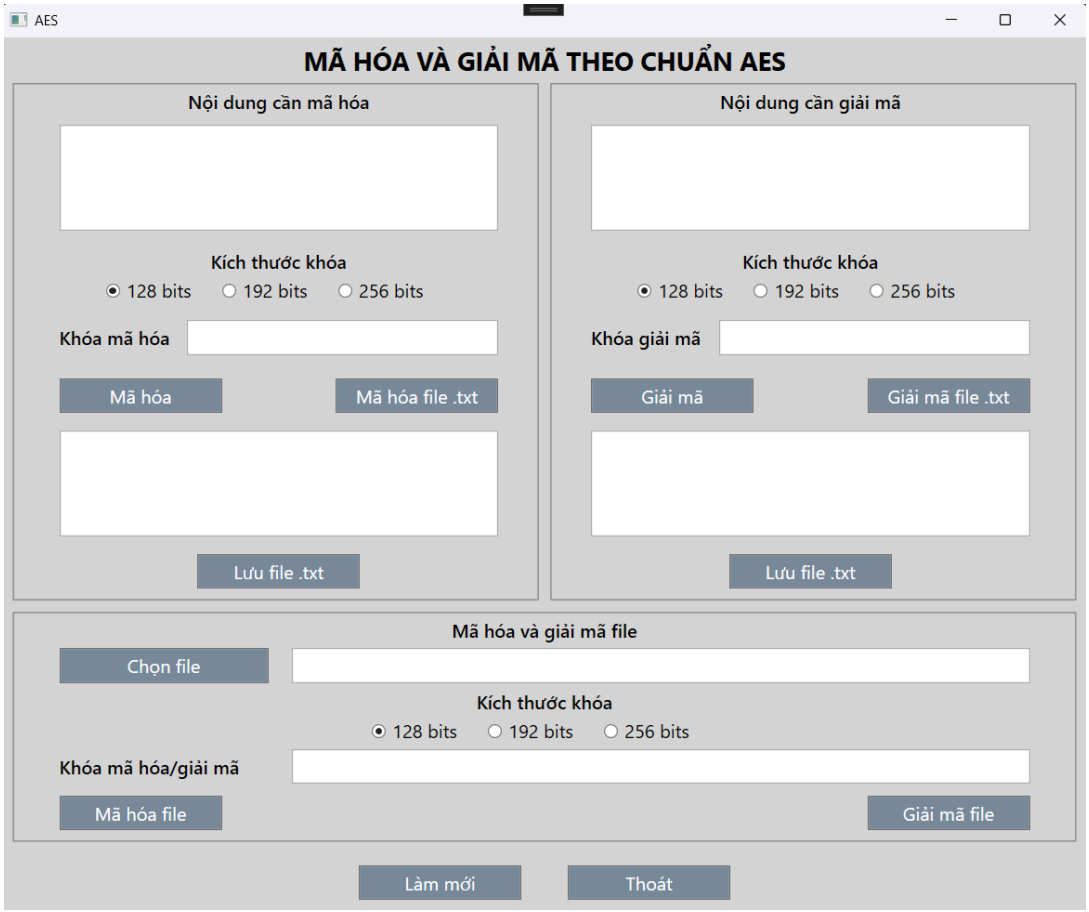
Ngôn ngữ triển khai thuật toán: Các thuật toán phục vụ cho việc mã hóa và giải mã AES được triển khai với ngôn ngữ C# (phiên bản .NET 8.0).

Công nghệ xây dựng giao diện: Giao diện của chương trình được xây dựng với công nghệ Windows Presentation Foundation (WPF) - một nền tảng giao diện mạnh mẽ thuộc .NET Framework. XAML được sử dụng để định nghĩa giao diện, trong khi C# được dùng để xử lý các sự kiện và logic ở phần code-behind.

Thư viện hỗ trợ mã hóa AES: Chương trình sử dụng thư viện *System.Security.Cryptography* của .NET Framework để triển khai mã hóa và giải mã dữ liệu theo chuẩn AES. Thư viện này cung cấp các phương thức hỗ trợ mã hóa mạnh mẽ, đảm bảo tính bảo mật và hiệu quả trong xử lý dữ liệu.

2.3.3.2. Kết quả triển khai

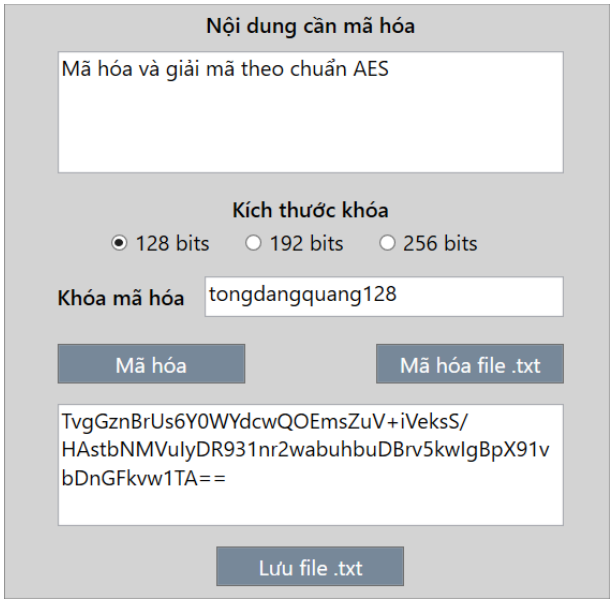
Qua quá trình triển khai, xây dựng và kiểm thử, chương trình mã hóa và giải mã AES bằng ngôn ngữ C# đã được hoàn thành với các chức năng chính xác và dễ sử dụng. Giao diện chính của chương trình như hình dưới.



Hình 2.23: Giao diện chương trình mã hóa và giải mã AES bằng C#

Khu vực mã hóa văn bản: Tại đây người dùng có thể thực hiện việc mã hóa các đoạn văn bản hoặc các tệp Text (.txt) khi cần thiết. Các thao tác tại khu vực này bao gồm:

- Nhập nội dung cần mã hóa.
- Lựa chọn kích thước khóa và nhập khóa bí mật của mình với kích thước tương ứng đã chọn.
- Có thể đọc nội dung văn bản từ tệp .txt lên ô nội dung để tiến hành mã hóa.
- Hiển thị bản mã khi đã mã hóa thành công.
- Có thể lưu lại bản mã sau khi mã hóa thành tệp văn bản có định dạng .txt nếu muốn.



Hình 2.24: Giao diện mã hóa văn bản bằng C#

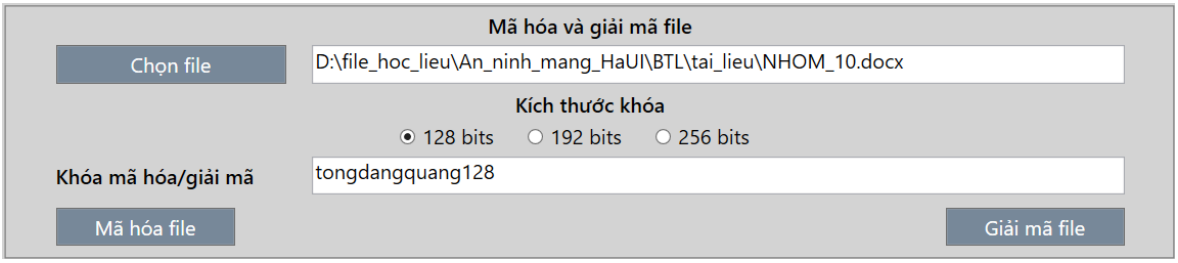
Khu vực giải mã văn bản: Tại đây người dùng có thể thực hiện việc giải mã các đoạn văn bản hoặc các tệp .txt khi cần thiết. Các thao tác tại khu vực này bao gồm:

- Nhập nội dung cần giải mã.
- Lựa chọn kích thước khóa và nhập khóa bí mật của mình với kích thước tương ứng đã chọn.
- Có thể đọc nội dung văn bản từ tệp .txt lên ô nội dung để tiến hành giải mã.
- Hiển thị bản mã khi đã mã hóa thành công.
- Có thể lưu lại bản rõ sau khi giải mã thành tệp văn bản có định dạng .txt nếu muốn.

Hình 2.25: Giao diện giải mã văn bản bằng C#

Khu vực mã hóa và giải mã các loại tệp tin: Tại đây người dùng có thể thực hiện việc mã hóa và giải mã nhiều loại tệp tin khác nhau như docx, pptx, xlsx, pdf, png, jpg, gif, mp3, mp4, … Các thao tác tại khu vực này bao gồm:

- Mở file cần mã hóa hoặc nhập trực tiếp đường dẫn đến file cần mã hóa/giải mã vào ô đường dẫn.
- Lựa chọn kích thước khóa và nhập khóa bí mật của mình với kích thước tương ứng đã chọn.
- Mã hóa tệp tin: Mã hóa và lưu lại lại tệp tin sau khi mã hóa.
- Giải mã tệp tin: Giải mã và lưu lại tệp tin sau khi giải mã.

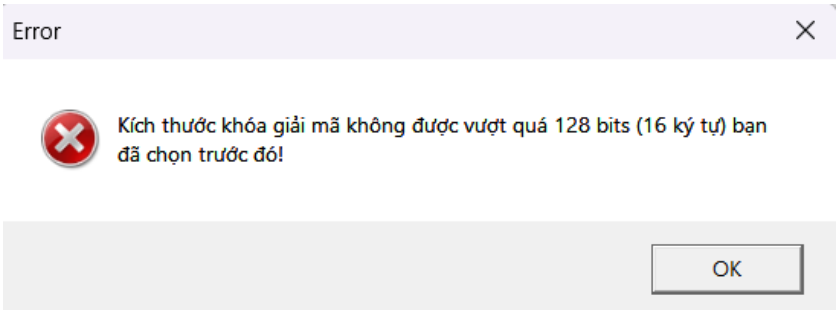


Hình 2.26: Giao diện mã hóa/giải mã tệp tin bằng C#

Khi thực hiện các thao tác như nhập khóa, mã hóa và giải mã, việc nhập sai các thông tin có thể xảy ra. Biết được các trường hợp này và để tránh các lỗi không mong muốn trong qua trình mã hóa và giải mã, chương trình cũng được thiết kế để bắt lỗi và xử lý lỗi. Khi có lỗi về mặt nhập liệu xảy ra, chương trình sẽ có các thông báo giúp người dùng nhận biết, từ đó giúp quá trình mã hóa và giải mã trở lên chính xác và hiệu quả hơn. Cụ thể các lỗi nhập liệu có

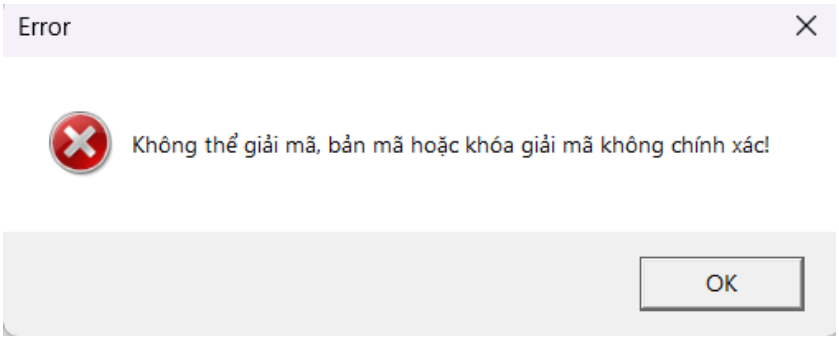
thể xảy ra khi người dùng thao tác và cách chương trình thông báo lỗi được thể hiện như sau:

- Trường hợp độ dài khóa nhập vào vượt quá kích thước khóa đã chọn:



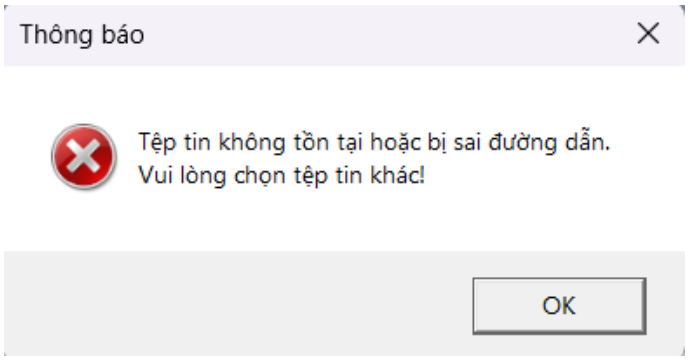
Hình 2.27: Thông báo khi độ dài khóa vượt quá kích thước khóa đã chọn

- Trường hợp người dùng nhập sai bản mã hoặc sai khóa khi giải mã:



Hình 2.28: Thông báo khi nhập sai bản mã hoặc sai khóa khi giải mã

- Trường hợp tệp tin cần mã hóa/giải mã không tồn tại hoặc đường dẫn đến tệp tin không đúng:



Hình 2.29: Thông báo khi không tìm thấy tệp tin cần mã hóa/giải mã

Tiểu kết chương 2

Chương 2 đã trình bày các kết quả nghiên cứu quan trọng liên quan đến hệ mã hóa khóa bí mật và chuẩn mã nâng cao AES, cùng với quá trình thiết kế và cài đặt chương trình minh họa.

Đầu tiên, chương đã phân tích chi tiết về hệ mã hóa khóa bí mật, bao gồm các khái niệm cơ bản, cơ chế hoạt động, và các yêu cầu cần thiết để đảm bảo tính an toàn và hiệu quả. Những nhận xét được rút ra nhấn mạnh vai trò quan trọng của loại hệ mã này trong các hệ thống bảo mật hiện đại.

Tiếp theo, chương đã tập trung nghiên cứu về chuẩn mã nâng cao AES. Quy trình mã hóa và giải mã AES đã được trình bày chi tiết, độ an toàn của AES cũng được đưa ra. Từ đó có được các nhận xét chính xác về AES cùng với các ứng dụng thực tế được rút ra.

Cuối cùng, chương đã trình bày quá trình thiết kế chương trình và cài đặt thuật toán AES. Bắt đầu từ việc thiết kế kịch bản giúp mô tả chi tiết các yêu cầu và quy trình hoạt động của chương trình. Sau đó việc cài đặt thuật toán và giao diện bằng ngôn ngữ Java và C# cũng được triển khai thành công.

Những kết quả đạt được trong chương 2 không chỉ cung cấp cái nhìn toàn diện về chuẩn AES mà còn tạo nền tảng thực tiễn vững chắc cho việc ứng dụng và triển khai trong các hệ thống bảo mật dữ liệu.

CHƯƠNG 3. KẾT LUẬN VÀ BÀI HỌC KINH NGHIỆM

3.1. Kiến thức kỹ năng học được trong quá trình thực hiện đề tài

Qua quá trình nghiên cứu và thực hiện đề tài, nhóm đã tích lũy được nhiều kiến thức chuyên môn cùng với những kỹ năng quan trọng. Đây chính là những kiến thức cơ sở, là tiền đề cho việc phát triển trong tương lai.

Về kiến thức chuyên môn, nhóm đã có cái nhìn từ tổng quan đến chi tiết về hệ mã hóa khóa bí mật, đặc biệt là về chuẩn mã hóa nâng cao AES, bao gồm khái niệm, cách thức hoạt động, cấu trúc thuật toán và các ứng dụng thực tế trong bảo mật dữ liệu. Đồng thời, nhóm cũng nắm vững các khái niệm liên quan đến mã hóa và giải mã dữ liệu trong hệ thống thông tin, tạo nền tảng cho việc phát triển các giải pháp bảo mật.

Về kỹ năng lập trình, quá trình thực hiện đề tài đã giúp nhóm cải thiện và nâng cao khả năng lập trình cùng đó là cách ứng dụng các thư viện hỗ trợ cho một bài toán cụ thể với các ngôn ngữ như Java và C#, đặc biệt là trong việc triển khai thuật toán AES. Nhóm cũng có cơ hội rèn luyện kỹ năng debug và xử lý lỗi trong các tình huống thực tế, từ đó nâng cao tư duy giải quyết vấn đề.

Ngoài ra, cách nghiên cứu thông qua việc thu thập và phân tích thông tin từ nhiều nguồn tài liệu khác nhau cũng giúp nhóm tăng cao khả năng nghiên cứu. Khả năng tổng hợp lý thuyết và thực hành đã giúp nhóm đưa ra các giải pháp phù hợp với yêu cầu thực tế của đề tài.

Cuối cùng, kỹ năng làm việc nhóm của mỗi thành viên cũng được cải thiện một cách đáng kể, bao gồm phân công công việc hiệu quả giữa các thành viên, giao tiếp và phối hợp chặt chẽ để đảm bảo tiến độ và chất lượng của đề tài. Những kỹ năng và kiến thức này không chỉ giúp nhóm hoàn thành đề tài mà còn là hành trang quan trọng trong các dự án sau này.

3.2. Bài học kinh nghiệm

Quá trình thực hiện đề tài đã giúp mỗi thành viên trong nhóm đều rút ra được những bài học kinh nghiệm quan trọng cho riêng mình. Đây là những bài học quý giá cho mỗi cá nhân, từ đó cải giúp cho quá trình cải thiện trở lên hiệu quả hơn.

Việc xây dựng một kế hoạch rõ ràng ngay từ đầu đóng vai trò then chốt trong việc hoàn thành đề tài đúng hạn. Nhóm nhận thấy rằng chia nhỏ công việc thành các giai đoạn cụ thể, đặt ra các mốc thời gian (deadline) và liên tục theo

dõi tiến độ giúp quản lý công việc hiệu quả hơn. Bài học rút ra là cần dành thời gian đủ lâu để lên kế hoạch và đảm bảo tính khả thi, linh hoạt của kế hoạch.

Trong quá trình triển khai thuật toán mã hóa và giải mã AES, nhóm đã tiến hành kiểm thử trên nhiều dữ liệu và điều kiện khác nhau để đánh giá tính chính xác và hiệu quả. Kết quả cho thấy rằng, kiểm tra liên tục không chỉ giúp phát hiện lỗi sớm mà còn đảm bảo rằng thuật toán hoạt động ổn định trong các trường hợp thực tế. Bài học rút ra là không nên bỏ qua khâu thử nghiệm và đánh giá kỹ càng trước khi đưa giải pháp vào thực tế.

Trong suốt quá trình thực hiện, nhóm đã đối mặt với nhiều vấn đề phát sinh ngoài dự kiến, như lỗi kỹ thuật, thay đổi yêu cầu, hoặc thiếu nguồn tài liệu phù hợp. Những thách thức này yêu cầu nhóm phải nhanh chóng điều chỉnh kế hoạch và tìm kiếm các giải pháp thay thế. Bài học quan trọng là cần giữ một tâm thế linh hoạt, luôn sẵn sàng thay đổi khi cần thiết, và phối hợp hiệu quả để giải quyết vấn đề.

Việc ghi chép đầy đủ từng bước thực hiện, các vấn đề gặp phải và cách khắc phục vô cùng quan trọng trong việc phát triển đề tài. Tài liệu hóa không chỉ giúp nhóm dễ dàng theo dõi quá trình làm việc mà còn tạo điều kiện thuận lợi cho việc mở rộng hoặc chỉnh sửa sau này. Bài học rút ra là cần duy trì thói quen ghi chép cẩn thận và tổ chức tài liệu khoa học.

3.3. Tính khả thi của chủ đề nghiên cứu

3.3.1. Thuận lợi

Một trong những thuận lợi lớn khi thực hiện đề tài đó là nguồn tài liệu về thuật toán AES rất phong phú và đa dạng. Với các tài liệu nghiên cứu, giáo trình và các hướng dẫn thực hành có sẵn, giúp nhóm dễ dàng tiếp cận và học hỏi. Đây là thuật toán đã được công nhận và sử dụng rộng rãi trong lĩnh vực bảo mật, đảm bảo độ tin cậy và giá trị ứng dụng thực tế.

Các công cụ lập trình phổ biến như Visual Studio, NetBeans, Eclipse, … cùng với các thư viện hỗ trợ mã hóa đều dễ dàng tiếp cận và sử dụng. Điều này tạo điều kiện thuận lợi để nhóm triển khai và thử nghiệm thuật toán AES.

Mỗi thành viên trong nhóm đều đã nắm được các kiến thức cơ sở về chuẩn mã hóa nâng cao AES. Bên cạnh đó các kỹ năng về lập trình của mỗi thành viên cũng tương đối ổn nhờ các môn học lập trình đã trải qua.

Ngoài ra, tính ứng dụng rộng rãi của AES cũng là một điểm mạnh. Trong bối cảnh chuyển đổi số, nhu cầu sử dụng mã hóa để bảo vệ dữ liệu cá nhân và

doanh nghiệp ngày càng tăng. Các ứng dụng thực tiễn của AES trong nhiều lĩnh vực như ngân hàng, y tế, giáo dục và thương mại điện tử đảm bảo rằng nghiên cứu này không chỉ mang tính học thuật mà còn có giá trị thực tiễn cao.

3.3.2. Khó khăn

Dù có nhiều thuận lợi, nhưng đề tài cũng đối mặt với một số khó khăn đáng kể. Trước hết, việc hiểu rõ và triển khai đúng thuật toán AES đòi hỏi kiến thức chuyên sâu về mã hóa và bảo mật. AES là một thuật toán phức tạp với nhiều bước xử lý như AddRoundKey, SubBytes, ShiftRows và MixColumns, nên yêu cầu nhóm phải nắm vững lý thuyết trước khi áp dụng thực tế để có thể triển khai chính xác.

Bên cạnh đó, một số tài liệu chuyên sâu về AES, đặc biệt là các tài liệu tiếng Anh hoặc yêu cầu kiến thức cao, cũng có thể gây khó khăn cho việc nghiên cứu và triển khai.

Ngoài ra, việc tối ưu hóa và đảm bảo hiệu suất của thuật toán khi làm việc với dữ liệu lớn là một thách thức không nhỏ. Việc kiểm thử AES trong nhiều điều kiện thực tế để đánh giá tính chính xác và hiệu quả cũng cần đầu tư thời gian và công sức.

TÀI LIỆU THAM KHẢO

[1]. Nguyễn Xuân Dũng, *Bảo mật thông tin - Mô Hình và ứng dụng,* NXB Thống kê, 2009.

[2]. Bùi Doãn Khanh, Nguyễn Đình Thúc, *Mã hóa thông tin - Lý thuyết và ứng dụng*, NXB Lao động xã hội, 2011.

[3]. Thái Thanh Tùng, *Mật mã học an toàn thông tin*, NXB Thông tin và truyền thông, 2011.

[4]. William Stallings**,** *Cryptography and Network Security Principles and Practices***,** Fourth Edition, Prentice Hall, 2005.

PHỤ LỤC 1. CHƯƠNG TRÌNH JAVA

Hàm cộng với khóa:

```
public static String[][] addRoundKey(String[][] state, String[][] key) {
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            int stateValue = Integer.parseInt(state[j][i], 16);
            int keyValue = Integer.parseInt(key[j][i], 16);
            int xorResult = stateValue ^ keyValue;
            state[j][i] = String.format("%02X", xorResult);
        }
    }
    return state;
}
```

Hàm thay thế byte:

```
public static String[][] subBytes(String[][] state) {
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            String hex = state[i][j];
            int row = Integer.parseInt(String.valueOf(hex.charAt(0)), 16);
            int col = Integer.parseInt(String.valueOf(hex.charAt(1)), 16);
            state[i][j] = S_BOX[row][col];
        }
    }
    return state;
}
```

Hàm thay thế byte đảo:

```
public static String[][] invSubBytes(String[][] state) {
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            String hex = state[i][j];
            int row = Integer.parseInt(String.valueOf(hex.charAt(0)), 16);
```

```
            int col = Integer.parseInt(String.valueOf(hex.charAt(1)), 16);
            state[i][j] = INV_S_BOX[row][col];
        }
    }
    return state;
}
```

Hàm dịch dòng:

```
public static String[][] shiftRows(String[][] state) {
    for (int i = 1; i < 4; i++) {
        String[] row = state[i];
        String[] shiftedRow = new String[4];
        System.arraycopy(row, i, shiftedRow, 0, 4 - i);
        System.arraycopy(row, 0, shiftedRow, 4 - i, i);
        state[i] = shiftedRow;
    }
    return state;
}
```

Hàm dịch dòng đảo:

```
public static String[][] invShiftRows(String[][] state) {
    for (int i = 1; i < 4; i++) {
        String[] row = state[i];
        String[] shiftedRow = new String[4];
        System.arraycopy(row, 4 - i, shiftedRow, 0, i);
        System.arraycopy(row, 0, shiftedRow, i, 4 - i);
        state[i] = shiftedRow;
    }
    return state;
}
```

Hàm trộn cột:

```
public static String[][] mixColumns(String[][] state) {
```

```
    int[][] mixColMatrix = {
        {2, 3, 1, 1}, {1, 2, 3, 1}, {1, 1, 2, 3}, {3, 1, 1, 2}};
    String[][] result = new String[4][4];
    for (int col = 0; col < 4; col++) {
        int[] tempColumn = new int[4];
        for (int row = 0; row < 4; row++) {
            int tempValue = 0;
            for (int k = 0; k < 4; k++) {
                int value = Integer.parseInt(state[k][col], 16);
                if (mixColMatrix[row][k] == 1) {
                    tempValue ^= value;
                } else if (mixColMatrix[row][k] == 2) {
                    tempValue ^= Integer.parseInt(mixCol2(state[k][col]), 16);
                } else if (mixColMatrix[row][k] == 3) {
                    tempValue ^= Integer.parseInt(mixCol3(state[k][col]), 16);
                }
            }
            tempColumn[row] = tempValue;
        }
        for (int row = 0; row < 4; row++) {
            result[row][col] = String.format("%02X", tempColumn[row]);
        }
    }
    return result;
}
```

Hàm trộn cột đảo:

```
public static String[][] invMixColumns(String[][] state) {
    int[][] invMixColMatrix = {
        {14, 11, 13, 9}, {9, 14, 11, 13}, {13, 9, 14, 11}, {11, 13, 9, 14}};
    String[][] result = new String[4][4];
    for (int col = 0; col < 4; col++) {
        int[] tempColumn = new int[4];
        for (int row = 0; row < 4; row++) {
```

```
            int tempValue = 0;
            for (int k = 0; k < 4; k++) {
                int value = Integer.parseInt(state[k][col], 16);
                if (invMixColMatrix[row][k] == 9) {
                    tempValue ^= Integer.parseInt(mixCol9(state[k][col]), 16);
                } else if (invMixColMatrix[row][k] == 11) {
                    tempValue ^= Integer.parseInt(mixCol11(state[k][col]), 16);
                } else if (invMixColMatrix[row][k] == 13) {
                    tempValue ^= Integer.parseInt(mixCol13(state[k][col]), 16);
                } else if (invMixColMatrix[row][k] == 14) {
                    tempValue ^= Integer.parseInt(mixCol14(state[k][col]), 16);
                }
            }
            tempColumn[row] = tempValue;
        }
        for (int row = 0; row < 4; row++) {
            result[row][col] = String.format("%02X", tempColumn[row]);
        }
    }
    return result;
}
```

Hàm mở rộng khóa:

```
public static String[][] keyExpansion(String[][] initialisationKey) {
    String[][] key = initialisationKey;
    String[][] expandKey = new String[4][44];
    String[] temp = new String[4];
    String[] word = new String[4];
    String[] w = new String[4];
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            expandKey[i][j] = key[i][j];
        }
    }
```

```
    for (int col = 4; col < 44; col++) {
        if ((col % 4) == 0) {
            int rCol = (col / 4);
            // Lấy giá trị từ khóa hiện tại và RCON (dịch trái 1 byte)
            for (int j = 0; j < 4; j++) {
                temp[j] = expandKey[j][col - 1];
                word[j] = RCON[j][rCol - 1];
                w[j] = expandKey[j][col - 4];
            }
            temp = rotWord(temp);          // Xoay từ
            temp = subWord(temp);          // Thay thế với S_BOX
            temp = xorHex(temp, word);  // XOR với RCON
            temp = xorHex(temp, w);       // XOR với từ trước đó
            for (int j = 0; j < 4; j++) {
                expandKey[j][col] = temp[j];
            }
        } else {
            for (int j = 0; j < 4; j++) {
                w[j] = expandKey[j][col - 4];
                temp[j] = expandKey[j][col - 1];
            }
            temp = xorHex(temp, w);  // XOR với từ trước đó
            for (int j = 0; j < 4; j++) {
                expandKey[j][col] = temp[j];
            }
        }
    }
    return expandKey;
}
```

Hàm mã hóa:

```
public static String encrypt(String plaintext, String inputKey) {
    String[][] originalKey = convertToMaxtrix4x4(inputKey.getBytes());
    String[][] key = keyExpansion(originalKey);
```

```
byte[] inputEncrypt = plaintext.getBytes();
inputEncrypt = addPadding(inputEncrypt);
String ciphertext = "";
for (int v = 0; v < inputEncrypt.length; v += 16) {
    byte[] subInputEncrypt = new byte[16];
    int index = 0;
    for (int h = v; h < v + 16; h++) {
        subInputEncrypt[index] = inputEncrypt[h];
        index++;
    }
    String[][] state = convertToMaxtrix4x4(subInputEncrypt);
    String[][] keyRound0 = new String[4][4];
    for (int row = 0; row < 4; row++) {
        for (int col = 0; col < 4; col++) {
            keyRound0[row][col] = key[row][col];
        }
    }
    state = addRoundKey(state, key);
    for (int i = 0; i < 10 - 1; i++) {
        state = subBytes(state);
        state = shiftRows(state);
        state = mixColumns(state);
        String[][] keyUsed = new String[4][4];
        for (int row = 0; row < 4; row++) {
            for (int col = (i + 1) * 4; col < (i + 1) * 4 + 4; col++) {
                keyUsed[row][col % 4] = key[row][col];
            }
        }
        state = addRoundKey(state, keyUsed);
    }
    state = subBytes(state);
    state = shiftRows(state);
    String[][] keyRound10 = new String[4][4];
    for (int row = 0; row < 4; row++) {
```

```
            for (int col = 40; col < 44; col++) {
                keyRound10[row][col % 4] = key[row][col];
            }
        }
        state = addRoundKey(state, keyRound10);
        String enCryptedText = "";
        for (int col = 0; col < 4; col++) {
            for (int row = 0; row < 4; row++) {
                enCryptedText += state[row][col];
            }
        }
        ciphertext += enCryptedText;
    }
    return ciphertext;
}
```

Hàm giải mã:

```
public static String decrypt(String ciphertext, String inputKey) {
    String[][] originalKey = convertToMaxtrix4x4(inputKey.getBytes());
    String[][] key = keyExpansion(originalKey);
    if (ciphertext.length() % 16 != 0) {
        return "Không hợp lệ!";
    }
    String decryptedText = "";
    for (int v = 0; v < ciphertext.length(); v += 32) {
        String subCiphertext = ciphertext.substring(v, v + 32);
        String[][] state = arrangeToMatrix4x4(subCiphertext);
        String[][] keyRound0 = new String[4][4];
        for (int row = 0; row < 4; row++) {
            for (int col = 40; col < 44; col++) {
                keyRound0[row][col % 4] = key[row][col];
            }
        }
        for (int i = 0; i < 10 - 1; i++) {
```

```
            state = invShiftRows(state);
            state = invSubBytes(state);
            String[][] keyUsed = new String[4][4];
            for (int row = 0; row < 4; row++) {
                for (int col = 40 - 4 * (i + 1); col < 40 - 4 * (i + 1) + 4; col++) {
                    keyUsed[row][col % 4] = key[row][col];
                }
            }
            state = addRoundKey(state, keyUsed);
            state = invMixColumns(state);
        }
        state = invShiftRows(state);
        state = invSubBytes(state);
        String[][] keyRound10 = new String[4][4];
        for (int row = 0; row < 4; row++) {
            for (int col = 0; col < 4; col++) {
                keyRound10[row][col] = key[row][col];
            }
        }
        state = addRoundKey(state, keyRound10);
        decryptedText += matric4x4ToString(state);
    }
    byte[] outputDecrypt = hexToByteArray(decryptedText);
    outputDecrypt = removePadding(outputDecrypt);
    String plaintext = "";
    for (byte b : outputDecrypt) {
        plaintext += String.format("%02X", b);
    }
    plaintext = hexToStringUTF8(plaintext);
    return plaintext;
}
```

PHỤ LỤC 2. CHƯƠNG TRÌNH C#

Hàm mã hóa:

```
private byte[] Encrypt(string plainText, byte[] key) {
   using(Aes aes = Aes.Create()) {
      aes.Key = key;
      aes.GenerateIV();
      aes.Mode = CipherMode.CBC;
      aes.Padding = PaddingMode.PKCS7;
      using(ICryptoTransform encryptor = aes.CreateEncryptor()) using(
         MemoryStream ms = new MemoryStream()) {
         ms.Write(aes.IV, 0, aes.IV.Length);
         using(CryptoStream cs =
               new CryptoStream(ms, encryptor, CryptoStreamMode.Write)) {
            byte[] inputBytes = Encoding.UTF8.GetBytes(plainText);
            cs.Write(inputBytes, 0, inputBytes.Length);
            cs.FlushFinalBlock();
         }
         return ms.ToArray();
      }
   }
}
```

Hàm giải mã:

```
private string Decrypt(byte[] cipherText, byte[] key) {
   using(Aes aes = Aes.Create()) {
      aes.Key = key;
      aes.Mode = CipherMode.CBC;
      aes.Padding = PaddingMode.PKCS7;
      using(MemoryStream ms = new MemoryStream(cipherText)) {
         byte[] iv = new byte[16];  // AES IV có độ dài cố định là 16 bytes
         ms.Read(iv, 0, iv.Length);
         aes.IV = iv;
```

```
            using(ICryptoTransform decryptor = aes.CreateDecryptor())
            using(CryptoStream cs = new CryptoStream(ms, decryptor,
                  CryptoStreamMode.Read))
            using(StreamReader sr = new StreamReader(cs, Encoding.UTF8)) {
               return sr.ReadToEnd();
            }
         }
      }
   }
```