**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

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

**KHOA CÔNG NGHỆ THÔNG TIN**

**Logo

Description automatically generated**

**BÀI BÁO CÁO**

**ỨNG DỤNG TRAO ĐỔI DỮ LIỆU TRONG LAN SỬ DỤNG MÃ HÓA HILL**

**Sinh viên thực hiện:** Nguyễn Thành Đạt – **MSSV:** 20H1120201

Lê Thành Phát – **MSSV:** 20H1120238

Phan Minh Hiển – **MSSV:** 20H1120208

**Lớp:** CN20CLCD

Thành phố Hồ Chí Minh – 2023

MỤC LỤC

[LỜI MỞ ĐẦU 1](#_Toc134272363)

[1. Lý do chọn đề tài 1](#_Toc134272364)

[2. Mục tiêu và phạm vi nghiên cứu 1](#_Toc134272365)

[I. Cơ sở lý thuyết. 2](#_Toc134272366)

[1.1. Ứng dụng trao đổi dữ liệu trong LAN. 2](#_Toc134272367)

[1.2. Mật mã Hill. 2](#_Toc134272368)

[*1.2.1. Các khái niệm cơ bản về mật mã học.* 2](#_Toc134272369)

[*1.2.2. Các phương pháp mã hóa thông thường* 3](#_Toc134272370)

[*1.2.3. Mật mã Hill.* 4](#_Toc134272371)

[*1.2.3.1. Tổng quan về mật mã Hill.* 4](#_Toc134272372)

[*1.2.3.2. Cách thức hoạt động và ví dụ về mật mã Hill.* 5](#_Toc134272373)

[II. Thiết kế và triển khai hệ thống 6](#_Toc134272374)

[2.2. Đặc tả hệ thống. 6](#_Toc134272375)

[2.2.1. Yêu cầu chức năng. 6](#_Toc134272376)

[2.2.2. Yêu cầu phi chức năng. 6](#_Toc134272377)

[2.2.3. Tính năng của hệ thống. 6](#_Toc134272378)

[2.2.4. Cấu trúc của hệ thống. 6](#_Toc134272379)

[2.2.5. Triển khai của hệ thống. 7](#_Toc134272380)

[2.3. Các công nghệ và công cụ được sử dụng 7](#_Toc134272381)

[2.4. Thiết kế giao diện người dùng 7](#_Toc134272382)

[III. Hoàn thiện sản phẩm. 9](#_Toc134272383)

[3.1. Thiết kế ứng dụng truyền dữ liệu qua LAN. 9](#_Toc134272384)

[3.2. Lập trình mật mã Hill. 10](#_Toc134272385)

[IV. Thực hiện thí nghiệm và đánh giá 13](#_Toc134272386)

[4.1. Mô tả thí nghiệm 13](#_Toc134272387)

[4.2. Phân tích và kết quả thí nghiệm 14](#_Toc134272388)

[KẾT LUẬN 15](#_Toc134272389)

[1. Tổng kết. 15](#_Toc134272390)

[2. Hướng phát triển của đề tài 15](#_Toc134272391)

[TÀI LIỆU THAM KHẢO 16](#_Toc134272392)

# LỜI MỞ ĐẦU

## 1. Lý do chọn đề tài

Trong thời đại công nghệ ngày càng phát triển và tiến bộ, việc bảo vệ dữ liệu khỏi những nguy cơ xâm nhập là một điều rất quan trọng. Mã hóa dữ liệu là một trong những phương pháp hiệu quả để đảm bảo tính bí mật và an toàn của thông tin.Một trong những hệ mật mã phổ biến là mật mà Hill.[Hệ mật mã này có nhiều ứng dụng trong đại số và trong mã hóa thông tin](https://www.studocu.com/vn/document/truong-dai-hoc-cong-nghe-thanh-pho-ho-chi-minh/dai-so-tuyen-tinh/mat-ma-hill-mat-ma-hill/29942438). Vì lý do trên và được sự gợi ý của thầy nên nhóm chúng em đã chọn đề tài: “Ứng dụng trao đổi dữ liệu trong LAN sử dụng mã hóa Hill”. Qua đề tài này, chúng em mong muốn tìm hiểu về nguyên lý hoạt động của hệ mã Hill, cách thức mã hóa và giải mã dữ liệu, cũng như các ưu nhược điểm của hệ mã này. Chúng em cũng hy vọng có thể xây dựng được một phần mềm ứng dụng trao đổi dữ liệu an toàn sử dụng mã hóa Hill trong mạng LAN.

## 2. Mục tiêu và phạm vi nghiên cứu

Mục tiêu của đề tài này là phát triển một ứng dụng trao đổi dữ liệu trong mạng LAN sử dụng mã hóa Hill, một thuật toán mã hóa tuyến tính có tính bảo mật cao và độ tin cậy. Đồng thời, đề tài cũng nhằm nghiên cứu các khái niệm cơ bản về mật mã học và các phương pháp mã hóa thông thường, cũng như tìm hiểu về các công nghệ và công cụ được sử dụng để triển khai ứng dụng trao đổi dữ liệu trong mạng LAN.

Phạm vi nghiên cứu của đề tài sẽ tập trung vào nghiên cứu và phát triển mật mã Hill nhằm tích hợp vào hệ thống trao đổi dữ liệu trong mạng LAN, bao gồm các chức năng như truy cập tài nguyên mạng, trao đổi dữ liệu và giải mã dữ liệu. Thí nghiệm và đánh giá kết quả cũng sẽ được tiến hành để đánh giá tính hiệu quả và tính ổn định của hệ thống trong quá trình hoạt động.

# I. Cơ sở lý thuyết.

## 1.1. Ứng dụng trao đổi dữ liệu trong LAN.

Hệ thống trao đổi dữ liệu trong mạng LAN là một phương tiện quan trọng để chia sẻ thông tin giữa các thiết bị trong mạng LAN. Để tạo ra hệ thống trao đổi dữ liệu trong mạng LAN, cần có các thiết bị mạng như máy tính, router, switch, modem, hub...và một hệ thống đường truyền mạng LAN.

Khi thực hiện trao đổi dữ liệu trong mạng LAN, thông tin được gửi đi từ nguồn (máy tính gửi) đến đích (máy tính nhận). Để thực hiện việc trao đổi dữ liệu một cách an toàn và bảo mật, cần sử dụng các phương pháp mã hóa dữ liệu như mã hóa Hill để bảo vệ thông tin trước khi gửi đi.

Cách thức hoạt động của ứng dụng trao đổi dữ liệu trong LAN thông thường như sau:

*Thiết bị gửi thông tin sẽ tạo ra gói tin dữ liệu và gửi nó đến router, switch hoặc hub trên mạng LAN.*

*Thiết bị mạng tiếp nhận gói tin dữ liệu và đưa nó đến đích cuối cùng trên mạng LAN.*

*Trong quá trình truyền tải, dữ liệu sẽ được mã hóa và giải mã trước khi được gửi và nhận.*

*Các thiết bị mạng trong mạng LAN sẽ tạo ra các bảng định tuyến để xác định đường đi của dữ liệu trên mạng LAN, đảm bảo dữ liệu được gửi đi và nhận được một cách nhanh chóng và chính xác.*

*Khi dữ liệu được nhận, nó sẽ được giải mã và hiển thị trên thiết bị nhận.*

## 1.2. Mật mã Hill.

## *1.2.1. Các khái niệm cơ bản về mật mã học.*

Mật mã học là một lĩnh vực của khoa học máy tính và toán học, liên quan đến các kỹ thuật để bảo vệ thông tin khỏi sự truy cập trái phép. Dưới đây là một số khái niệm cơ bản trong mật mã học:

*Mã hóa và giải mã:* Mã hóa là quá trình biến đổi thông tin ban đầu thành một dạng khác, gọi là bản mã, sao cho chỉ có người có khóa mới có thể đọc được. Giải mã là quá trình đảo ngược mã hóa để khôi phục lại thông tin ban đầu từ bản mã.

*Khóa:* Là một thông tin bí mật được sử dụng để điều khiển quá trình mã hóa và giải mã. Khóa có thể là một chuỗi bit, một số, một từ hoặc một ma trận. Khóa cần được lựa chọn và bảo mật cẩn thận để tránh bị đoán hay tấn công.

*Mật khẩu:* Là một loại khóa đặc biệt được sử dụng để xác thực danh tính của người dùng khi truy cập vào hệ thống, tài khoản hoặc tài liệu. Mật khẩu thường là một chuỗi ký tự hoặc số do người dùng tự chọn và nhớ. Mật khẩu cũng cần được mã hóa để bảo vệ chúng khỏi việc bị đánh cắp hay phá.

*Thuật toán mã hóa:* Là một tập hợp các quy tắc toán học được sử dụng để thực hiện quá trình mã hóa và giải mã. Có nhiều loại thuật toán mã hóa khác nhau, có thể phân loại theo cách sử dụng khóa, theo cấu trúc của bản rõ và bản mã, hoặc theo tính chất của quá trình biến đổi.

*Mật mã đối xứng:* Là loại mã hóa sử dụng cùng một khóa cho cả hai quá trình mã hóa và giải mã. Mật mã đối xứng có ưu điểm là tốc độ nhanh và đơn giản, nhưng có nhược điểm là khó quản lý và phân phối khóa.

*Mật mã bất đối xứng:* Là loại mã hóa sử dụng hai khóa khác nhau cho quá trình mã hóa và giải mã. Một khóa được gọi là khóa công khai và được công bố cho tất cả mọi người, một khóa được gọi là khóa bí mật và chỉ được giữ riêng cho chủ sở hữu. Mật mã bất đối xứng có ưu điểm là an toàn và dễ quản lý khóa, nhưng có nhược điểm là tốc độ chậm và phức tạp.

*Chữ ký số:* Là một phương pháp để xác thực tính xác thực và toàn vẹn của thông tin. Chữ ký số sử dụng mật mã bất đối xứng để tạo ra một chuỗi bit gọi là chữ ký kỹ thuật số cho thông tin. Chữ ký này chỉ có thể được tạo ra bởi chủ sở hữu của khóa bí mật và có thể được kiểm tra bởi bất kỳ ai có khóa công khai tương ứng.

## *1.2.2. Các phương pháp mã hóa thông thường*

Các phương pháp mã hóa thông thường là các phương pháp sử dụng các thuật toán toán học để biến đổi thông tin ban đầu thành một dạng khác, gọi là bản mã, sao cho chỉ có người có khóa mới có thể đọc được. Các phương pháp mã hóa thông thường có thể được phân loại theo cách sử dụng khóa, theo cấu trúc của bản rõ và bản mã, hoặc theo tính chất của quá trình biến đổi. Dưới đây là một số phương pháp mã hóa thông thường:

*Mã hóa cổ điển:* Là các phương pháp mã hóa sử dụng các kỹ thuật đơn giản như thay thế, hoán vị, dịch chuyển hoặc kết hợp các ký tự trong bản rõ để tạo ra bản mã. Các ví dụ về mã hóa cổ điển là mã César, mã Vigenère, mã Playfair, mã Hill...¹

Mã hóa một chiều (hash): Là các phương pháp mã hóa sử dụng một hàm toán học để ánh xạ một chuỗi dữ liệu bất kỳ thành một chuỗi có độ dài cố định, gọi là giá trị băm. Các giá trị băm không thể được giải mã để lấy lại dữ liệu ban đầu. Các ví dụ về mã hóa một chiều là MD5, SHA-1, SHA-256...²

*Mã hóa đối xứng (symmetric key encryption):* Là các phương pháp mã hóa sử dụng cùng một khóa cho cả hai quá trình mã hóa và giải mã. Mã hóa đối xứng có ưu điểm là tốc độ nhanh và đơn giản, nhưng có nhược điểm là khó quản lý và phân phối khóa. Các ví dụ về mã hóa đối xứng là DES, AES, RC4...³

*Mã hóa bất đối xứng (public key encryption):* Là các phương pháp mã hóa sử dụng hai khóa khác nhau cho quá trình mã hóa và giải mã. Một khóa được gọi là khóa công khai và được công bố cho tất cả mọi người, một khóa được gọi là khóa bí mật và chỉ được giữ riêng cho chủ sở hữu. Mã hóa bất đối xứng có ưu điểm là an toàn và dễ quản lý khóa, nhưng có nhược điểm là tốc độ chậm và phức tạp. Các ví dụ về mã hóa bất đối xứng là RSA, ECC, ElGamal...

## *1.2.3. Mật mã Hill.*

## *1.2.3.1. Tổng quan về mật mã Hill.*

Mật mã Hill là một phương pháp mã hóa đối xứng sử dụng phép nhân ma trận để mã hóa thông tin. Phương pháp này được đặt tên theo tên nhà toán học Lester Hill, người đã giới thiệu nó vào năm 1929.

Mật mã Hill sử dụng một khóa bí mật dưới dạng ma trận vuông K. Để mã hóa thông tin, các ký tự của văn bản sẽ được chuyển đổi thành các số tương ứng trong bảng mã ASCII, sau đó được sắp xếp thành các vector hàng của ma trận. Sau đó, mỗi vector hàng được nhân với ma trận K để tạo ra một vector hàng khác. Cuối cùng, các số trong vector mới này sẽ được chuyển đổi lại thành các ký tự trong bảng mã ASCII để tạo thành văn bản mã hóa.

Để giải mã thông tin, vector hàng của văn bản mã hóa sẽ được nhân với ma trận nghịch đảo của K, được tính toán trước đó, để tạo ra vector hàng gốc ban đầu.

Mật mã Hill có một số ưu điểm, bao gồm độ dài khóa có thể lớn hơn so với một số phương pháp mã hóa khác và khả năng mã hóa các ký tự đặc biệt. Tuy nhiên, phương pháp này có một số nhược điểm, bao gồm khả năng bị tấn công bằng cách sử dụng ma trận giả, và khả năng bị tấn công bằng cách sử dụng phân tích tần số để phá vỡ mã hóa. Do đó, việc sử dụng mật mã Hill cần phải cân nhắc kỹ lưỡng và áp dụng các biện pháp bảo mật phù hợp để tăng độ bảo mật của hệ thống mã hóa.

## *1.2.3.2. Cách thức hoạt động và ví dụ về mật mã Hill.*

Để mã hóa một chuỗi ký tự sử dụng mật mã Hill ta cần lưu ý rằng việc sử dụng mật mã này yêu cầu khái niệm về ma trận khả nghịch, nghịch đảo của ma trận và phép toán modulo. Do đó, việc hiểu và áp dụng đúng các khái niệm này là rất quan trọng.

Để thực hiện được mã hóa và giải mã mật mã Hill ta cần thực hiện các bước sau:

*Chuyển đổi chuỗi ký tự cần mã hóa thành các vector hàng của ma trận kích thước mxN, với m là kích thước của ma trận khóa và N là số lượng các ký tự trong chuỗi.*

*Xác định ma trận khóa K, một ma trận vuông kích thước mxm, để sử dụng cho việc mã hóa. Ma trận K này cần phải là một ma trận khả nghịch trong phép toán modulo.*

*Nhân từng vector hàng của ma trận với ma trận khóa K để thu được các vector hàng mã hóa. Kết quả được chuyển đổi thành chuỗi ký tự mã hóa.*

Để giải mã chuỗi ký tự mã hóa sử dụng mật mã Hill, ta cần thực hiện các bước sau:

*Chuyển đổi chuỗi ký tự mã hóa thành các vector hàng của ma trận kích thước mxN.*

*Xác định ma trận nghịch đảo của ma trận khóa K, và sử dụng nó để nhân từng vector hàng của chuỗi ký tự mã hóa để thu được các vector hàng gốc.*

*Chuyển đổi các vector hàng này thành chuỗi ký tự ban đầu.*

Ví dụ:

# II. Thiết kế và triển khai hệ thống

## 2.2. Đặc tả hệ thống.

## 2.2.1**. Yêu cầu chức năng.**

Cho phép người dùng trao đổi dữ liệu với nhau trong mạng LAN.

Mã hóa dữ liệu để đảm bảo tính bảo mật và độ tin cậy trong quá trình trao đổi.

Giải mã dữ liệu để có thể đọc được thông tin gửi đến

## 2.2.2. Yêu cầu phi chức năng.

Hệ thống phải có khả năng kết nối và truyền dữ liệu trong mạng LAN.

Hệ thống phải có khả năng tạo ra khóa mã hóa Hill để sử dụng trong quá trình mã hóa dữ liệu.

Hệ thống phải có khả năng kiểm tra tính đúng đắn và an toàn của dữ liệu trước khi mã hóa và sau khi giải mã.

## 2.2.3. Tính năng của hệ thống.

*Tạo khóa mã hóa Hill*: Hệ thống sẽ cho phép người dùng tạo ra ma trận khóa để sử dụng trong quá trình mã hóa.

*Mã hóa dữ liệu*: Hệ thống sẽ thực hiện mã hóa dữ liệu bằng cách áp dụng ma trận khóa Hill lên các khối dữ liệu nhỏ.

*Giải mã dữ liệu*: Hệ thống sẽ thực hiện giải mã dữ liệu bằng cách áp dụng ma trận khóa Hill nghịch đảo lên các khối dữ liệu được mã hóa.

*Kiểm tra tính đúng đắn và an toàn của dữ liệu*: Hệ thống sẽ thực hiện các kiểm tra bảo mật trước khi mã hóa dữ liệu và kiểm tra độ chính xác của dữ liệu sau khi giải mã.

## 2.2.4. Cấu trúc của hệ thống.

*Giao diện người dùng*: Hệ thống sẽ có giao diện đơn giản và dễ sử dụng để người dùng có thể thao tác dễ dàng.

*Module mã hóa*: Hệ thống sẽ có module mã hóa để thực hiện quá trình mã hóa dữ liệu.

*Module giải mã*: Hệ thống sẽ có module giải mã để thực hiện quá trình giải mã dữ liệu.

*Module tạo khóa*: Hệ thống sẽ có module tạo khóa để người dùng có thể tạo ra ma trận khóa cho quá trình mã hóa.

## 2.2.5. Triển khai của hệ thống.

*Cài đặt và cấu hình mạng LAN*: Cần cài đặt và cấu hình mạng LAN cho hệ thống để có thể truyền dữ liệu giữa các thiết bị trên mạng. Các thiết bị mạng bao gồm router, switch, các máy tính trên mạng.

*Triển khai phần trao đổi dữ liệu*: Sau khi mã hóa đã được triển khai, cần triển khai phần trao đổi dữ liệu. Điều này bao gồm viết code để gửi và nhận dữ liệu giữa các máy tính trên mạng. Cần kiểm tra và sửa lỗi nếu có.

*Triển khai mã hóa Hill*: Cần viết code để triển khai mã hóa Hill trên các máy tính trong mạng. Điều này bao gồm viết các hàm tính toán ma trận khóa, ma trận nghịch đảo và phép tính toán mã hóa. Sau khi viết xong, cần thực hiện kiểm tra và sửa lỗi nếu có.

*Kiểm tra và đánh giá hệ thống*: Sau khi triển khai xong, cần tiến hành kiểm tra và đánh giá hệ thống. Kiểm tra nên được thực hiện bằng cách gửi và nhận dữ liệu giữa các máy tính trên mạng. Nếu kết quả đạt yêu cầu, thì hệ thống đã được triển khai thành công.

## 2.3. Các công nghệ và công cụ được sử dụng

Ngôn ngữ lập trình: C#

Môi trường phát triển tích hợp (IDE): Visual Studio Code.

Git: Công cụ quản lý phiên bản Git có thể được sử dụng để quản lý mã nguồn của dự án và phát triển theo các phiên bản khác nhau.

Công cụ để phân tích và kiểm tra dữ liệu trao đổi trên mạng: Wireshark.

## 2.4. Thiết kế giao diện người dùng

*Giao diện đơn giản và dễ sử dụng*: Giao diện sẽ được thiết kế đơn giản, rõ ràng và dễ sử dụng để giúp người dùng có thể dễ dàng tiếp cận và sử dụng ứng dụng.

*Các phím chức năng rõ ràng*: Các phím chức năng sẽ được thiết kế rõ ràng và hiển thị trực quan để người dùng có thể dễ dàng sử dụng các chức năng mã hóa và giải mã.

*Hiển thị kết quả trực tiếp*: Khi người dùng nhập vào thông tin cần mã hóa hoặc giải mã, kết quả sẽ được hiển thị trực tiếp trên giao diện, giúp người dùng có thể kiểm tra kết quả và sửa chữa nếu cần.

*Thông báo lỗi chính xác*: Nếu người dùng nhập thông tin không đúng hoặc xảy ra lỗi, ứng dụng sẽ thông báo lỗi chính xác để giúp người dùng sửa chữa và tiếp tục sử dụng ứng dụng.

*Thân thiện với người dùng*: Giao diện sẽ được thiết kế để tăng tính thân thiện và gần gũi với người dùng, giúp người dùng cảm thấy thoải mái và dễ sử dụng ứng dụng.

# III. Hoàn thiện sản phẩm.

## Graphical user interface, application, Teams Description automatically generated3.1. Thiết kế ứng dụng truyền dữ liệu qua LAN.

Hình 1: Giao diện gửi và nhận tin nhắn

Hình 2: Ứng dụng gửi tin nhắn qua LAN

## 3.2. Lập trình mật mã Hill.

Code lập trình mật mã Hill kèm chú thích :

using System;

class HillCipher {

static void Main() {

// Define the plaintext message to be encrypted

string message = "Nguyen Thanh Dat sinh ngay 05/10/2002.";

// Define the encryption key matrix (must be a square matrix)

int[,] key = { { 2, 0, 1, 0 }, { 0, 1, 0, 0 }, { 1, 0, 3, 2 }, { 0, 0, 2, 1 } };

string encrypted = encrypt(message, key);

string decrypted = decrypt(encrypted, key);

Console.WriteLine("Message:" + message);

Console.WriteLine("Encrypted Message: " + encrypted);

Console.WriteLine("Decrypted Message: " + decrypted);

}

static string encrypt(string message, int[,] key)

{

int messageLength = message.Length;

string[] words = message.Split(' ');

int[] wordLengths = new int[words.Length];

for (int i = 0; i < words.Length; i++) {

wordLengths[i] = words[i].Length;

}

// remove spaces

String messageTemp = message;

// Ensure that the message length is a multiple of the key size

int padding = message.Length % key.GetLength(0);

int paddRight = 0;

if (padding != 0) {

paddRight = message.Length + (key.GetLength(0) - padding);

message = message.PadRight(paddRight, 'X');

paddRight = paddRight - messageTemp.Length;

}

// Define variables for the plaintext and ciphertext matrices

int[,] plaintext = new int[key.GetLength(0), message.Length / key.GetLength(0)];

int[,] ciphertext = new int[key.GetLength(0), message.Length / key.GetLength(0)];

// Convert the message to a matrix of numerical values based on the alphabet

int messageIndex = 0;

for (int i = 0; i < message.Length; i++) {

plaintext[messageIndex % key.GetLength(0), messageIndex / key.GetLength(0)] = message[i] - ' ';

messageIndex++;

// if (message[i] != ' ') {

// }

}

// Multiply the plaintext matrix by the key matrix to get the ciphertext matrix

for (int i = 0; i < key.GetLength(0); i++) {

for (int j = 0; j < messageIndex / key.GetLength(0); j++) {

for (int k = 0; k < key.GetLength(0); k++) {

ciphertext[i, j] += key[i, k] \* plaintext[k, j];

}

ciphertext[i, j] %= 94;

}

}

// Convert the ciphertext matrix back to a string of characters based on the alphabet

string output = "";

for (int i = 0; i < messageIndex / key.GetLength(0); i++) {

for (int j = 0; j < key.GetLength(0); j++) {

output += (char)(ciphertext[j, i] + ' ');

}

}

output = output + "~" + messageLength.ToString();

return output;

}

static string decrypt(string cipherText, int[,] keyMatrix)

{

string[] parts = cipherText.Split('~');

int messageLength = int.Parse(parts[1]);

cipherText = parts[0];

int n = keyMatrix.GetLength(0);

int m = (int)Math.Ceiling((double)cipherText.Length / n);

int[,] cipherMatrix = new int[n, m];

for (int i = 0; i < cipherText.Length; i++)

{

int row = i % n;

int col = i / n;

cipherMatrix[row, col] = (cipherText[i] - ' ') % 94;

}

int[,] adj = new int[n, n];

int det = determinantOfMatrix(keyMatrix, n);

det = det % 94;

if (det < 0)

det += 94;

int invDet = modInverse(det, 94);

adjoint(keyMatrix, adj);

for (int i = 0; i < n; i++)

for (int j = 0; j < n; j++)

adj[i, j] = (((adj[i, j] % 94) + 94) \* invDet) % 94;

int[,] plainMatrix = new int[n, m];

for (int i = 0; i < n; i++)

for (int j = 0; j < m; j++)

for (int k = 0; k < n; k++)

plainMatrix[i, j] += adj[i, k] \* cipherMatrix[k, j];

string plainText = "";

for (int i = 0; i < cipherText.Length; i++)

{

int row = i % n;

int col = i / n;

plainText += (char)((plainMatrix[row, col] % 94) + ' ');

}

// Split the output string into words based on the word lengths

string finalOutput = plainText.Substring(0, messageLength);

return finalOutput;

}

static void adjoint(int[,] mat, int[,] adj)

{

int n = adj.GetLength(0);

if (n == 1)

{

adj[0,0] = 1;

return;

}

int sign = 1;

int[,] temp = new int[n,n];

for (int i = 0; i < n; i++)

for (int j = 0; j < n; j++)

{

getCofactor(mat,temp,i,j,n);

sign = ((i + j) % 2 == 0) ? 1 : -1;

adj[j,i] = sign \* determinantOfMatrix(temp,n - 1);

}

}

static void getCofactor(int[,] mat, int[,] temp, int p, int q, int n)

{

int i = 0, j = 0;

for (int row = 0; row < n; row++)

{

for (int col = 0; col < n; col++)

{

if (row != p && col != q)

{

temp[i,j++] = mat[row,col];

if (j == n - 1)

{

j = 0;

i++;

}

}

}

}

}

static int determinantOfMatrix(int[,] mat, int n)

{

if (n == 1)

return mat[0,0];

int D = 0;

int[,] temp = new int[n,n];

int sign = 1;

for (int f = 0; f < n; f++)

{

getCofactor(mat,temp,0,f,n);

D += sign \* mat[0,f] \* determinantOfMatrix(temp,n - 1);

sign = -sign;

}

return D;

}

static void adjjacent(int[,] mat, int[,] adj)

{

int n = adj.GetLength(0);

if (n == 1)

{

adj[0,0] = 1;

return;

}

int sign = 1;

int[,] temp = new int[n,n];

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

{

getCofactor(mat,temp,i,j,n);

sign = ((i+j) % 2 == 0) ? 1 : -1;

adj[j,i] = sign \* determinantOfMatrix(temp,n - 1);

}

}

}

static int modInverse(int a, int m)

{

a = a % m;

for (int x = 1; x < m; x++)

{

if ((a \* x) % m == 1)

return x;

}

return 1;

}

}

# IV. Thực hiện thí nghiệm và đánh giá

## 4.1. Mô tả thí nghiệm

Để đánh giá hiệu suất của hệ thống Ứng dụng trao đổi dữ liệu trong LAN sử dụng mã hóa Hill, nhóm chúng em tiến hành cách bước sau các bước sau:

Thiết lập môi trường thử nghiệm: Sử dụng phần mềm đã được code hoàn chỉnh cũng với mạng LAN trên máy tính.

Cài đặt hệ thống: Cài đặt hệ thống trên máy và tiến hành khởi chạy.

Kiểm tra việc trao đổi dữ liệu: Sử dụng công cụ Wireshark để bắt gói tin trước và sau khi chèn mật mã hill.

Đánh giá kết quả: Dựa trên các tiêu chí như tốc độ trao đổi dữ liệu, độ chính xác và độ bảo mật để đánh giá kết quả thực hiện của hệ thống.

Lặp lại quá trình với các thông số khác nhau: Thực hiện thí nghiệm với các thông số khác nhau như kích thước dữ liệu, mật độ dữ liệu, và độ phức tạp của thuật toán mã hóa để đánh giá hiệu suất của hệ thống trong các tình huống khác nhau.

## 4.2. Phân tích và kết quả thí nghiệm

Hình 4: Dữ liệu đã được mã hóa bằng mật mã Hill.

Hình 3: Gửi dữ liệu khi chưa có bảo mật Hill

Kết quả trước và sau khi thực hiện thí nghiệm ( Hình 1 và Hình 2) đã cho ta thấy ứng dụng đã thực hiện thành công quá trình gửi và nhận dữ liệu. Ngoài ra thí nghiệm cũng cho ta thấy sự thành công khi chèn mật mã Hill, thông tin được bảo mật mà được mã hóa khi gửi đến cho người nhận. Thí nghiệm đã được thực hiện nhiều lần với từng cách thức khác nhau và đều cho ra kết quả như mong đợi.

# KẾT LUẬN

## 1. Tổng kết.

Sau khi thực hiện đề tài, nhóm chúng em đã thành công trong việc mã hóa dữ liệu bằng mật mã Hill, cụ thể là gửi tin nhắn từ sever cho client đã được mã hóa đảm bảo độ bảo mật và an toàn.

## 2. Hướng phát triển của đề tài

Tăng cường tính năng bảo mật của ứng dụng bằng cách thêm các phương pháp mã hóa mới, cải thiện thuật toán hiện tại hoặc sử dụng các công nghệ mã hóa mới để tăng cường tính bảo mật của hệ thống.

Phát triển ứng dụng với các tính năng mới, bao gồm tính năng chia sẻ tệp an toàn và tính năng mã hóa toàn bộ ổ đĩa hoặc thư mục, giúp người dùng bảo vệ các tài liệu quan trọng của họ. Ngoài ra, có thể phát triển tính năng quản lý người dùng để quản lý quyền truy cập cho các tài liệu được chia sẻ.

Xây dựng các phiên bản ứng dụng cho các nền tảng khác nhau, bao gồm các thiết bị di động, để đáp ứng nhu cầu sử dụng của người dùng trên nhiều thiết bị khác nhau.

Phát triển tính năng tích hợp lưu trữ đám mây để người dùng có thể lưu trữ tài liệu của mình trực tuyến và truy cập chúng từ bất kỳ đâu và bằng bất kỳ thiết bị nào.

# TÀI LIỆU THAM KHẢO

1. [Cách truyền dữ liệu nhanh giữa 2 máy tính Windows 10 hiệu quả (topthuthuat.com)](https://topthuthuat.com/chia-se-du-lieu-qua-mang-lan/)

2. [Mã hóa là gì ? 4 phương pháp mã hóa thông dụng (pavietnam.vn)](https://kb.pavietnam.vn/ma-hoa-la-gi-4-phuong-phap-ma-hoa-thong-dung.html)

3. [(231) Bài 17: Phân tích quá trình trao đổi dữ liệu giữa các thiết bị trong hệ thống mạng LAN - YouTube](https://www.youtube.com/watch?v=_IrU_DGsMoM&ab_channel=T%E1%BB%B1H%E1%BB%8DcCCNA)

4. Mã hóa là gì ? 4 phương pháp mã hóa thông dụng. https://kb.pavietnam.vn/ma-hoa-la-gi-4-phuong-phap-ma-hoa-thong-dung.html.

5. Mã hóa thông tin dữ liệu là gì? Các phương pháp mã hóa?. https://luatduonggia.vn/ma-hoa-thong-tin-du-lieu-la-gi-cac-phuong-phap-ma-hoa-pho-bien/.

6. Tại sao cần mã hóa mật khẩu ? Và các kiểu mã hóa cơ bản. https://viblo.asia/p/tai-sao-can-ma-hoa-mat-khau-va-cac-kieu-ma-hoa-co-ban-yMnKML0Q57P.