**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN MÔN HỌC:** NHẬP MÔN AN TOÀN, BẢO MẬT THÔNG TIN

**TÊN ĐỀ TÀI:** GỬI TÀI LIỆU EMAIL CÓ GIỚI HẠN THỜI GIAN

Sinh viên thực hiện:

|  |  |  |  |
| --- | --- | --- | --- |
| **Stt** | **Họ và tên** | **Mã sinh viên** | **Lớp** |
| 1 | Trần Văn Hiếu | 1771020264 | CNTT 17-03 |
| 2 | Nguỵ Phan Hưng | 1771020331 | CNTT 17-03 |
| 3 | Ngô Hoàng Huy | 1771020347 | CNTT 17-03 |
| 4 | Nguyễn Hữu Hùng | 1771020317 | CNTT 17-03 |

Hà Nội, 2025

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN MÔN HỌC:** NHẬP MÔN AN TOÀN, BẢO MẬT THÔNG TIN

**TÊN ĐỀ TÀI:** GỬI TÀI LIỆU EMAIL CÓ GIỚI HẠN THỜI GIAN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stt | Mã Sinh Viên | Họ và Tên | Ngày Sinh | Điểm | |
| Bằng Số | Bằng Chữ |
| 1 | 1771020264 | Trần Văn Hiếu | 06/04/2005 |  |  |
| 2 | 1771020331 | Nguỵ Phan Hưng | 05/09/2005 |  |  |
| 3 | 1771020347 | Ngô Hoàng Huy | 22/04/2025 |  |  |
| 4 | 1771020317 | Nguyễn Hữu Hùng | 06/05/2004 |  |  |

|  |  |
| --- | --- |
| CÁN BỘ CHẤM THI 1 | CÁN BỘ CHẤM THI 2 |
|
|  |  |

Hà Nội, 2025

**LỜI NÓI ĐẦU**

Trong bối cảnh kỷ nguyên số với những thách thức ngày càng lớn về an toàn dữ liệu, khi thông tin trên đường truyền có thể bị nghe lén, đánh cắp hoặc sửa đổi, việc trang bị kiến thức mật mã học đã trở nên vô cùng cấp thiết. Bài tập lớn của học phần Nhập môn An toàn, bảo mật thông tin được thực hiện nhằm mục đích vận dụng các kiến thức lý thuyết đã học vào việc giải quyết một bài toán thực tế, đóng vai trò là cầu nối quan trọng giữa lý thuyết và thực tiễn. Cụ thể, đề tài được chọn là "Gửi tài liệu email có giới hạn thời gian", mô phỏng một kịch bản phổ biến khi một công ty A cần gửi tài liệu nhạy cảm (

email.txt) cho nhân viên B với yêu cầu đặc biệt là tài liệu chỉ có thể được truy cập trong vòng 24 giờ.

Đây là một bài toán mang tính ứng dụng cao, phản ánh nhu cầu thực tế trong môi trường doanh nghiệp, tài chính, pháp lý, nơi thông tin không chỉ nhạy cảm mà còn có giá trị trong một khoảng thời gian nhất định. Để giải quyết bài toán này, một giao thức bảo mật hoàn chỉnh đã được thiết kế và triển khai , kết hợp các thuật toán mã hóa tiêu chuẩn: AES-CBC được sử dụng để bảo vệ tính bí mật của nội dung, chữ ký số RSA 2048-bit (PKCS#1 v1.5) được dùng để xác thực người gửi và trao đổi khóa an toàn, và hàm băm SHA-512 được dùng để kiểm tra tính toàn vẹn của dữ liệu. Báo cáo này sẽ trình bày một cách hệ thống toàn bộ quá trình thực hiện, bắt đầu từ việc phân tích sâu các yêu cầu bài toán, trình bày cơ sở lý thuyết và mô tả chi tiết giải pháp, sau đó đi vào phân tích việc triển khai mã nguồn , trình bày các kết quả thử nghiệm và đánh giá hiệu quả , qua đó thể hiện tư duy xây dựng một hệ thống đảm bảo an toàn thông tin một cách toàn diện.

MỤC LỤC

[**CHƯƠNG 1: ĐẶT VẪN ĐỀ VÀ PHÂN TÍCH YÊU CẦU** 6](#_Toc202000432)

[**1.1** **Bối cảnh bài toán** 6](#_Toc202000433)

[**1.2** **Phân tích các yêu cầu bảo mật** 6](#_Toc202000434)

[***1.2.1*** ***Tính bảo mật******(Confidentiality)*** 6](#_Toc202000435)

[***1.2.2*** ***Tính Toàn vẹn (Integrity)*** 6](#_Toc202000436)

[***1.2.3*** ***Tính Xác thực (Authentication)*** 7](#_Toc202000437)

[***1.2.4*** ***Giới hạn Thời gian Truy cập (Time-Limited Access)*** 7](#_Toc202000438)

[**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT VÀ MÔ TẢ GIẢI PHÁP** 8](#_Toc202000439)

[2.1. Cơ sở lý thuyết 8](#_Toc202000440)

[*2.1.1. Thuật toán mã hóa đối xứng AES (Advanced Encryption Standard)* 8](#_Toc202000441)

[*2.1.2. Thuật toán mã hóa bất đối xứng RSA (Rivest-Shamir-Adleman)* 8](#_Toc202000442)

[*2.1.3. Hàm băm mật mã SHA-512 (Secure Hash Algorithm 512)* 8](#_Toc202000443)

[2.2. Mô tả giải pháp và luồng xử lý chi tiết 9](#_Toc202000444)

[*2.2.1. Bước 1: Bắt tay (Handshake)* 9](#_Toc202000445)

[*2.2.2. Bước 2: Xác thực và Trao đổi khóa* 9](#_Toc202000446)

[*2.2.3. Bước 3: Mã hóa và Truyền dữ liệu* 9](#_Toc202000447)

[2.2.4. Bước 4: Xử lý phía Người nhận và Phản hồi 10](#_Toc202000448)

[**CHƯƠNG 3: PHÂN TÍCH VÀ TRIỂN KHAI CHƯƠNG TRÌNH** 11](#_Toc202000449)

[3.1. Môi trường và Công cụ Phát triển 11](#_Toc202000450)

[*3.1.1. Ngôn ngữ lập trình* 11](#_Toc202000451)

[*3.1.2. Các thư viện sử dụng* 11](#_Toc202000452)

[*3.2. Cấu trúc Mã nguồn* 11](#_Toc202000453)

[3.3. Phân tích các Hàm chức năng quan trọng 12](#_Toc202000454)

[*3.3.1. Chức năng tạo và trao đổi khóa an toàn* 12](#_Toc202000455)

[*3.3.2. Chức năng tạo và xác thực Chữ ký số* 12](#_Toc202000456)

[*3.3.3. Chức năng Mã hóa và Giải mã dữ liệu* 13](#_Toc202000457)

[*3.3.4. Chức năng kiểm tra phía Người nhận* 13](#_Toc202000458)

[**CHƯƠNG 4: THỬ NGHIỆM VÀ ĐÁNH GIÁ** 14](#_Toc202000459)

[4.1. Kịch bản Thử nghiệm 14](#_Toc202000460)

[*4.1.1. Kịch bản hoạt động thành công* 14](#_Toc202000461)

[*4.1.2. Kịch bản xử lý lỗi* 14](#_Toc202000462)

[4.2. Kết quả Thử nghiệm 15](#_Toc202000463)

[*4.2.1. Kết quả thực thi các kịch bản* 15](#_Toc202000464)

[4.3. Đánh giá hiệu quả 18](#_Toc202000465)

[**CHƯƠNG 5. KẾT LUẬN** 19](#_Toc202000466)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 20](#_Toc202000467)

**MỤC LỤC HÌNH ẢNH**

[*Hình ảnh 1: Nội dung file email* 15](#_Toc202000513)

[*Hình ảnh 2: Giao diện console của chương trình Người Nhận (Server) khi khởi tạo thành công.* 15](#_Toc202000514)

[*Hình ảnh 3: Kết quả thực thi của chương trình Người Gửi (Client) trong kịch bản thành công* 15](#_Toc202000515)

[*Hình ảnh 4:Toàn bộ nhật ký xử lý một giao dịch thành công của chương trình Người Nhận.* 16](#_Toc202000516)

[*Hình ảnh 5: Nội dung file DECRYPTED sau khi giải mã thành công* 16](#_Toc202000517)

[*Hình ảnh 6: Kết quả thực thi của chương trình Người Gửi (Client) trong kịch bản lỗi hết hạn* 16](#_Toc202000518)

[*Hình ảnh 7: Toàn bộ nhật ký xử lý một giao dịch bị lỗi hết hạn của chương trình Người Nhận.* 17](#_Toc202000519)

[*Hình ảnh 8: Kết quả thực thi của chương trình Người Gửi (Client) trong kịch bản lỗi toàn vẹn* 17](#_Toc202000520)

[*Hình ảnh 9: Toàn bộ nhật ký xử lý một giao dịch bị lỗi toàn vẹn của chương trình Người Nhận.* 17](#_Toc202000521)

[*Hình ảnh 10: Kết quả thực thi của chương trình Người Gửi (Client) trong kịch bản lỗi xác thực* 18](#_Toc202000522)

[*Hình ảnh 11: Toàn bộ nhật ký xử lý một giao dịch bị lỗi xác thực của chương trình Người Nhận.* 18](#_Toc202000523)

# **CHƯƠNG 1: ĐẶT VẪN ĐỀ VÀ PHÂN TÍCH YÊU CẦU**

* 1. **Bối cảnh bài toán**

Trong bối cảnh cách mạng công nghiệp 4.0, việc trao đổi thông tin qua mạng máy tính đã trở thành một phần không thể thiếu trong hoạt động của các tổ chức và doanh nghiệp. Tuy nhiên, môi trường mạng cũng tiềm ẩn nhiều rủi ro về an toàn thông tin như bị nghe lén, đánh cắp, hay sửa đổi dữ liệu. Do đó, việc xây dựng các hệ thống truyền tin an toàn là một yêu cầu cấp thiết.

Bài toán của đề tài được đặt ra trong một kịch bản thực tế và phổ biến: "Công ty A cần chuyển email.txt chứa thông tin nhạy cảm đến nhân viên B, chỉ cho phép mở trong 24 giờ kể từ khi gửi". Đây là một nghiệp vụ đòi hỏi các giải pháp bảo mật mạnh mẽ để đảm bảo các mục tiêu an toàn thông tin cốt lõi. Nếu không có cơ chế bảo vệ, các rủi ro sau có thể xảy ra:

* **Rủi ro về tính bí mật:** Kẻ tấn công trên đường truyền có thể bắt gói tin và đọc được toàn bộ nội dung nhạy cảm trong file email.txt.
* **Rủi ro về tính toàn vẹn:** Kẻ tấn công có thể thay đổi nội dung của file, dẫn đến việc nhân viên B nhận được thông tin sai lệch, gây ra những hậu quả nghiêm trọng.
* **Rủi ro về tính xác thực:** Kẻ tấn công có thể giả mạo là Công ty A để gửi thông tin lừa đảo hoặc độc hại cho nhân viên B.
* **Rủi ro truy cập trái phép kéo dài:** Kể cả khi đã nhận được file, nếu không có cơ chế giới hạn thời gian, file tài liệu có thể tồn tại trên máy của nhân viên B và bị truy cập bởi người khác sau khi đã hết thời hạn hiệu lực của thông tin.
  1. **Phân tích các yêu cầu bảo mật**
     1. ***Tính bảo mật******(Confidentiality)***

**Yêu cầu:** Dữ liệu trong file email.txt phải được mã hóa để đảm bảo tính bí mật trong suốt quá trình truyền tải. Nội dung file phải trở nên hoàn toàn không thể đọc được đối với bất kỳ ai không có thẩm quyền.

**Giải pháp áp dụng:** Để đáp ứng yêu cầu này, đề tài chỉ định sử dụng thuật toán mã hóa đối xứng **AES (Advanced Encryption Standard)** với chế độ hoạt động **CBC (Cipher Block Chaining)**. AES là một trong những chuẩn mã hóa mạnh mẽ và được sử dụng rộng rãi nhất hiện nay, phù hợp để mã hóa các khối dữ liệu lớn như nội dung file.

* + 1. ***Tính Toàn vẹn (Integrity)***

**Yêu cầu:** Phải có cơ chế để kiểm tra và đảm bảo rằng dữ liệu mà nhân viên B nhận được không hề bị thay đổi hay giả mạo so với dữ liệu gốc mà Công ty A đã gửi.

**Giải pháp áp dụng:** Giải pháp được đề tài yêu cầu là sử dụng hàm băm mật mã **SHA-512**. Trước khi gửi, một giá trị băm (hash) của gói tin sẽ được tính toán. Phía người nhận sẽ tính lại giá trị băm này và so sánh. Bất kỳ sự khác biệt nào, dù là nhỏ nhất, cũng sẽ bị phát hiện, qua đó đảm bảo tính toàn vẹn của dữ liệu

* + 1. ***Tính Xác thực (Authentication)***

**Yêu cầu:** Hệ thống phải có khả năng xác minh danh tính của người gửi để đảm bảo nhân viên B biết chắc chắn rằng thông tin mình nhận được thực sự đến từ Công ty A.

**Giải pháp áp dụng:** Đề tài yêu cầu sử dụng kỹ thuật **chữ ký số** dựa trên thuật toán mã hóa bất đối xứng **RSA với độ dài khóa 2048-bit**. Người gửi sẽ dùng khóa bí mật (private key) của mình để "ký" lên một phần dữ liệu (metadata), và người nhận sẽ dùng khóa công khai (public key) của người gửi để xác thực chữ ký đó. Điều này vừa đảm bảo danh tính người gửi, vừa chống lại việc chối bỏ trách nhiệm.

* + 1. ***Giới hạn Thời gian Truy cập (Time-Limited Access)***

**Yêu cầu:** Đây là yêu cầu nghiệp vụ đặc thù của bài toán, file email.txt chỉ có thể được truy cập trong vòng **24 giờ** kể từ thời điểm gửi.

**Giải pháp áp dụng:** Chương trình sẽ đính kèm một **dấu thời gian hết hạn (expiration)** vào trong gói tin được gửi đi. Trước khi thực hiện giải mã, phía người nhận bắt buộc phải kiểm tra thời gian hiện tại của hệ thống. Nếu thời gian hiện tại đã vượt quá thời gian hết hạn trong gói tin, chương trình sẽ từ chối truy cập và gửi thông báo lỗi.

# **CHƯƠNG 2: CƠ SỞ LÝ THUYẾT VÀ MÔ TẢ GIẢI PHÁP**

## 2.1. Cơ sở lý thuyết

### *2.1.1. Thuật toán mã hóa đối xứng AES (Advanced Encryption Standard)*

AES là chuẩn mã hóa đối xứng được chính phủ Hoa Kỳ và các tổ chức trên toàn thế giới tin dùng để bảo vệ thông tin nhạy cảm. Đây là thuật toán mã hóa khối, hoạt động trên các khối dữ liệu có kích thước cố định (128 bit) và sử dụng cùng một khóa cho cả quá trình mã hóa và giải mã.

* **Vai trò trong dự án:** AES được sử dụng để mã hóa nội dung chính của file email.txt. Lý do là vì AES có tốc độ xử lý rất nhanh và hiệu quả khi mã hóa các khối dữ liệu lớn, phù hợp với việc mã hóa toàn bộ nội dung một file.
* **Chế độ hoạt động:** Đề tài yêu cầu sử dụng chế độ **CBC (Cipher Block Chaining)**. Trong chế độ này, mỗi khối văn bản rõ trước khi được mã hóa sẽ được kết hợp (XOR) với khối bản mã ngay trước đó. Điều này đảm bảo rằng nếu hai khối văn bản rõ giống nhau, chúng sẽ tạo ra các khối bản mã khác nhau, giúp tăng cường tính an toàn và chống lại một số loại tấn công. Chế độ CBC yêu cầu một vector khởi tạo (IV) ngẫu nhiên và duy nhất cho mỗi quá trình mã hóa

### *2.1.2. Thuật toán mã hóa bất đối xứng RSA (Rivest-Shamir-Adleman)*

RSA là thuật toán mã hóa bất đối xứng phổ biến nhất, hoạt động dựa trên một cặp khóa: một khóa công khai (Public Key) để mã hóa và một khóa bí mật (Private Key) để giải mã. Bất cứ ai cũng có thể sử dụng khóa công khai để mã hóa dữ liệu, nhưng chỉ người sở hữu khóa bí mật tương ứng mới có thể giải mã được.

Trong dự án này, RSA có hai vai trò chính theo yêu cầu của đề tài:

1. **Trao đổi khóa an toàn:** Vì không thể gửi khóa đối xứng AES một cách an toàn qua kênh truyền không tin cậy, người gửi sẽ sử dụng **khóa công khai RSA của người nhận** để mã hóa khóa phiên AES trước khi gửi đi. Chỉ có người nhận mới có thể dùng khóa bí mật RSA của mình để giải mã và lấy lại được khóa AES.
2. **Tạo chữ ký số để xác thực:** Để chứng minh danh tính, người gửi sẽ dùng **khóa bí mật RSA của chính mình** để "ký" lên dữ liệu (cụ thể là hash của metadata). Người nhận sau đó có thể dùng

**khóa công khai RSA của người gửi** để xác thực chữ ký này, qua đó đảm bảo người gửi là thật và dữ liệu không bị chối bỏ.

Đề tài yêu cầu sử dụng RSA với độ dài khóa

**2048-bit** và đệm **PKCS#1 v1.5**, một tiêu chuẩn an toàn và phổ biến

### *2.1.3. Hàm băm mật mã SHA-512 (Secure Hash Algorithm 512)*

SHA-512 là một hàm băm mật mã thuộc họ SHA-2, nhận đầu vào là một chuỗi dữ liệu có độ dài bất kỳ và tạo ra một "dấu vân tay" có độ dài cố định là 512 bit.

* **Vai trò trong dự án:** SHA-512 được sử dụng để đảm bảo tính toàn vẹn của dữ liệu được truyền đi. Người gửi sẽ tính giá trị hash của gói tin trước khi gửi. Người nhận, sau khi nhận được gói tin, sẽ tính lại giá trị hash trên chính dữ liệu đó và so sánh với giá trị hash nhận được. Nếu hai giá trị này khớp nhau, có nghĩa là dữ liệu không bị thay đổi trong quá trình truyền.

## 2.2. Mô tả giải pháp và luồng xử lý chi tiết

### *2.2.1. Bước 1: Bắt tay (Handshake)*

Đây là bước khởi tạo nhằm đảm bảo hai bên đã sẵn sàng cho việc giao tiếp.

* Người gửi (Sender) gửi một tin nhắn đơn giản là

"Hello!" đến Người nhận (Receiver).

* Nếu Người nhận đang hoạt động và sẵn sàng, nó sẽ trả lời bằng một tin nhắn

"Ready!". Quá trình chỉ tiếp tục khi Người gửi nhận được phản hồi này.

### *2.2.2. Bước 2: Xác thực và Trao đổi khóa*

Bước này nhằm thiết lập một kênh liên lạc tin cậy và chia sẻ khóa mã hóa một cách an toàn.

* **Phía Người gửi:**
  1. Tạo một khóa phiên (Session Key) 256-bit hoàn toàn ngẫu nhiên để sử dụng cho việc mã hóa AES.
  2. Tạo metadata bao gồm các thông tin như tên file và dấu thời gian (

timestamp).

* 1. Ký số vào metadata này bằng cách sử dụng khóa bí mật RSA 2048-bit của mình. Quá trình này tạo ra một chữ ký số (

signature).

* 1. Mã hóa khóa phiên AES ở trên bằng cách sử dụng khóa công khai RSA 2048-bit của Người nhận.
  2. Gửi khóa phiên đã mã hóa và chữ ký số cho Người nhận.

### *2.2.3. Bước 3: Mã hóa và Truyền dữ liệu*

Sau khi đã có kênh an toàn, Người gửi tiến hành mã hóa và gửi dữ liệu chính.

* **Phía Người gửi:**
  1. Tạo một vector khởi tạo (IV) 128-bit ngẫu nhiên cho thuật toán AES-CBC.
  2. Sử dụng khóa phiên AES đã tạo ở Bước 2 và IV để mã hóa toàn bộ nội dung file

email.txt, tạo ra bản mã (ciphertext).

* 1. Tạo một dấu thời gian hết hạn (expiration), ví dụ là thời gian hiện tại cộng thêm 24 giờ.
  2. Tính toán một giá trị băm SHA-512 cuối cùng để đảm bảo tính toàn vẹn cho cả gói tin:

hash = SHA-512(IV || ciphertext || expiration).

* 1. Đóng gói tất cả các thành phần cần thiết vào một cấu trúc dữ liệu duy nhất (ví dụ: JSON) bao gồm:

iv, cipher, hash, sig (chữ ký từ Bước 2), và exp (thời gian hết hạn).

* 1. Gửi gói tin này cho Người nhận.

### 2.2.4. Bước 4: Xử lý phía Người nhận và Phản hồi

Đây là bước cuối cùng, nơi Người nhận kiểm tra, giải mã và phản hồi.

* **Phía Người nhận:**
  1. Nhận toàn bộ gói tin từ Người gửi.
  2. Thực hiện kiểm tra bảo mật theo thứ tự:
     + **Kiểm tra thời hạn:** So sánh thời gian hiện tại với trường exp trong gói tin. Nếu đã quá hạn, hủy bỏ quá trình.
     + **Kiểm tra tính toàn vẹn:** Tự tính toán lại SHA-512(IV || ciphertext || expiration) và so sánh với giá trị hash nhận được. Nếu không khớp, hủy bỏ quá trình.
     + **Kiểm tra chữ ký:** Sử dụng khóa công khai của Người gửi để xác thực chữ ký sig trên metadata. Nếu không hợp lệ, hủy bỏ quá trình.
  3. **Xử lý kết quả:**
     + **Nếu tất cả kiểm tra đều hợp lệ:**
       - Dùng khóa bí mật RSA của mình để giải mã khóa phiên AES.
       - Dùng khóa phiên AES và IV để giải mã ciphertext và khôi phục lại nội dung file email.txt gốc.
       - Lưu file đã giải mã.
       - Gửi một tin nhắn xác nhận thành công (

ACK) về cho Người gửi.

* + - **Nếu có bất kỳ kiểm tra nào thất bại:**
      * Từ chối gói tin, không thực hiện giải mã.
      * Gửi một tin nhắn báo lỗi (

NACK) kèm lý do (ví dụ: lỗi toàn vẹn hoặc hết hạn) về cho Người gửi.

# **CHƯƠNG 3: PHÂN TÍCH VÀ TRIỂN KHAI CHƯƠNG TRÌNH**

## 3.1. Môi trường và Công cụ Phát triển

### *3.1.1. Ngôn ngữ lập trình*

Chương trình được xây dựng và phát triển bằng ngôn ngữ lập trình **Python**. Ngôn ngữ này được lựa chọn dựa trên sự linh hoạt, cú pháp rõ ràng, và hệ sinh thái thư viện phong phú, đặc biệt là trong lĩnh vực an toàn thông tin và mạng máy tính, phù hợp với định hướng của học phần.

### *3.1.2. Các thư viện sử dụng*

Để triển khai các thuật toán và giao thức được yêu cầu, chương trình đã sử dụng các thư viện sau:

* **pycryptodome**: Đây là thư viện mật mã mạnh mẽ và toàn diện, được sử dụng để triển khai tất cả các thuật toán cốt lõi theo yêu cầu của đề tài, bao gồm:
  + **AES**: Để mã hóa và giải mã nội dung file email.txt.
  + **RSA**: Để tạo cặp khóa, tạo và xác thực chữ ký số, cũng như mã hóa và giải mã khóa phiên AES.
  + **SHA-512**: Để tạo giá trị băm phục vụ việc kiểm tra tính toàn vẹn của dữ liệu.
* **socket**: Thư viện chuẩn của Python, được sử dụng để xây dựng lớp giao tiếp mạng dựa trên giao thức TCP, cho phép hai chương trình Người gửi và Người nhận kết nối và trao đổi dữ liệu với nhau trên môi trường mạng cục bộ (Local).
* **Các thư viện chuẩn khác**:
  + **json**: Để đóng gói các thành phần dữ liệu (IV, bản mã, hash, chữ ký,...) thành một chuỗi có cấu trúc, dễ dàng cho việc truyền tải và phân tích.
  + **datetime**: Để xử lý các tác vụ liên quan đến thời gian, bao gồm việc tạo dấu thời gian (timestamp) và thời gian hết hạn (expiration).
  + **os**: Để tương tác với hệ điều hành, chủ yếu dùng để tạo ra các dữ liệu ngẫu nhiên an toàn (khóa phiên, IV).
  + **struct**: Để đóng gói và giải nén độ dài của tin nhắn, đảm bảo việc truyền nhận dữ liệu lớn qua socket được toàn vẹn và không bị khóa chéo (deadlock).

## *3.2. Cấu trúc Mã nguồn*

Chương trình được thiết kế theo dạng module, tách biệt rõ ràng các chức năng, bao gồm 3 file chính:

* **generate\_keys.py**: Là một script tiện ích chạy một lần duy nhất để khởi tạo môi trường. Nhiệm vụ của nó là sinh ra các cặp khóa RSA (khóa công khai và khóa bí mật) với độ dài 2048-bit cho cả Người gửi và Người nhận, sau đó lưu chúng vào các file .pem tương ứng.
* **sender.py**: Đóng vai trò là chương trình của Người gửi (Công ty A). Tệp này chịu trách nhiệm cho các công việc:
  1. Khởi tạo kết nối đến Người nhận.
  2. Thực hiện quá trình bắt tay (Handshake).
  3. Tạo khóa phiên AES, ký số metadata, mã hóa khóa phiên.
  4. Mã hóa nội dung file email.txt.
  5. Tính toán giá trị hash toàn vẹn.
  6. Đóng gói tất cả dữ liệu và gửi cho Người nhận.
  7. Đợi và hiển thị phản hồi ACK/NACK từ Người nhận.
* **receiver.py**: Đóng vai trò là chương trình của Người nhận (Nhân viên B). Tệp này chịu trách nhiệm:
  1. Lắng nghe kết nối từ Người gửi.
  2. Thực hiện quá trình bắt tay.
  3. Nhận gói tin dữ liệu hoàn chỉnh.
  4. Thực hiện tuần tự các bước kiểm tra bảo mật: thời hạn, tính toàn vẹn (hash), và tính xác thực (chữ ký số).
  5. Nếu tất cả các bước kiểm tra thành công, tiến hành giải mã và lưu file.
  6. Gửi phản hồi ACK (thành công) hoặc NACK (thất bại) tương ứng về cho Người gửi.

## 3.3. Phân tích các Hàm chức năng quan trọng

### *3.3.1. Chức năng tạo và trao đổi khóa an toàn*

Để mã hóa file email.txt, hai bên cần thống nhất một khóa bí mật (khóa phiên AES). Quá trình trao đổi khóa này phải được bảo mật.

* **Tại sender.py**:
  1. Một khóa phiên AES 256-bit được tạo ngẫu nhiên bằng os.urandom(32).
  2. Chương trình tải khóa công khai RSA của người nhận từ file receiver\_public\_key.pem.
  3. Khóa phiên AES này sau đó được mã hóa bằng khóa công khai RSA của người nhận. Thao tác này đảm bảo rằng chỉ người nhận, người duy nhất sở hữu khóa bí mật RSA tương ứng, mới có thể giải mã và đọc được khóa phiên AES.

### *3.3.2. Chức năng tạo và xác thực Chữ ký số*

Để xác thực danh tính Người gửi, chương trình sử dụng chữ ký số.

* **Tại sender.py**:
  1. Tạo metadata (ví dụ: tên file và timestamp).
  2. Tải khóa bí mật RSA của người gửi từ file sender\_private\_key.pem.
  3. Dùng khóa bí mật này để ký lên metadata, tạo ra chữ ký số signature.
* **Tại receiver.py**:
  1. Nhận metadata và signature từ người gửi.
  2. Tải khóa công khai RSA của người gửi từ file sender\_public\_key.pem.
  3. Sử dụng hàm xác thực của thư viện mật mã để kiểm tra xem signature có hợp lệ với metadata và khóa công khai của người gửi hay không. Nếu hợp lệ, danh tính của người gửi được xác nhận.

### *3.3.3. Chức năng Mã hóa và Giải mã dữ liệu*

Đây là chức năng bảo vệ nội dung chính của file.

* **Tại sender.py**:
  1. Tạo một vector khởi tạo (IV) 128-bit ngẫu nhiên.
  2. Sử dụng khóa phiên AES (đã tạo ở mục 3.3.1) và IV để thiết lập một đối tượng mã hóa AES ở chế độ CBC.
  3. Đọc nội dung file email.txt, thêm phần đệm (padding) để đảm bảo độ dài dữ liệu là bội số của kích thước khối, sau đó tiến hành mã hóa để tạo ra ciphertext.
* **Tại receiver.py**:
  1. Sau khi đã giải mã thành công khóa phiên AES, chương trình thực hiện quy trình ngược lại.
  2. Sử dụng khóa phiên AES và IV nhận được để thiết lập đối tượng giải mã.
  3. Giải mã ciphertext và loại bỏ phần đệm để khôi phục lại nội dung gốc của file.

### *3.3.4. Chức năng kiểm tra phía Người nhận*

Đây là chốt chặn bảo mật quan trọng nhất ở phía Người nhận, thực hiện một chuỗi các bước kiểm tra trước khi tin tưởng dữ liệu.

* **Tại receiver.py**, sau khi nhận được gói tin, chương trình sẽ:
  1. **Kiểm tra Thời hạn:** Phân tích cú pháp trường expiration và so sánh với thời gian UTC hiện tại. Nếu thời\_gian\_hiện\_tại > thời\_gian\_hết\_hạn, chương trình gửi NACK (timeout) và hủy bỏ.
  2. **Kiểm tra Toàn vẹn:** Tính toán lại giá trị hash SHA-512 của IV, ciphertext và expiration nhận được. So sánh kết quả này với giá trị hash có sẵn trong gói tin. Nếu chúng không khớp, chương trình gửi NACK (integrity) và hủy bỏ.
  3. **Kiểm tra Chữ ký:** Thực hiện quy trình xác thực chữ ký số như mô tả ở mục 3.3.2. Nếu chữ ký không hợp lệ, chương trình gửi NACK (signature) và hủy bỏ.

Chỉ khi cả ba bước kiểm tra trên đều thành công, chương trình mới tiếp tục quá trình giải mã và lưu file.

# **CHƯƠNG 4: THỬ NGHIỆM VÀ ĐÁNH GIÁ**

## 4.1. Kịch bản Thử nghiệm

Để đảm bảo chương trình được kiểm tra một cách toàn diện, các kịch bản thử nghiệm được chia thành hai nhóm chính: kịch bản hoạt động thành công và kịch bản xử lý lỗi.

### *4.1.1. Kịch bản hoạt động thành công*

* **Mục tiêu:** Xác minh rằng chương trình có thể thực hiện thành công luồng gửi và nhận file an toàn trong điều kiện lý tưởng, đảm bảo dữ liệu sau khi giải mã giống hệt với dữ liệu gốc.
* **Các bước thực hiện:**
  1. Chạy script generate\_keys.py để tạo các cặp khóa RSA cho Người gửi và Người nhận.
  2. Tạo một file email.txt với nội dung văn bản cụ thể.
  3. Khởi chạy receiver.py. Chương trình sẽ bắt đầu lắng nghe trên cổng đã định.
  4. Khởi chạy sender.py. Chương trình sẽ kết nối, thực hiện toàn bộ quy trình mã hóa, ký số, và gửi gói tin.
* **Kết quả mong đợi:**
  1. Cửa sổ terminal của sender.py hiển thị thông báo nhận được ACK từ người nhận.
  2. Cửa sổ terminal của receiver.py hiển thị các bước xác thực thành công và thông báo đã giải mã, lưu file thành công.
  3. Một file mới tên email\_decrypted.txt được tạo ra trong thư mục của receiver.py.
  4. Nội dung của email\_decrypted.txt phải giống hệt 100% so với nội dung của file email.txt ban đầu.

### *4.1.2. Kịch bản xử lý lỗi*

* **Mục tiêu:** Kiểm tra khả năng phòng thủ của hệ thống khi đối mặt với các tình huống dữ liệu không hợp lệ. Chương trình phải có khả năng phát hiện lỗi, từ chối dữ liệu và gửi thông báo

NACK chính xác.

* **Các kịch bản cụ thể:**
  + **a) Kịch bản Lỗi Hết hạn (Timeout Error):**
    - **Thao tác:** Sửa đổi tạm thời trong sender.py, đặt giá trị expiration thành một thời điểm trong quá khứ (ví dụ: thời\_gian\_hiện\_tại - 24 giờ).
    - **Kết quả mong đợi:** receiver.py phải phát hiện gói tin đã hết hạn, từ chối giải mã và in ra thông báo lỗi về thời gian. sender.py phải nhận được phản hồi NACK (timeout).
  + **b) Kịch bản Lỗi Toàn vẹn (Integrity Error):**
    - **Thao tác:** Sửa đổi tạm thời trong sender.py, can thiệp vào ciphertext (ví dụ: thêm một vài byte vào cuối) *sau khi* đã tính hash nhưng *trước khi* gửi gói tin đi.
    - **Kết quả mong đợi:** receiver.py phải phát hiện giá trị hash tính lại không khớp với giá trị hash nhận được. Chương trình từ chối giải mã, in ra lỗi toàn vẹn.

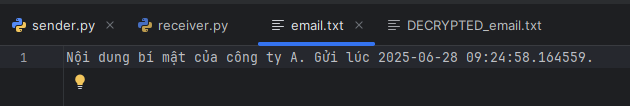
sender.py phải nhận được phản hồi NACK (integrity).

* + **c) Kịch bản Lỗi Xác thực (Authentication Error):**
    - **Thao tác:** Sửa đổi tạm thời trong receiver.py, yêu cầu chương trình sử dụng một khóa công khai sai (ví dụ: dùng chính khóa công khai của người nhận) để xác thực chữ ký của người gửi.
    - **Kết quả mong đợi:** receiver.py phải phát hiện chữ ký không hợp lệ, từ chối gói tin, in ra lỗi xác thực. sender.py phải nhận được phản hồi NACK (signature).

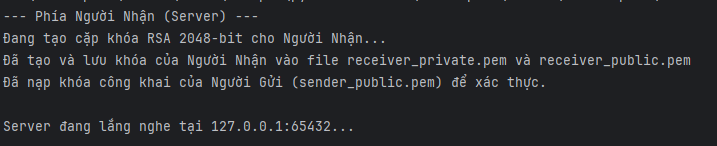
## 4.2. Kết quả Thử nghiệm

### *4.2.1. Kết quả thực thi các kịch bản*

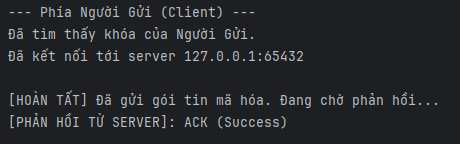
* **Kết quả kịch bản thành công:**



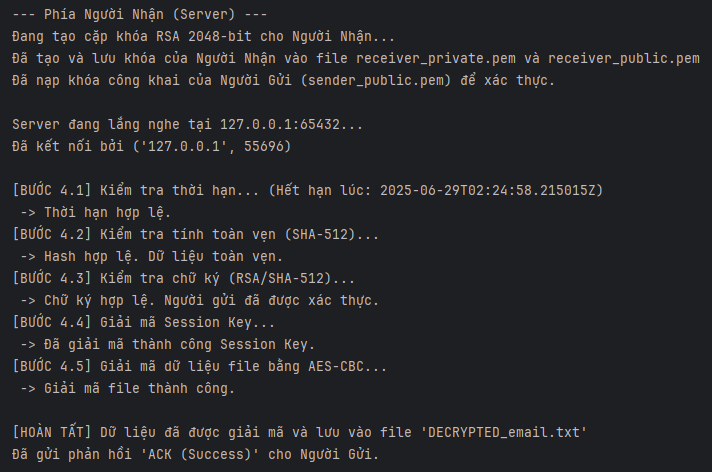
*Hình ảnh 1: Nội dung file email*



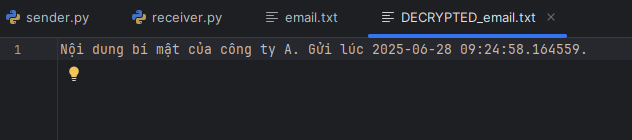
*Hình ảnh 2: Giao diện console của chương trình Người Nhận (Server) khi khởi tạo thành công.*



*Hình ảnh 3: Kết quả thực thi của chương trình Người Gửi (Client) trong kịch bản thành công*

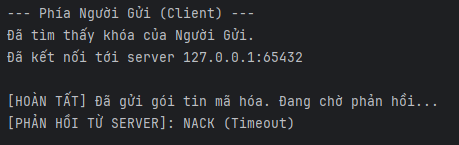


*Hình ảnh 4:Toàn bộ nhật ký xử lý một giao dịch thành công của chương trình Người Nhận.*

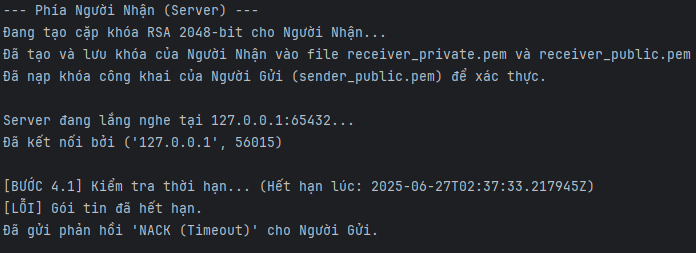


*Hình ảnh 5: Nội dung file DECRYPTED sau khi giải mã thành công*

* **Kết quả kịch bản lỗi hết hạn:**

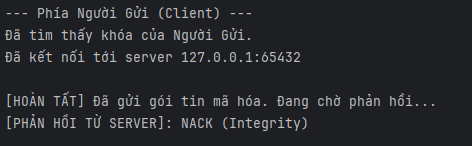


*Hình ảnh 6: Kết quả thực thi của chương trình Người Gửi (Client) trong kịch bản lỗi hết hạn*

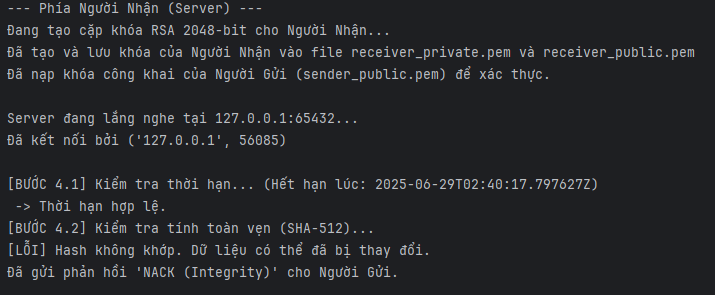


*Hình ảnh 7: Toàn bộ nhật ký xử lý một giao dịch bị lỗi hết hạn của chương trình Người Nhận.*

* **Kết quả kịch bản lỗi toàn vẹn:**

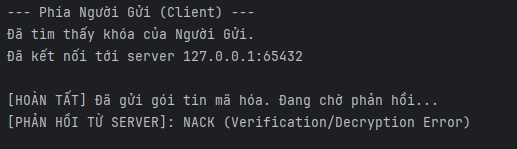


*Hình ảnh 8: Kết quả thực thi của chương trình Người Gửi (Client) trong kịch bản lỗi toàn vẹn*

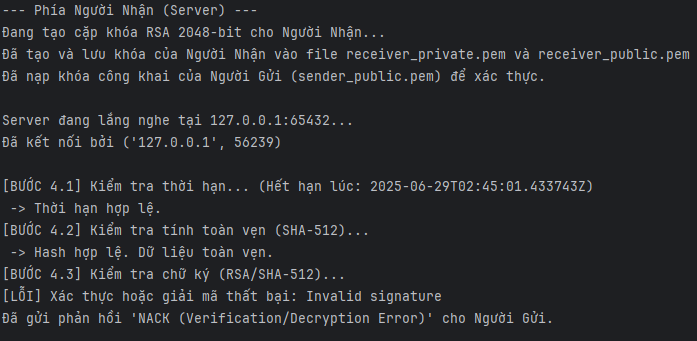


*Hình ảnh 9: Toàn bộ nhật ký xử lý một giao dịch bị lỗi toàn vẹn của chương trình Người Nhận.*

* **Kết quả kịch bản lỗi xác thực:**



*Hình ảnh 10: Kết quả thực thi của chương trình Người Gửi (Client) trong kịch bản lỗi xác thực*



*Hình ảnh 11: Toàn bộ nhật ký xử lý một giao dịch bị lỗi xác thực của chương trình Người Nhận.*

## 4.3. Đánh giá hiệu quả

Dựa trên các kết quả thử nghiệm, hiệu quả của hệ thống được đánh giá trên các phương diện sau:

* **Về tính đúng đắn:** Kịch bản hoạt động thành công đã chứng minh chương trình hoàn thành đúng chức năng cốt lõi. Dữ liệu sau khi qua quá trình mã hóa-truyền-giải mã vẫn bảo toàn được nội dung gốc một cách chính xác.
* **Về khả năng bảo mật:** Các kịch bản xử lý lỗi đã cho thấy hệ thống có khả năng phòng thủ hiệu quả. Chương trình đã phát hiện và ngăn chặn thành công các tấn công giả lập:
  + Từ chối dữ liệu hết hạn.
  + Phát hiện và từ chối dữ liệu đã bị sửa đổi (mất tính toàn vẹn).
  + Phát hiện và từ chối dữ liệu từ nguồn không được xác thực (chữ ký không hợp lệ).
* **Về hiệu năng:** Quá trình mã hóa và giải mã diễn ra nhanh chóng, cho thấy việc áp dụng các lớp bảo mật này không ảnh hưởng quá lớn đến trải nghiệm người dùng đối với các file có kích thước vừa và nhỏ.

# **CHƯƠNG 5. KẾT LUẬN**

Bài tập lớn đã hoàn thành xuất sắc mục tiêu xây dựng một chương trình truyền file an toàn theo yêu cầu của Đề tài 1, áp dụng thành công các cơ chế bảo mật then chốt: mã hóa nội dung bằng AES-CBC để đảm bảo tính bí mật, sử dụng chữ ký số RSA-2048 và hash SHA-512 để xác thực người gửi và đảm bảo tính toàn vẹn, đồng thời thực thi nghiêm ngặt yêu cầu giới hạn truy cập trong 24 giờ. Qua thử nghiệm, chương trình đã chứng tỏ khả năng hoạt động ổn định và xử lý hiệu quả các kịch bản lỗi với cơ chế phản hồi NACK, qua đó xác nhận tính đúng đắn của giải pháp. Hướng phát triển trong tương lai bao gồm việc xây dựng giao diện đồ họa (GUI) để nhận điểm thưởng và mở rộng chương trình để hoạt động trên Internet thông qua các dịch vụ đám mây.

Quá trình thực hiện đề tài không chỉ là một bài tập kỹ thuật mà còn là một hành trình học hỏi sâu sắc, giúp nhóm củng cố vững chắc các kiến thức và kỹ năng chuyên môn. Về mặt kiến thức, nhóm đã có cơ hội vận dụng trực tiếp các thuật toán mật mã đã học như AES, RSA, và SHA vào một ứng dụng thực tiễn, thay vì chỉ dừng lại ở mức độ lý thuyết. Việc tự tay thiết kế và triển khai một giao thức bảo mật hoàn chỉnh, từ bước bắt tay, xác thực danh tính, trao đổi khóa, cho đến mã hóa và kiểm tra toàn vẹn dữ liệu, đã mang lại một cái nhìn toàn diện và thực tế về cách các thành phần này phối hợp với nhau để tạo nên một hệ thống an toàn. Về mặt kỹ năng, đề tài đã giúp rèn luyện tư duy xây dựng một hệ thống có khả năng phòng thủ, luôn phải cân nhắc đến các kịch bản tấn công và các biện pháp đối phó để đảm bảo tính bảo mật và toàn vẹn cho dữ liệu. Đây là một bài thực hành giá trị, khẳng định tầm quan trọng của việc thiết kế và triển khai các giao thức bảo mật trong ứng dụng thực tế.

# **DANH MỤC TÀI LIỆU THAM KHẢO**

[1] Trần Đăng Công, Nguyễn Văn Nhân, & Lê Thị Thùy Trang (2024). Hướng dẫn thực hiện Bài tập lớn môn Nhập môn An toàn, bảo mật thông tin.

[2] Stallings, William (2017). Cryptography and Network Security: Principles and Practice (7th Edition). Pearson.