

**BỘ GIÁO DỤC VÀ ĐÀO TẠO TRƯỜNG ĐẠI HỌC ĐẠI NAM**

BÀI TẬP LỚN

**TÊN HỌC PHẦN: AN TOÀN VÀ BẢO MẬT THÔNG TIN**

**ĐỀ TÀI: GỬI BÁO CÁO CÔNG TY QUA SERVER TRUNG GIAN**

**G.V hướng dẫn: T.S Trần Đăng Công**

**Nhóm: Nhóm 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Mã Sinh Viên** | **Họ và Tên** | **Ngày Sinh** | **Lớp** |
| **1** | **1671020083** | **Nguyễn Tiến Đạt** | **10/05/2004** | **CNTT 16-04** |
| **2** | **1671020118** | **Lê Bá Hoan** | **18/12/2003** | **CNTT 16-04** |
| **3** | **1671020150** | **Đào Gia Hưng** | **05/05/2004** | **CNTT 16-04** |



**BỘ GIÁO DỤC VÀ ĐÀO TẠO TRƯỜNG ĐẠI HỌC ĐẠI NAM**

BÀI TẬP LỚN

**TÊN HỌC PHẦN: AN TOÀN VÀ BẢO MẬT THÔNG TIN**

**ĐỀ TÀI: GỬI BÁO CÁO CÔNG TY QUA SERVER TRUNG GIAN**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Mã Sinh Viên** | **Họ và Tên** | **Ngày Sinh** | **Điểm** | |
| **Bằng Số** | **Bằng Chữ** |
| **1** | **1671020083** | **Nguyễn Tiến Đạt** | **10/05/2004** |  |  |
| **2** | **1671020120** | **Lê Bá Hoan** | **18/12/2003** |  |  |
| **2** | **1671020150** | **Đào Gia Hưng** | **05/05/2004** |  |  |

**CÁN BỘ CHẤM THI 1 CÁN BỘ CHẤM THI 2**

# LỜI NÓI ĐẦU

Trong thời đại công nghệ thông tin bùng nổ, việc trao đổi dữ liệu giữa các bộ phận trong công ty, cũng như giữa doanh nghiệp với đối tác, khách hàng hay cơ quan quản lý trở nên thường xuyên và không thể thiếu. Đặc biệt, việc gửi các báo cáo nội bộ, báo cáo tài chính, hoặc các dữ liệu nhạy cảm yêu cầu phải có phương thức truyền tải đảm bảo tính bảo mật, toàn vẹn và độ tin cậy cao. Tuy nhiên, sự gia tăng của các mối đe dọa mạng và rủi ro mất mát dữ liệu khiến quá trình gửi nhận báo cáo đòi hỏi phải có giải pháp trung gian an toàn và hiệu quả hơn.

Đề tài **"Gửi báo cáo công ty qua Server trung gian"** được thực hiện nhằm nghiên cứu và xây dựng một mô hình truyền tải dữ liệu thông qua một máy chủ trung gian có tích hợp các cơ chế bảo mật. Phương án sử dụng server trung gian cho phép phân lập và kiểm soát chặt chẽ việc gửi nhận báo cáo, giúp hạn chế truy cập trái phép, ngăn chặn rò rỉ dữ liệu và phát hiện các hành vi bất thường. Bên cạnh đó, việc tích hợp các kỹ thuật mã hóa dữ liệu, xác thực người dùng, và kiểm tra tính toàn vẹn sẽ bảo đảm rằng báo cáo được gửi đến đúng nơi, đúng người, không bị can thiệp trong suốt quá trình truyền tải.

Đề tài là sự kết hợp giữa lý thuyết đã học trong môn An toàn và bảo mật thông tin với thực tiễn ứng dụng trong môi trường doanh nghiệp. Thông qua việc thiết kế, xây dựng và thử nghiệm hệ thống gửi báo cáo qua server trung gian, sinh viên sẽ có cơ hội áp dụng các kiến thức về mã hóa đối xứng, bất đối xứng, hàm băm, giao thức xác thực và quản lý khóa. Qua đó, đề tài không chỉ góp phần củng cố kiến thức chuyên ngành mà còn giúp rèn luyện kỹ năng tư duy hệ thống, khả năng phân tích rủi ro và phát triển giải pháp bảo mật thực tế trong bối cảnh doanh nghiệp ngày càng số hóa..

**MỤC LỤC**

[LỜI NÓI ĐẦU 3](#_Toc202650600)

[CHƯƠNG 1: GIỚI THIỆU 6](#_Toc202650601)

[1. Tổng quan môn học. 6](#_Toc202650602)

[1.1 Các thuật toán được sử dụng 6](#_Toc202650603)

[1.2. Cách hoạt động 8](#_Toc202650604)

[1.2.1. Cách hoạt động 10](#_Toc202650605)

[1.3.Đặt vấn đề. 12](#_Toc202650606)

[*1.3.1* Chức năng 12](#_Toc202650607)

[*1.3.2 Yêu cầu bảo mật* : 12](#_Toc202650608)

[*1.3.3 Luồng xử lý* : 12](#_Toc202650609)

[CHƯƠNG 2. XÂY DỰNG CHƯƠNG TRÌNH 14](#_Toc202650610)

[2.2.1 Sender (Người gửi). 15](#_Toc202650611)

[2.2.2 Receiver (Người nhận). 17](#_Toc202650612)

[2.2.3 Server. 19](#_Toc202650613)

[2.2.4 .Share. 21](#_Toc202650614)

[CHƯƠNG 3. KẾT QUẢ CHƯƠNG TRÌNH 24](#_Toc202650615)

[3.1 Kết quả 24](#_Toc202650616)

[3.2 Người gửi. 28](#_Toc202650617)

[3.3 Server. 28](#_Toc202650618)

[3.4 Người nhận. 29](#_Toc202650619)

[KẾT LUẬN 30](#_Toc202650620)

[DANH MỤC TÀI LIỆU THAM KHẢO 31](#_Toc202650621)

# CHƯƠNG 1: GIỚI THIỆU

## Tổng quan môn học.



Trong bối cảnh công nghệ số phát triển mạnh mẽ, thông tin đã trở thành một tài sản vô cùng quan trọng đối với cá nhân, tổ chức và doanh nghiệp. Tuy nhiên, đi kèm với sự phát triển đó là những mối đe dọa ngày càng tinh vi về mất mát, rò rỉ hoặc bị tấn công dữ liệu. Do đó, việc bảo vệ thông tin trở thành một yêu cầu tất yếu và cấp bách trong mọi lĩnh vực.

* Để nắm rõ vấn đề chủ thể phải có những kiến thức cơ bản và cốt lõi về:
* Các nguyên lý bảo mật thông tin (bảo mật, toàn vẹn, xác thực, chống chối bỏ…),
* Các thuật toán mật mã hiện đại: AES, RSA, SHA,..
* Kỹ thuật phát hiện và phòng chống tấn công.
* Giao thức bảo mật trong truyền thông mạng.
* Triển khai bảo mật trong các hệ thống thực tế.

## Các thuật toán được sử dụng

Các thuật toán trong An toàn bảo mật thông tin :

* **RSA**
* **AES**
* **SHA**

RSA là một thuật toán mã hóa khóa công khai, được sử dụng rộng rãi trong nhiều ứng dụng bảo mật, như chữ ký số, xác thực, mã hóa đường truyền, v.v. RSA được đặt theo tên của ba nhà khoa học đã phát minh ra nó: Ron Rivest, Adi Shamir và Leonard Adleman



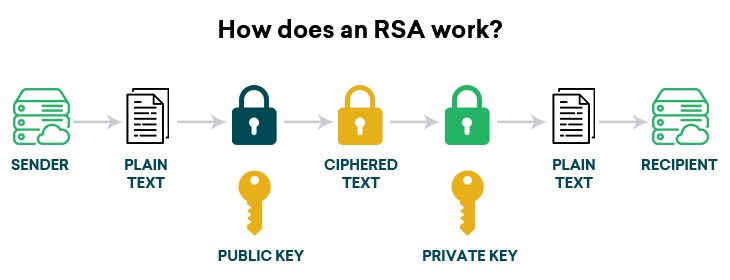
**Tổng quan :**

* + - * RSA là thuật toán mã hóa bất đối xứng, sử dụng một cặp khóa công khai và khóa bí mật.
      * Dựa trên nguyên lý khó phân tích thừa số của số nguyên lớn.
      * Dùng phổ biến trong mã hóa dữ liệu, tạo chữ ký số và xác thực.

Ứng dụng :

* + - * Gửi/nhận dữ liệu an toàn qua Internet (thường dùng để trao đổi khóa AES).
      * Chữ ký số trong tài liệu, email, giao dịch điện tử.
      * Xác thực người dùng trong các hệ thống bảo mật cao.
      * Nền tảng của giao thức HTTPS/SSL trong trình duyệt web
  1. **Cách hoạt động**

Hệ thống mật mã hóa khóa công khai, hay còn gọi là mật mã bất đối xứng, hoạt động dựa trên cặp khóa độc đáo: một khóa công khai (public key) và một khóa riêng tư (private key). Trong quá trình này, khi người gửi (Sender) muốn truyền một thông điệp an toàn, họ sẽ bắt đầu với văn bản gốc (Plain Text), tức là nội dung thông tin chưa được mã hóa. Để đảm bảo tính bảo mật, người gửi sẽ sử dụng khóa công khai của người nhận (Recipient's Public Key) để mã hóa văn bản gốc này. Khóa công khai, như tên gọi của nó, có thể được chia sẻ rộng rãi mà không ảnh hưởng đến an ninh.



Sau khi được mã hóa bằng khóa công khai, văn bản gốc sẽ biến thành văn bản mã hóa (Ciphered Text). Lúc này, thông điệp đã trở nên vô nghĩa đối với bất kỳ ai không có khóa phù hợp, đảm bảo rằng ngay cả khi bị chặn trên đường truyền, nội dung vẫn được bảo vệ. Để truy cập lại thông tin ban đầu, người nhận (Recipient) sẽ sử dụng khóa riêng tư (Private Key) của mình. Khóa riêng tư này là duy nhất và được giữ bí mật tuyệt đối bởi người nhận. Chỉ có khóa riêng tư tương ứng với khóa công khai đã dùng để mã hóa mới có thể giải mã được văn bản mã hóa, biến nó trở lại thành văn bản gốc. Nhờ cơ chế này, mật mã hóa khóa công khai cho phép truyền thông tin an toàn giữa các bên mà không yêu cầu việc trao đổi một khóa bí mật chung trước đó, là nền tảng cho nhiều ứng dụng bảo mật quan trọng trong thế giới kỹ thuật số.

**AES.**

AES (Advanced Encryption Standard) là một thuật toán mã hóa đối xứng được sử dụng để bảo vệ dữ liệu trong các hệ thống máy tính và mạng. AES là một chuẩn mã hóa được công bố bởi Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (NIST) vào năm 2001, thay thế cho chuẩn mã hóa DES (Data Encryption Standard) cũ.

Tổng quan :

* + - * AES là thuật toán mã hóa đối xứng, nghĩa là dùng cùng một khóa để mã hóa và giải

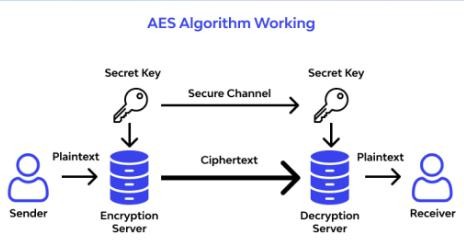
mã.

* + - * Là chuẩn mã hóa được chính phủ Hoa Kỳ công nhận và sử dụng rộng rãi.
      * Hỗ trợ độ dài khóa: 128, 192, hoặc 256 bit.
      * Mã hóa dữ liệu theo từng khối 128 bit, qua nhiều vòng xử lý (10, 12 hoặc 14 vòng)

Ứng dụng:

* + - * Bảo vệ dữ liệu trong các phần mềm, ứng dụng di động.
      * Mã hóa ổ cứng, file, tài liệu cá nhân.
      * Giao tiếp bảo mật: VPN, HTTPS, Wi-Fi (WPA2/WPA3).
      * Hệ thống ngân hàng, tài chính yêu cầu mã hóa tốc độ cao và mạnh mẽ.
    1. **Cách hoạt động**

Thuật toán AES (Advanced Encryption Standard) là một dạng mật mã hóa đối xứng, nghĩa là quá trình mã hóa và giải mã đều sử dụng cùng một khóa bí mật (Secret Key). Để thiết lập một kênh truyền thông an toàn, trước tiên, người gửi (Sender) và người nhận (Receiver) phải thống nhất và chia sẻ một khóa bí mật chung này thông qua một kênh an toàn (Secure Channel). Việc truyền tải khóa này qua một kênh không bị giám sát là cực kỳ quan trọng, vì sự lộ lọt của khóa sẽ làm mất đi tính bảo mật của toàn bộ hệ thống.



Khi người gửi muốn truyền tải văn bản gốc (Plaintext), họ sẽ đưa thông tin này đến một máy chủ mã hóa (Encryption Server). Tại đây, văn bản gốc sẽ được mã hóa bằng chính khóa bí mật đã được thống nhất. Kết quả của quá trình này là văn bản mã hóa (Ciphertext) – một chuỗi dữ liệu không thể đọc hiểu được. Văn bản mã hóa này sau đó được truyền đi qua các kênh thông thường, không cần phải là kênh an toàn, vì nội dung đã được bảo vệ. Khi văn bản mã hóa đến được phía người nhận, nó sẽ được chuyển đến một máy chủ giải mã (Decryption Server). Tại đây, máy chủ sẽ sử dụng chính khóa bí mật đó để giải mã văn bản mã hóa, khôi phục lại văn bản gốc (Plaintext). Lúc này, người nhận có thể đọc hiểu thông tin ban đầu. Tóm lại, hiệu quả của AES và các thuật toán đối xứng khác phụ thuộc hoàn toàn vào việc bảo mật và quản lý chặt chẽ khóa bí mật.

**SHA**

SHA-256 là một hàm băm mật mã, một loại thuật toán được sử dụng để chuyển đổi dữ liệu thành một chuỗi có độ dài cố định, trong trường hợp này là 256 bit (32 byte). Nó được sử dụng để bảo mật dữ liệu và xác minh tính toàn vẹn của thông tin, đảm bảo rằng dữ liệu không bị thay đổi.



Tổng quan:

SHA-256 là thuật toán băm (hash function), không phải mã hóa.

Nó biến đổi dữ liệu đầu vào thành chuỗi băm 256 bit, không thể đảo ngược.

Không dùng để mã hóa/giải mã, mà để kiểm tra tính toàn vẹn dữ liệu.

**Ứng dụng**:

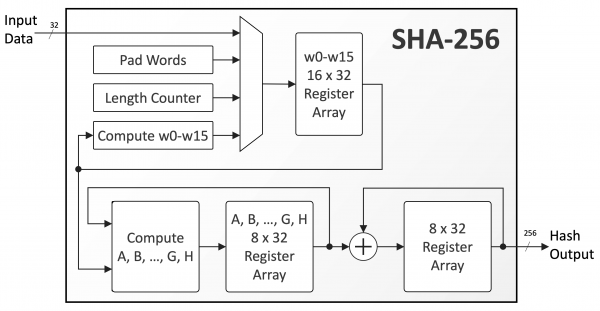
Kiểm tra toàn vẹn file/tài liệu (tạo mã băm và so sánh).

Xác thực mật khẩu (mật khẩu được lưu dưới dạng hash).

Ứng dụng trong chữ ký số, blockchain (như Bitcoin).

Sử dụng trong giao thức bảo mật như TLS/SSL.

Cách hoạt động :



Thuật toán SHA-256 hoạt động bằng cách nhận dữ liệu đầu vào (Input Data), sau đó chuẩn bị dữ liệu bằng cách đệm và tính toán các từ khởi tạo (w0-w15), lưu chúng vào một mảng thanh ghi. Tiếp theo, thuật toán thực hiện một loạt các phép tính phức tạp trên các thanh ghi trạng thái (A, B, ..., H), cập nhật chúng sau mỗi vòng lặp. Cuối cùng, các giá trị trạng thái này được cộng dồn vào các giá trị băm tích lũy và xuất ra chuỗi băm (Hash Output) 256 bit duy nhất. Quá trình này đảm bảo rằng mọi thay đổi nhỏ trong dữ liệu đầu vào đều tạo ra một chuỗi băm hoàn toàn khác biệt, phục vụ mục đích kiểm tra tính toàn vẹn của dữ liệu.

## 1.3.Đặt vấn đề.

Trong môi trường doanh nghiệp hiện đại, nơi việc gửi các báo cáo nội bộ hoặc báo cáo tài chính diễn ra thường xuyên qua mạng, đặc biệt trong điều kiện băng thông hạn chế và nguy cơ tấn công mạng gia tăng, yêu cầu về tính bảo mật, toàn vẹn và xác thực của dữ liệu ngày càng trở nên cấp thiết. Khi một nhân viên gửi file báo cáo như report.docx đến máy chủ quản lý thông qua một server trung gian, hệ thống không chỉ cần đảm bảo việc chia nhỏ dữ liệu để tối ưu đường truyền, mà còn phải ngăn chặn nguy cơ bị rò rỉ, chỉnh sửa hoặc giả mạo trong suốt quá trình truyền. Đề tài này tập trung xây dựng một giải pháp truyền báo cáo an toàn, kết hợp giữa mã hóa đối xứng (DES) để tăng tốc độ mã hóa dữ liệu, mã hóa bất đối xứng và ký số bằng RSA 1024-bit theo chuẩn PKCS#1 v1.5 sử dụng hàm băm SHA-512, cùng với cơ chế kiểm tra tính toàn vẹn dữ liệu qua SHA-512. Hệ thống mô phỏng đầy đủ quy trình handshake, xác thực người gửi, trao đổi khóa, mã hóa và kiểm tra chữ ký nhằm đảm bảo báo cáo được gửi đến đúng đích, không bị thay đổi và chỉ người nhận hợp lệ mới có thể giải mã nội dung.

* + 1. **Chức năng**
       - Gửi file báo cáo report.txt từ công ty đến đối tác thông qua server trung gian.
       - Lưu log thời gian giao dịch tại server trung gian (thời điểm gửi, nhận, xác nhận).
       - Không cho server trung gian tham gia vào xử lý bảo mật (chỉ chuyển tiếp và ghi log).
       - Hệ thống mô phỏng quá trình gửi – nhận – xác thực với mã hóa, ký số và kiểm tra toàn vẹn.

## *1.3.2 Yêu cầu bảo mật* :

Mã hóa nội dung: Sử dụng AES-GCM (Authenticated Encryption).

Ký số và trao khóa: Dùng RSA 1024-bit (OAEP) + SHA-512.

Kiểm tra tính toàn vẹn: Dùng SHA-512, hash chuỗi nonce || ciphertext || tag.

Ký metadata: Gồm tên file, timestamp, ID giao dịch → ký bằng RSA/SHA-512.

Tất cả thông tin trao đổi đều được **bảo mật đầu-cuối** (end-to-end encryption).

## *1.3.3 Luồng xử lý* :

* + - * Handshake giữa người gửi và người nhận (qua server trung gian).
      * Ký số metadata, trao đổi khóa session bằng RSA.
      * Mã hóa file bằng AES-GCM, tạo nonce và tính hash.
      * Gửi gói tin chứa ciphertext, tag, hash và chữ ký số.
      * Người nhận kiểm tra hash và chữ ký**,** nếu hợp lệ thì giải mã và lưu file; nếu lỗi thì gửi phản hồi từ chối.
      * Server trung gian chỉ chuyển tiếp dữ liệu và lưu log thời gian giao dịch

# CHƯƠNG 2. XÂY DỰNG CHƯƠNG TRÌNH

## 2.1.Đặc tả

Mô tả bài toán

Đề tài xây dựng một hệ thống truyền file báo cáo (report.txt) từ công ty đến đối tác thông qua server trung gian, với yêu cầu đảm bảo an toàn, toàn vẹn và xác thực dữ liệu. Server trung gian chỉ chuyển tiếp và lưu log thời gian, không tham gia vào xử lý bảo mật. Quá trình truyền file bao gồm handshake, trao khóa, mã hóa bằng AES-GCM và ký số bằng RSA nhằm đảm bảo dữ liệu đến đúng đích, không bị thay đổi hay giả mạo trong quá trình gửi qua mạng.

Mục tiêu:

* + - * Gửi file báo cáo từ người gửi đến người nhận qua một server trung gian mà vẫn đảm bảo bảo mật đầu cuối.
      * Lưu log thời gian gửi và nhận tại server trung gian để phục vụ kiểm tra và đối soát.
      * Ứng dụng các kĩ thuật mã hóa, xác thực và kiểm tra toàn vẹn vào quy trình truyền file.

Yêu cầu bảo mật:

* + - * Mã hóa: Sử dụng thuật toán AES-GCM để mã hóa nội dung file, đảm bảo tính bảo mật và toàn vẹn dữ liệu nhờ tag xác thực.
      * Trao khóa & ký số: SessionKey được mã hóa bằng RSA 1024-bit (OAEP), metadata (tên file, thời gian, ID giao dịch) được ký số bằng RSA + SHA-512 để xác thực người gửi.
      * Kiểm tra toàn vẹn: Sử dụng SHA-512 để tính hash của chuỗi nonce || ciphertext || tag, giúp phát hiện nếu dữ liệu bị thay đổi trong quá trình truyền.

Luồng xử lý:

* + - * Handshake giữa người gửi và người nhận (qua server trung gian).
      * Ký số metadata, trao đổi khóa session bằng RSA.
      * Mã hóa file bằng AES-GCM, tạo nonce và tính hash.
      * Gửi gói tin chứa ciphertext, tag, hash và chữ ký số.
      * Người nhận kiểm tra hash và chữ ký; nếu hợp lệ thì giải mã và lưu file, nếu lỗi thì gửi phản hồi từ chối.
      * Server trung gian ghi log thời gian của các bước gửi, nhận, xác nhận.

## 2.1.1.Yêu cầu

**Sender:**

* + - * Ký metadata (tên file, thời gian, ID giao dịch) bằng RSA/SHA-512.
      * Mã hóa SessionKey bằng RSA 1024-bit (OAEP).
      * Mã hóa file report.txt bằng AES-GCM và tạo tag, hash.
      * Gửi gói tin gồm nonce, ciphertext, tag, hash và chữ ký số qua server trung gian.
      * Nhận phản hồi ACK hoặc NACK từ người nhận thông qua server trung gian.

**Receiver:**

* + - * Nhận gói tin từ server trung gian.
      * Kiểm tra chữ ký RSA, hash và tag để xác minh tính toàn vẹn và xác thực.
      * Nếu hợp lệ, giải mã file bằng AES-GCM và lưu report.txt.
      * Gửi ACK về cho người gửi. Nếu không hợp lệ, gửi NACK (thông báo lỗi integrity).

## 2.2Xây dựng.

## 2.2.1 Sender (Người gửi).

Vai trò: Là nơi người dùng chọn và gửi file report.txt đến người nhận thông qua server trung gian.

Chức năng chính:

Tạo giao diện web bằng Flask để chọn file.

Kiểm tra định dạng, kích thước file.

Thực hiện handshake với server trung gian.

Mã hóa file bằng AES-GCM, sinh nonce và tag.

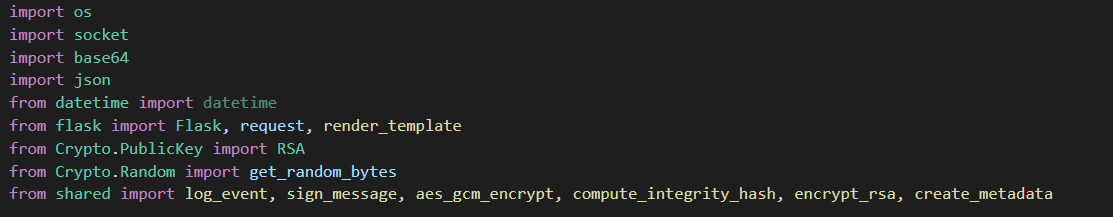
Tính hash SHA-512(nonce + cipher + tag).

Ký số hash và metadata bằng RSA/SHA-512.

Mã hóa session key AES bằng RSA-OAEP.

Gửi gói JSON chứa dữ liệu mã hóa, chữ ký, metadata đến server trung gian.

**Import các thư viện cần thiết.**



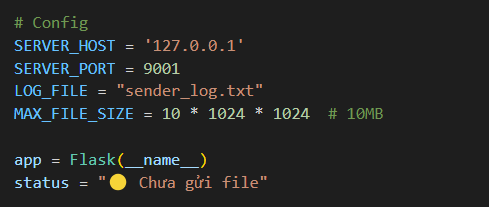
os, socket, json, base64: quản lý hệ thống, giao tiếp mạng, mã hóa dữ liệu.

flask: tạo giao diện web.

Crypto: các hàm bảo mật.

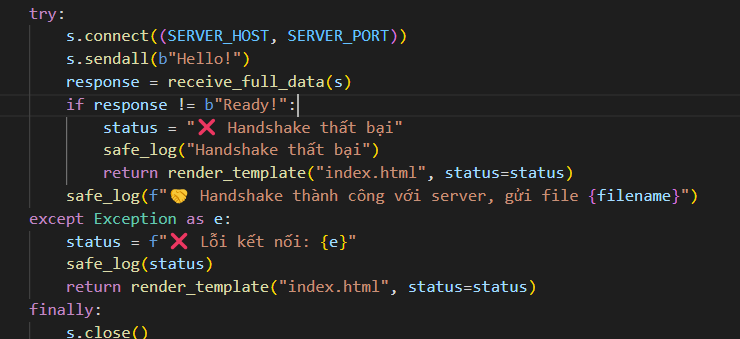
shared: chứa các hàm do người viết tự định nghĩa (mã hóa, ký số, log…).

**Khởi tạo ứng dụng và cấu hình.**



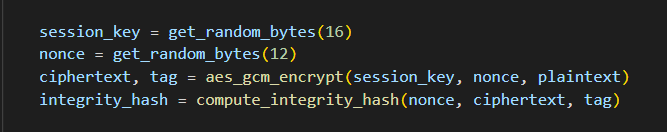
Tạo ứng dụng web Flask để người dùng gửi file.  
Cấu hình địa chỉ và cổng để kết nối tới server trung gian.

**Handshake ban đầu với server.**

****

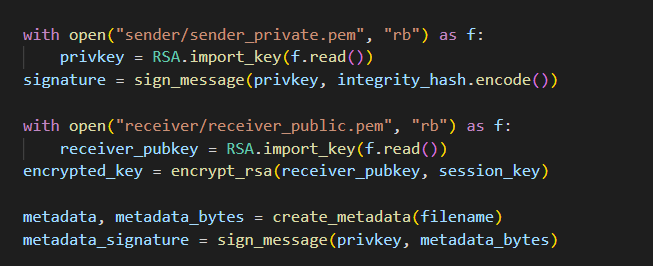
Người gửi kiểm tra xem phía người nhận có sẵn sàng nhận file không bằng cách gửi "Hello!".  
Nếu nhận "Ready!" thì tiếp tục gửi dữ liệu.

**Mã hóa nội dung file bằng AES-GCM.**

****

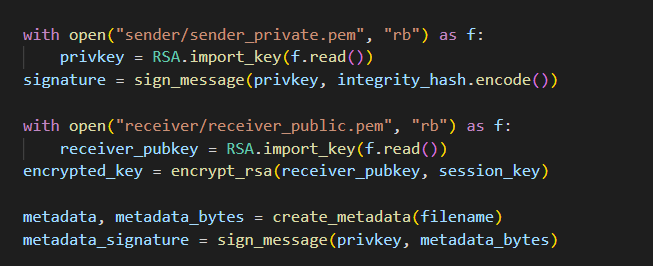
Mã hóa nội dung file bằng AES-GCM với nonce ngẫu nhiên.  
Tạo tag dùng để xác thực khi giải mã.

**Ký số hash và metadata.**

****

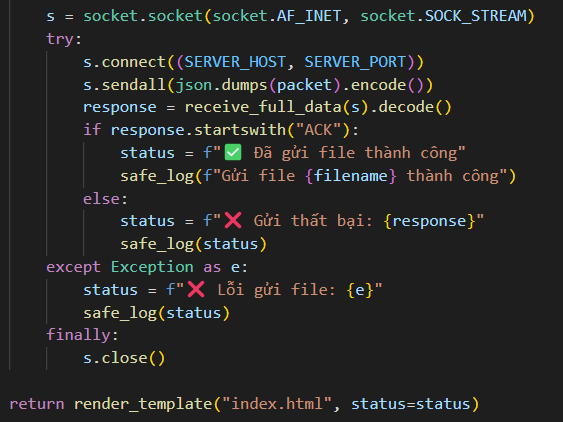
Ký số hash để xác thực file không bị thay đổi.  
Ký metadata (gồm: tên file, thời gian, ID) để chứng minh nguồn gốc gửi.

**Mã hóa session key bằng RSA.**

****

Đảm bảo khóa AES được gửi an toàn cho người nhận, thông qua mã hóa RSA-OAEP.

**Nhận ACK hoặc NACK.**

****

Xác nhận người nhận đã giải mã và xác minh thành công hay thất bại.

## 2.2.2 Receiver (Người nhận).

Vai trò: Nhận và xác minh file từ người gửi, đảm bảo toàn vẹn và xác thực nguồn gốc.

Chức năng chính:

Lắng nghe dữ liệu từ server trung gian.

Trả lời "Ready!" khi nhận "Hello!" để xác nhận kết nối.

Giải mã gói tin JSON, kiểm tra:

Chữ ký metadata.

Hash toàn vẹn.

Chữ ký số.

Giải mã khóa AES bằng RSA.

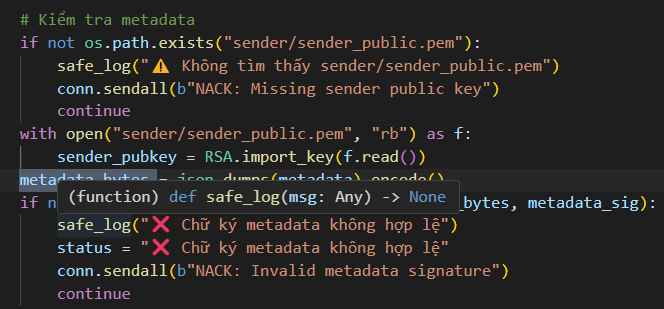
Giải mã nội dung file bằng AES-GCM.

Lưu file thành report.txt.

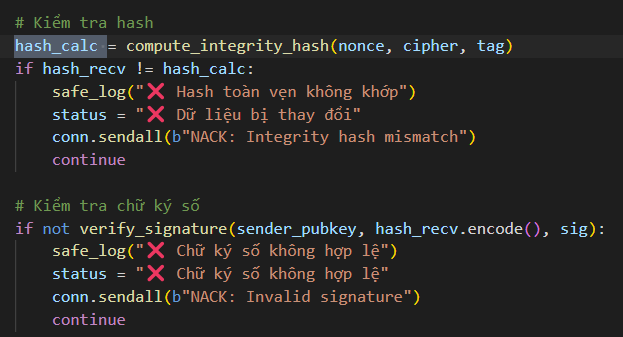
Gửi phản hồi "ACK" nếu thành công, "NACK" nếu thất bại.

Về thư viện và các khởi tạo về cấu hình giống với người gửi.

**Xác minh metadata.**

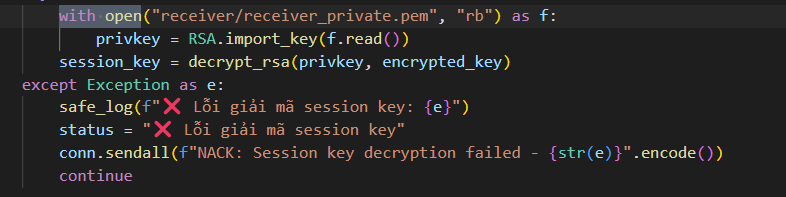


Kiểm tra chữ ký metadata để chắc chắn file thực sự đến từ người gửi hợp pháp.**Kiểm tra toàn vẹn và chữ ký số.**

****

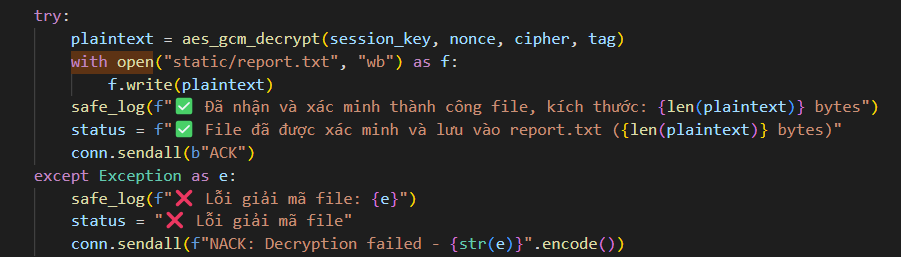
So sánh hash để kiểm tra dữ liệu có bị thay đổi không.  
Kiểm tra chữ ký hash để xác minh dữ liệu do đúng người gửi ký.

**Giải mã khóa phiên bằng RSA.**

****

Giải mã session key (khóa AES) bằng private key của người nhận.

**Giải mã file bằng AES-GCM.**

****

Sử dụng AES-GCM để giải mã nội dung file.  
Nếu thành công, file sẽ được lưu lại tại static/report.txt.

## 2.2.3 Server.

Vai trò: Là cầu nối giữa người gửi và người nhận, không xử lý bảo mật, chỉ chuyển tiếp và ghi log.

Chức năng chính:

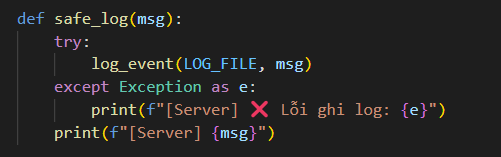
Lắng nghe kết nối từ Sender (port 9001).

Nhận dữ liệu và chuyển tiếp nguyên vẹn cho Receiver (port 9003).

Nhận phản hồi (ACK/NACK) từ Receiver và chuyển lại cho Sender.

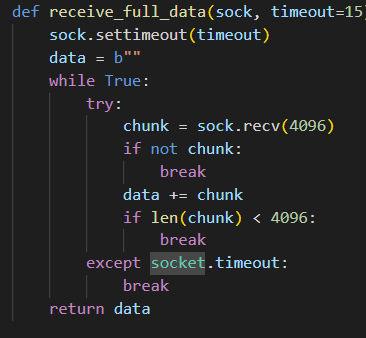
Ghi log thời gian nhận/gửi dữ liệu vào server\_log.txt.

**Ghi log an toàn.**

****

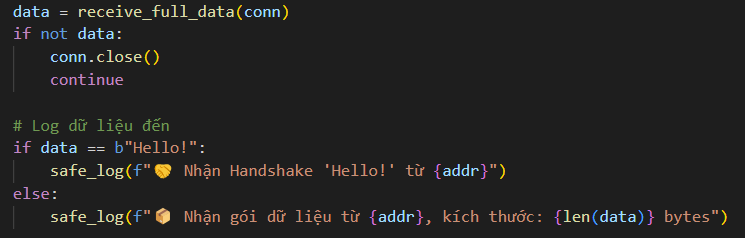
Ghi lại các hành động quan trọng (như nhận/gửi gói tin, ACK/NACK) vào file server\_log.txt — phục vụ kiểm tra và xác minh thời gian giao dịch (yêu cầu của đề tài).

**Nhận đầy đủ dữ liệu từ socket.**

****

Hàm này đảm bảo nhận hết toàn bộ dữ liệu từ người gửi hoặc từ người nhận (không bị cắt giữa chừng do giới hạn gói tin).

**Nhận dữ liệu từ Sender.**

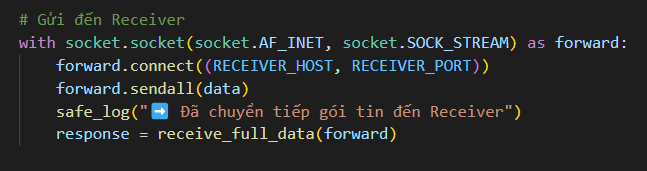
****

Nhận gói tin từ người gửi:

Nếu là "Hello!" thì đó là Handshake ban đầu.

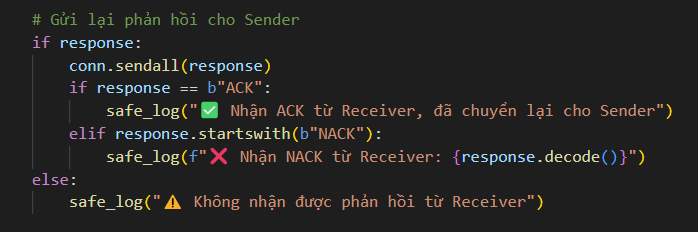
Nếu là dữ liệu JSON → chuyển tiếp sang người nhận.

**Chuyển tiếp gói tin sang Receiver.**

****

Server trung gian không xử lý nội dung, chỉ đóng vai trò "trung chuyển dữ liệu".

**Nhận phản hồi từ Receiver và chuyển ngược lại.**

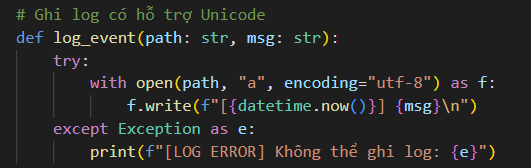
****

Phản hồi "ACK" hoặc "NACK" từ phía Receiver sẽ được chuyển ngược lại cho Sender.

## 2.2.4 .Share.

Vai trò: Cung cấp các hàm xử lý bảo mật cho toàn bộ hệ thống.

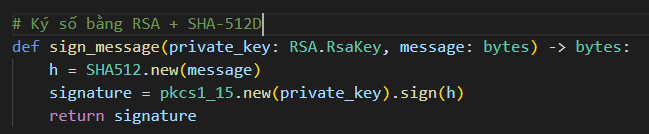
**Hàm log\_event() – Ghi log có hỗ trợ Unicode.**

****

Ghi thông điệp log vào file văn bản, kèm theo thời gian, dùng trong:

* sender, receiver, server
* Phục vụ yêu cầu “lưu log thời gian giao dịch tại server trung gian” trong đề bài.

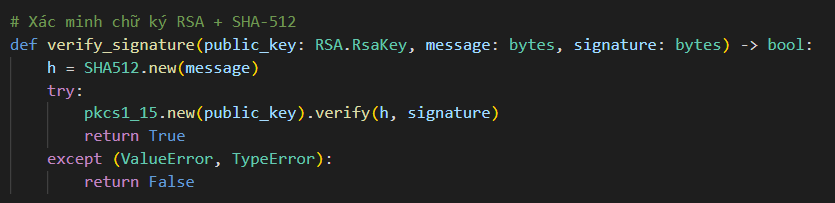
**Hàm sign\_message() – Ký số bằng RSA + SHA-512.**

****

Ký một chuỗi dữ liệu bằng **RSA và hàm băm SHA-512** (dùng trong: ký hash, ký metadata), đảm bảo:

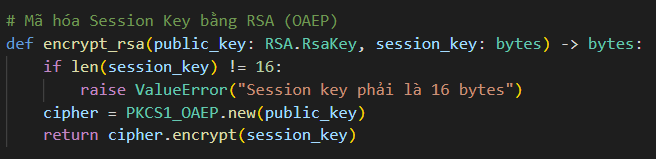
* Xác thực nguồn gốc (chữ ký không thể giả mạo nếu không có khóa riêng).
* Không thể thay đổi nội dung sau khi ký

**Hàm verify\_signature() – Xác minh chữ ký.**

****

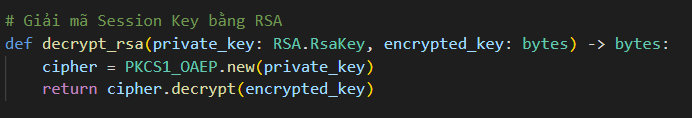
Kiểm tra tính hợp lệ của chữ ký số bằng khóa công khai của người gửi.  
Nếu sai → có thể do dữ liệu bị thay đổi, hoặc không đúng người ký.

**Hàm encrypt\_rsa() – Mã hóa session key bằng RSA (OAEP).**



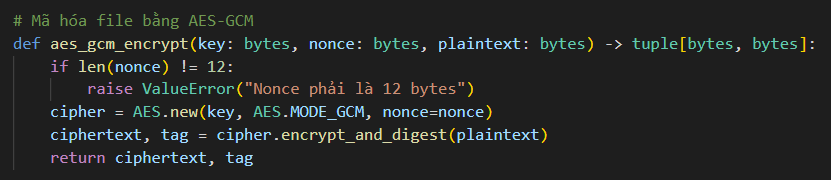
Mã hóa khóa AES (session\_key) bằng **RSA với padding OAEP**, đảm bảo chỉ có người nhận (có khóa riêng) mới giải mã được khóa AES.

**Hàm decrypt\_rsa() – Giải mã session key.**



Giải mã khóa AES (session key) từ dữ liệu đã mã hóa RSA.

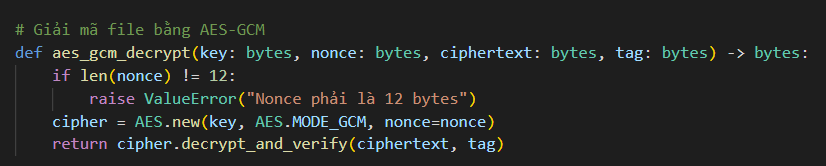
**Hàm aes\_gcm\_encrypt() – Mã hóa nội dung file.**



Mã hóa file bằng AES-GCM:

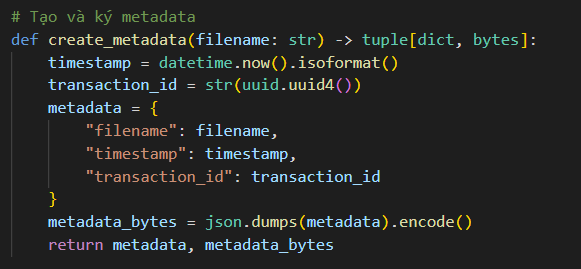
* Giao thức mã hóa đối xứng mạnh và nhanh.
* Trả về ciphertext và tag (dùng xác thực tính toàn vẹn).

**Hàm aes\_gcm\_decrypt() – Giải mã nội dung file.**

****

Giải mã file bằng AES-GCM. Nếu tag không khớp → giải mã thất bại → dữ liệu bị thay đổi.

**Hàm create\_metadata() – Tạo và ký metadata.**

****

Tạo metadata gồm:

* filename: tên file
* timestamp: thời gian gửi
* transaction\_id: mã giao dịch duy nhất (UUID)  
  → Sau đó trả về phiên bản metadata và metadata\_bytes để ký.

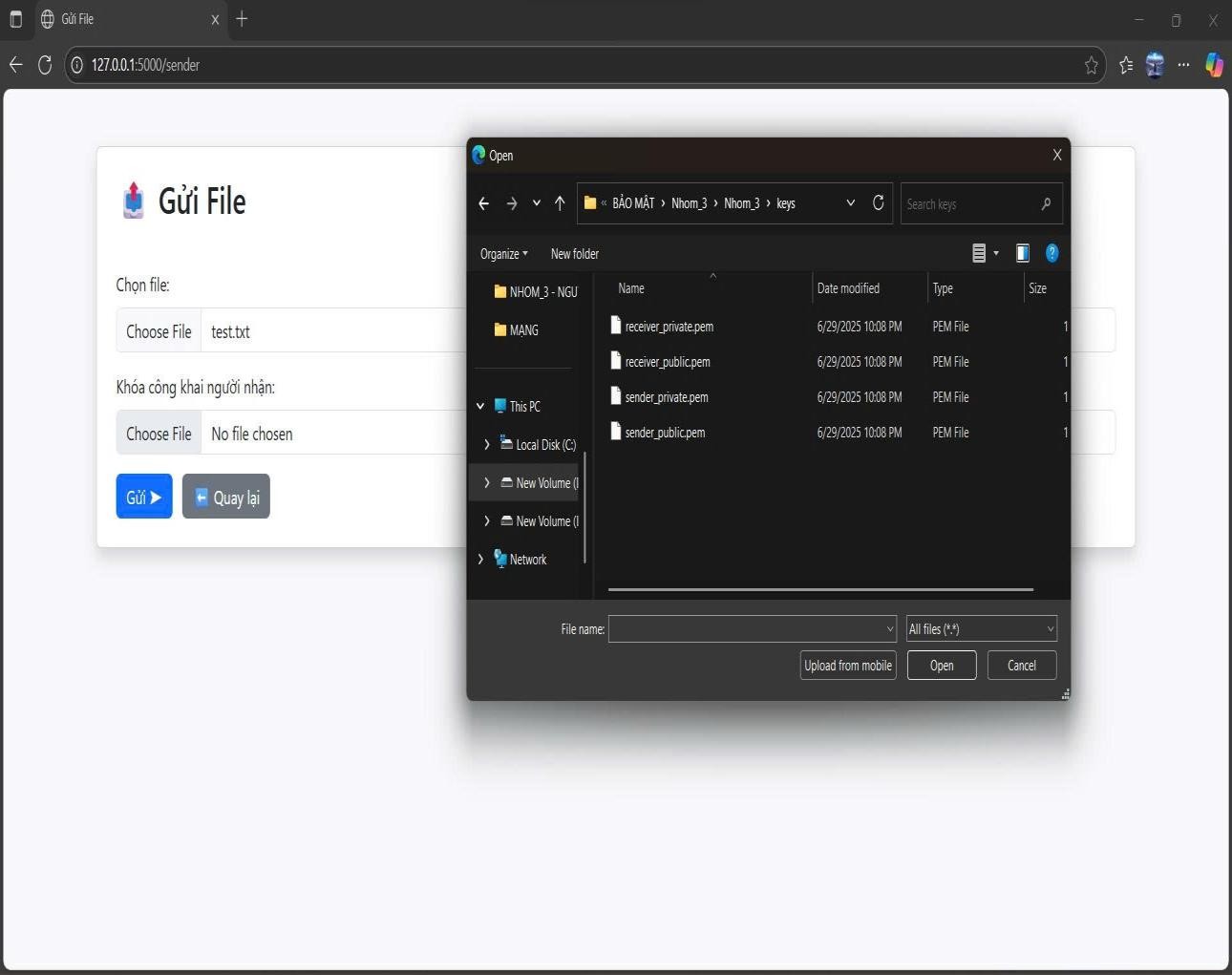
# CHƯƠNG 3. KẾT QUẢ CHƯƠNG TRÌNH

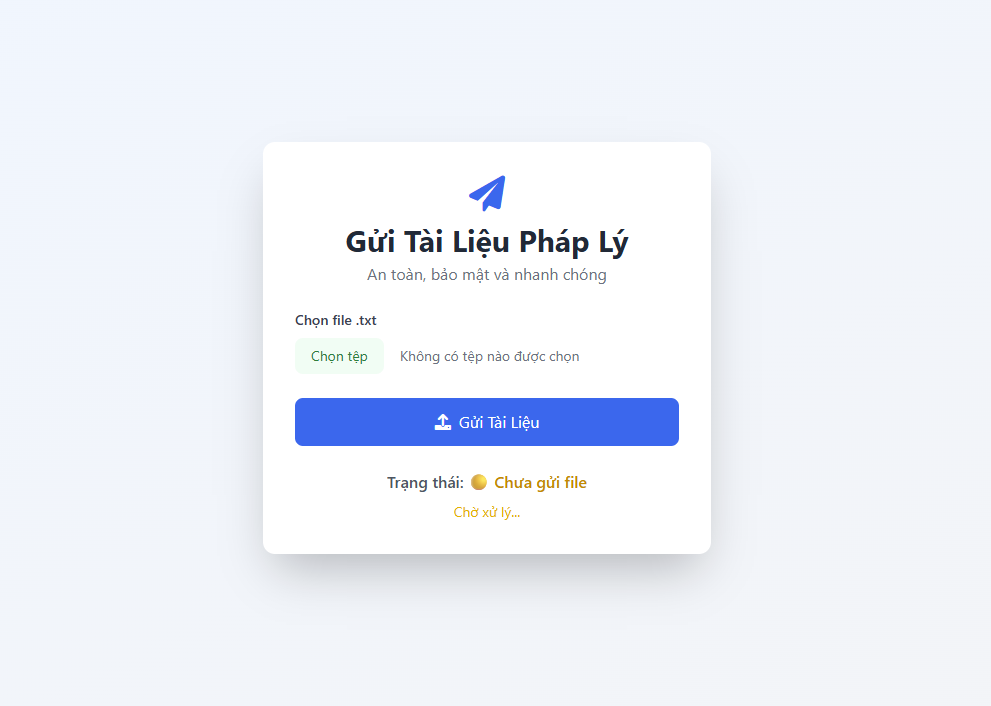
## 3.1 Kết quả

Giao diện người gửi

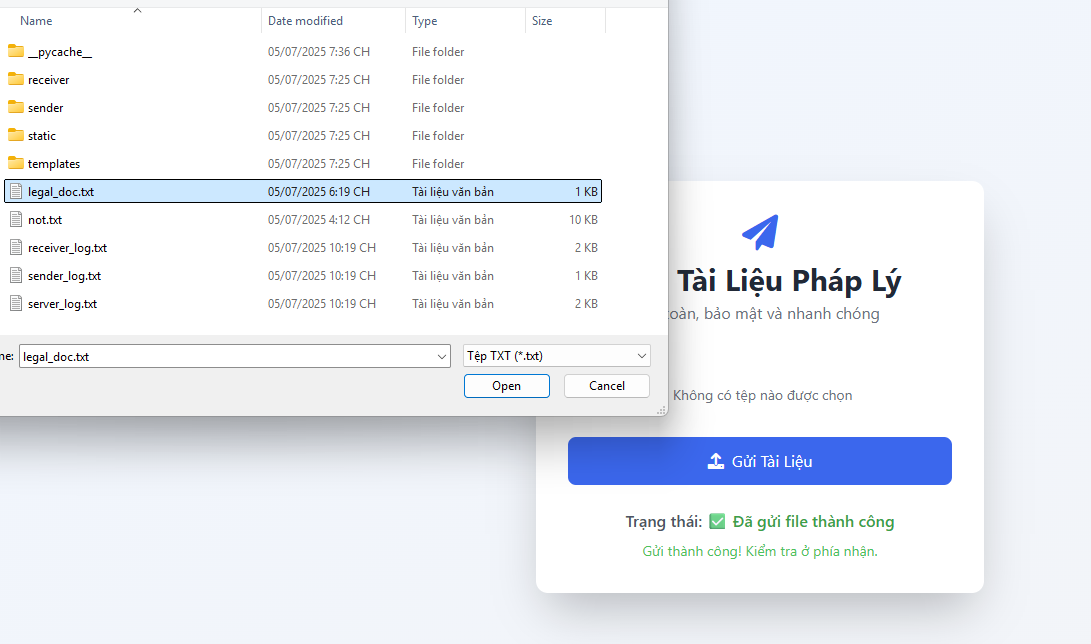
Sau khi chạy code , chúng ta sẽ có 1 giao diện hiển thị cho phần người gửi gồm các mục :

* + - * Chọn tệp
      * Gửi tài liệu
      * Hiển thị trạng thái hoạt động.

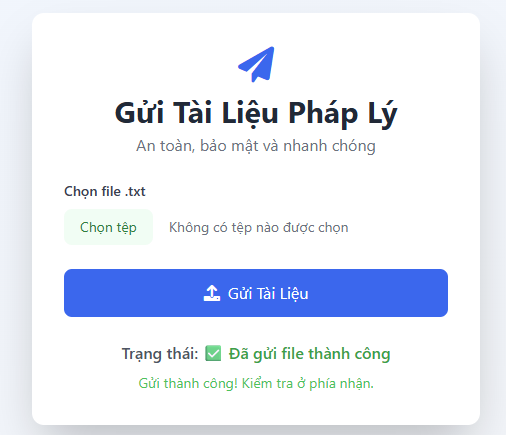




Người gửi trước khi gửi sẽ cần có 1 file text , khi vào gioa diện người gửi sẽ chọn file cần gửi.

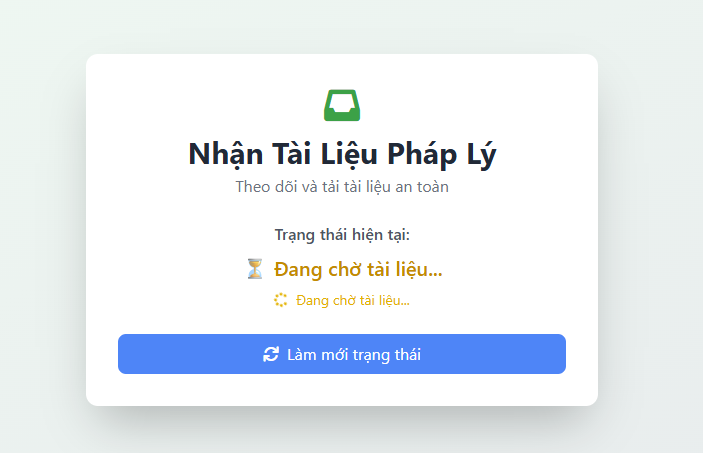


Sau khi gửi file thành công sẽ có 1 giao diện hiển thị thông báo hoạt động.

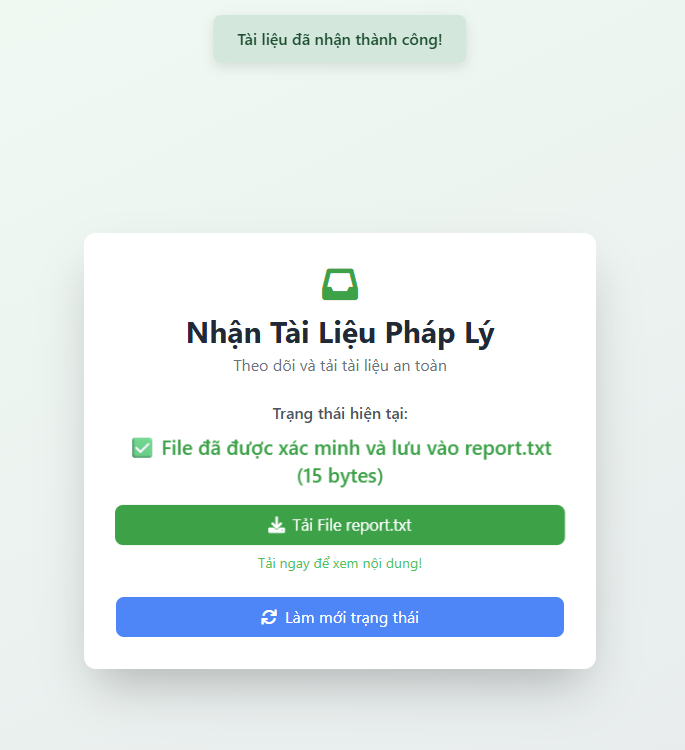


Giao diện nhận và giải mã.

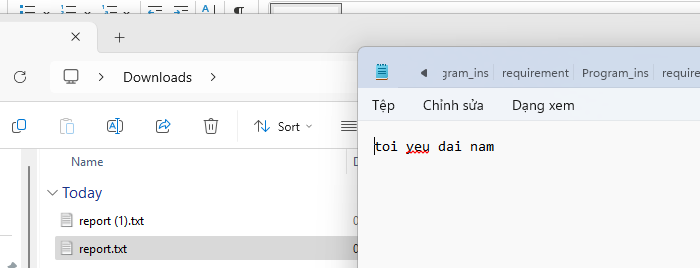
Giao diện gồm trạng thái trước khi nhận được file từ phía người gửi và nút làm mới.



Sau khi vào giao diện người nhận và gửi rồi sẽ tự động giải mã sau đó ta có thể chọn tải về



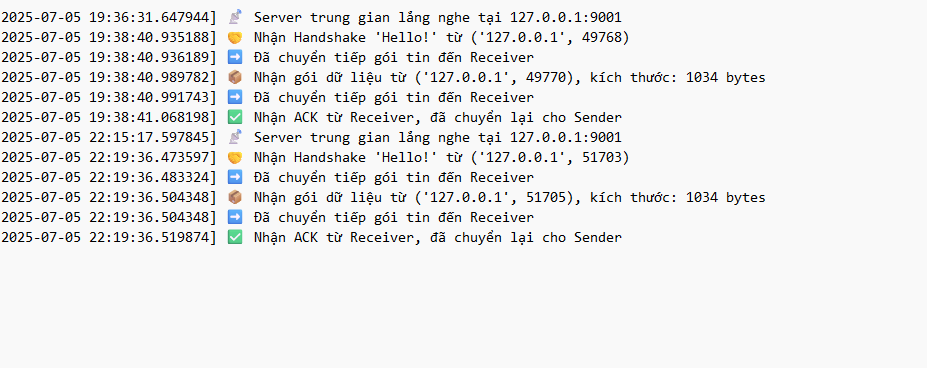
Sau khi quy trình thực hiện thành công file đã được mã hóa



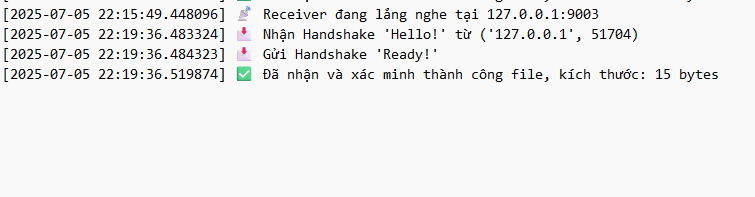
Ta có thể xem log hoạt động của trang được lưu ở file txt.

## 3.2 Người gửi.

## 3.3 Server.



## 3.4 Người nhận.



# KẾT LUẬN

Trong đề tài này, nhóm đã xây dựng thành công một hệ thống gửi file report.txt từ công ty đến đối tác thông qua một server trung gian, đảm bảo các tiêu chí về bảo mật như: mã hóa nội dung bằng AES-GCM, trao đổi khóa bằng RSA 1024-bit (OAEP), kiểm tra toàn vẹn bằng SHA-512 và xác thực nguồn gốc qua chữ ký số RSA/SHA-512. Hệ thống tuân thủ đúng mô hình giao tiếp gián tiếp, trong đó server trung gian chỉ đóng vai trò trung chuyển và ghi log thời gian, không tham gia vào quá trình mã hóa hay xác minh bảo mật.

Quá trình triển khai được chia thành các thành phần độc lập gồm: người gửi, người nhận, server trung gian và thư viện dùng chung. Việc phân tách này giúp chương trình rõ ràng, dễ mở rộng và thuận tiện trong kiểm thử. Các thao tác bảo mật được tích hợp trực tiếp vào quy trình gửi và nhận file, đảm bảo dữ liệu được bảo vệ ngay từ khâu chuẩn bị cho đến khi lưu trữ tại đầu nhận. Hệ thống sử dụng socket TCP để truyền dữ liệu và Flask để tạo giao diện web trực quan.

Kết quả thực nghiệm cho thấy hệ thống hoạt động ổn định, đúng theo yêu cầu của đề bài và đáp ứng đầy đủ các nguyên tắc về bảo mật thông tin: tính **bí mật**, **xác thực**, **toàn vẹn** và **không chối bỏ**. Qua đề tài này, nhóm đã nắm vững hơn các kỹ thuật mã hóa hiện đại, hiểu rõ cách kết hợp nhiều cơ chế bảo mật trong cùng một hệ thống, đồng thời rèn luyện kỹ năng thiết kế phần mềm bảo mật hiệu quả và có tính thực tiễn cao.

# DANH MỤC TÀI LIỆU THAM KHẢO

1. **William Stallings.** *Cryptography and Network Security – Principles and Practice*, 7th Edition. Pearson Education, 2017.
2. **Bruce Schneier.** *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd Edition. Wiley, 1996.
3. **National Institute of Standards and Technology (NIST).**  
   *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*.  
   NIST Special Publication 800-38D, 2007.
4. **Python Cryptographic Authority.** *PyCryptodome Documentation*.
5. **Flask Documentation – Web framework for Python.**