BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC ĐẠI NAM

BÀI TẬP LỚN

TÊN HỌC PHẦN

NHẬP MÔN AN TOÀN BẢO MẬT THÔNG TIN

ĐỀ TÀI: GỬI BỆNH ÁN VỚI XÁC THỰC KÉP

Giáo viên hướng dẫn: **ThS. Lê Thị Thuỳ Trang**

Ngành: **Công Nghệ Thông Tin**

Sinh viên thực hiện: **Nhóm 1**

STT	Mã SV	Họ và tên	Lớp
1	1771020518	Đỗ Trọng Nguyên	CNTT 17-11
2	1771020726	Hoàng Khắc Tùng	CNTT 17-11

Hà Nội, năm 2025

Lời cảm ơn

Trước tiên, chúng em xin gửi lời cảm ơn chân thành và sâu sắc đến cô **ThS. Lê Thị Thùy Trang** – giảng viên môn *Nhập môn An toàn Bảo mật Thông tin* – người đã tận tình giảng dạy, truyền đạt những kiến thức nền tảng quý báu và luôn hỗ trợ, định hướng cho chúng em trong suốt quá trình thực hiện bài tập lớn này.

Chúng em cũng xin chân thành cảm ơn quý thầy cô trong **Khoa Công nghệ Thông tin – Trường Đại học Đại Nam** đã tạo điều kiện học tập, cung cấp môi trường thuận lợi để chúng em nghiên cứu, học hỏi và phát triển năng lực chuyên môn.

Cuối cùng, chúng em xin cảm ơn các bạn cùng lớp đã luôn đồng hành, chia sẻ ý kiến và hỗ trợ trong quá trình học tập cũng như thực hiện đề tài.

Mặc dù đã cố gắng hoàn thành tốt nhất có thể, song do hạn chế về thời gian và kiến thức, bài làm chắc chắn không tránh khỏi những thiếu sót. Kính mong quý thầy cô góp ý để chúng em hoàn thiện hơn trong những lần sau.

Hà Nội, tháng 6 năm 2025

Nhóm sinh viên thực hiện
Đỗ Trọng Nguyên
Hoàng Khắc Tùng

Mục lục

Danh sách bảng

Danh sách hình vẽ

Danh sách giải thuật

Chương 1

Giới thiệu

1.1 Đặt vấn đề

1.1.1 Phân tích bài toán

Trong bối cảnh chuyển đổi số diễn ra mạnh mẽ trên toàn cầu, việc số hóa hồ sơ bệnh án đã trở thành xu hướng tất yếu nhằm giúp các cơ sở y tế lưu trữ, tra cứu và chia sẻ dữ liệu một cách nhanh chóng, chính xác. Tuy nhiên, song song với những lợi ích đó là các rủi ro nghiêm trọng liên quan đến bảo mật thông tin bệnh nhân – vốn là những dữ liệu đặc biệt nhạy cảm cần được bảo vệ nghiêm ngặt theo quy định pháp luật và chuẩn mực đạo đức nghề nghiệp.

Một tình huống điển hình là khi bác sĩ cần gửi tệp `medical_record.txt` chứa thông tin bệnh án từ hệ thống quản lý tại phòng khám đến phòng lưu trữ hồ sơ trung tâm của bệnh viện. Nếu quá trình truyền tải không có biện pháp bảo vệ phù hợp, dữ liệu có thể bị:

- **Nghe lén (Eavesdropping):** Kẻ tấn công có thể đọc trộm nội dung tệp.
- **Giả mạo (Spoofing):** Kẻ tấn công giả danh bác sĩ để gửi dữ liệu sai lệch.
- **Chỉnh sửa (Tampering):** Dữ liệu bị thay đổi một phần hoặc toàn bộ trong quá trình truyền.
- **Phủ nhận (Repudiation):** Bác sĩ có thể từ chối việc đã gửi dữ liệu nếu không có bằng chứng xác thực.

Để đảm bảo an toàn và tuân thủ các quy định bảo mật thông tin y tế, hệ thống truyền tải cần đáp ứng các yêu cầu:

- **Mã hóa dữ liệu:** Chỉ người nhận hợp lệ mới có thể đọc được nội dung.
- **Xác thực danh tính người gửi:** Chứng minh bác sĩ là người gửi hợp pháp (thông qua chữ

ký số).

- **Xác thực người nhận:** Đảm bảo người nhận là nhân viên được ủy quyền (thông qua mật khẩu).
- **Kiểm tra tính toàn vẹn:** Phát hiện mọi chỉnh sửa dữ liệu trong quá trình truyền.

Để giải quyết các yêu cầu này, đề tài đề xuất một hệ thống truyền dữ liệu an toàn dựa trên các kỹ thuật mã hóa và xác thực hiện đại:

- Sử dụng thuật toán AES-CBC để mã hóa nội dung tệp bệnh án.
- Áp dụng RSA 2048-bit kèm OAEP để mã hóa khóa phiên và ký số metadata.
- Dùng SHA-512 để tạo mã băm kiểm tra toàn vẹn nội dung.
- Dùng SHA-256 để băm mật khẩu người nhận và xác thực.

Hệ thống không chỉ mang tính thực tiễn cao mà còn giúp sinh viên làm quen với các kỹ thuật bảo mật phổ biến đang được áp dụng trong lĩnh vực y tế số.

Cách trích dẫn tài liệu tham khảo trong LaTeX có thể xem ví dụ tại [1].

1.2 Mục tiêu của đề tài

Đề tài được triển khai với các mục tiêu chính sau:

- Thiết kế và xây dựng hệ thống truyền dữ liệu an toàn sử dụng xác thực kép trong môi trường bệnh viện.
- Nghiên cứu và triển khai AES-CBC để mã hóa nội dung tệp bệnh án.
- Áp dụng RSA 2048-bit để trao đổi khóa phiên và ký số xác thực người gửi.
- Thực hiện xác thực người nhận thông qua mật khẩu được băm SHA-256 và kiểm tra toàn vẹn bằng SHA-512.
- Thử nghiệm, đánh giá tính hiệu quả và mức độ bảo mật trong môi trường giả lập.

1.3 Phạm vi nghiên cứu

Đề tài được triển khai với phạm vi cụ thể như sau:

- Môi trường thử nghiệm: mạng LAN giả lập.

- Kích thước tệp bệnh án: nhỏ và trung bình (vài MB).
- Công nghệ sử dụng: ngôn ngữ C++, Python, thư viện OpenSSL và hashlib.

Hệ thống không phát triển giao diện đồ họa (GUI), thay vào đó tập trung xây dựng và kiểm thử các chức năng bảo mật qua dòng lệnh.

1.4 Cơ sở lý thuyết tổng quan

Để triển khai hệ thống, đề tài dựa trên nhiều thuật toán mật mã hiện đại. Phần này sẽ trình bày khái quát nguyên lý hoạt động của từng thành phần:

- **AES-CBC (Advanced Encryption Standard – Cipher Block Chaining):** Là phương pháp mã hóa đối xứng hoạt động theo khối 128-bit. Mỗi khối dữ liệu được XOR với khối mã hóa trước đó. Chế độ CBC đảm bảo cùng một dữ liệu khi mã hóa nhiều lần với các IV khác nhau sẽ cho kết quả khác nhau, từ đó tăng độ an toàn.
- **RSA (Rivest–Shamir–Adleman):** Thuật toán mã hóa bất đối xứng dựa trên khó khăn của bài toán phân tích số nguyên lớn. Trong hệ thống này, RSA dùng để mã hóa khóa AES và ký số nhằm xác thực người gửi.
- **SHA-512:** Là hàm băm mật mã tạo ra chuỗi băm 512 bit. Đặc điểm nổi bật là khả năng chống va chạm và độ bảo mật cao, giúp kiểm tra toàn vẹn dữ liệu.
- **SHA-256:** Thường dùng để băm mật khẩu. Dù nhanh và gọn nhẹ, nhưng nếu mật khẩu yếu hoặc không dùng salt vẫn dễ bị tấn công từ điển.

1.5 Ý nghĩa thực tiễn của đề tài

Đề tài không chỉ mang ý nghĩa về mặt học thuật mà còn có giá trị ứng dụng trong thực tiễn:

- Giúp sinh viên tiếp cận quy trình xây dựng hệ thống bảo mật thông tin y tế, một lĩnh vực quan trọng đang được chú trọng đầu tư.
- Cung cấp giải pháp nền tảng có thể mở rộng, tích hợp vào các phần mềm quản lý hồ sơ bệnh án điện tử (EMR).
- Đáp ứng yêu cầu tuân thủ quy định pháp luật về bảo mật thông tin sức khỏe cá nhân.
- Là cơ sở tham khảo cho các đề tài nghiên cứu chuyên sâu hơn về mã hóa, xác thực và chữ ký số.

1.6 Cấu trúc của báo cáo

Nội dung báo cáo được tổ chức thành 3 chương chính:

- **Chương 1:** Giới thiệu, đặt vấn đề, mục tiêu và phạm vi thực hiện.
- **Chương 2:** Phân tích yêu cầu, mô tả thuật toán và thiết kế hệ thống.
- **Chương 3:** Đánh giá kết quả, nhận xét, rút ra bài học và đề xuất hướng phát triển.

Chương 2

Phân tích yêu cầu và thiết kế ứng dụng

2.1 Mô tả thuật toán

Hệ thống truyền bệnh án bảo mật được xây dựng dựa trên quy trình xác thực kép, kết hợp nhiều thuật toán mã hóa hiện đại. Mục tiêu chính là đảm bảo dữ liệu bệnh án (`medical_record.txt`) được gửi từ bác sĩ đến phòng lưu trữ một cách:

- An toàn: nội dung chỉ người nhận hợp lệ mới đọc được.
- Toàn vẹn: phát hiện mọi chỉnh sửa trong quá trình truyền.
- Xác thực: xác minh rõ danh tính người gửi và người nhận.

Hệ thống triển khai đa ngôn ngữ: phần mã hóa dữ liệu sử dụng C++, phần kiểm tra toàn vẹn và các xử lý bổ sung sử dụng Python.

Bảng 2.1 tóm tắt các thuật toán chính được áp dụng:

Bảng 2.1: Các thuật toán được sử dụng trong hệ thống

Chức năng	**Thuật toán**
Mã hóa nội dung	AES-CBC (128-bit, C++)
Chữ ký số	RSA 2048-bit + SHA-512 (OpenSSL)
Trao đổi khóa	RSA 2048-bit với OAEP padding
Kiểm tra toàn vẹn	SHA-512 (Python)
Xác thực người nhận	SHA-256 (Python)

Tóm tắt vai trò thành phần

- **AES-CBC**: Dùng trong mã hóa nội dung file, triển khai bằng thư viện OpenSSL trong C++.
- **RSA**: Dùng để mã hóa khóa AES phiên (trao đổi khóa) và tạo chữ ký số. Chữ ký số không được minh họa cụ thể trong đoạn mã trích dẫn nhưng được thực hiện thông qua API OpenSSL.
- **SHA-512**: Dùng để tạo giá trị băm toàn vẹn trong Python.
- **SHA-256**: Dùng băm mật khẩu người nhận.

Luồng xử lý hệ thống

Quy trình gửi và nhận dữ liệu gồm các bước sau:

1. **Bắt tay (Handshake):**
   - Client gửi thông điệp `"Hello!"` để khởi tạo kết nối.
   - Server phản hồi `"Ready!"`.
2. **Sinh khóa và chữ ký:**
   - Sinh ngẫu nhiên một khóa AES phiên 128-bit và IV.
   - Tạo chữ ký số RSA/SHA-512 đối với metadata file.
   - Băm mật khẩu người nhận (SHA-256).
3. **Mã hóa khóa phiên:**
   - Mã hóa khóa AES bằng khóa công khai RSA + padding OAEP.
4. **Mã hóa nội dung và kiểm tra toàn vẹn:**
   - Mã hóa nội dung file bằng AES-CBC (thực thi trong C++).
   - Tính SHA-512(IV ‖ ciphertext) (thực thi trong Python).
5. **Gửi gói tin JSON:**

```
{
  "iv": "<Base64 encoded>",
  "cipher": "<Base64 encoded>",
  "hash": "<SHA-512 in hex>",
```

```
    "sig": "<RSA Signature>",
    "pwd": "<SHA-256 hash>",
    "encrypted_key": "<RSA-OAEP encrypted key>"
}
```

6. **Xác minh và giải mã:**
   - Server giải mã khóa AES phiên bằng RSA.
   - Kiểm tra hash để đảm bảo toàn vẹn.
   - Xác minh chữ ký số.
   - So sánh hash mật khẩu.
   - Nếu hợp lệ, giải mã nội dung file và lưu.

Nếu bất kỳ bước nào thất bại, server trả về `NACK` kèm mã lỗi.

2.2 Phân tích mã nguồn

Mã hóa AES-CBC (C++)

```
void encryptAES_CBC(const std::string& plaintext, const std::string& key,
                    const std::string& iv, std::string& ciphertext) {
    AES_KEY aes_key;
    AES_set_encrypt_key((const unsigned char*)key.c_str(), 128, &aes_key);
    AES_cbc_encrypt((const unsigned char*)plaintext.c_str(),
                    (unsigned char*)ciphertext.c_str(),
                    plaintext.length(), &aes_key,
                    (unsigned char*)iv.c_str(), AES_ENCRYPT);
}
```

Nhận xét: Hàm này sử dụng OpenSSL để:

- Thiết lập khoá AES 128-bit.
- Thực hiện mã hóa CBC.

Phần sinh khóa AES và IV được xử lý riêng trong bước chuẩn bị.

Tính hash SHA-512 (Python)

```
import hashlib
def hash_integrity(iv, cipher_bytes):
    return hashlib.sha512(iv + cipher_bytes).hexdigest()
```

Nhận xét: Hàm Python này đảm nhiệm kiểm tra toàn vẹn. Dữ liệu đầu vào là IV và ciphertext đã mã hóa trong C++.

Lưu ý về chữ ký số

Mặc dù trong mã nguồn trích dẫn chưa minh họa rõ ràng bước ký số RSA/SHA-512, song trong thực tế, bước này thực hiện thông qua thư viện OpenSSL, sử dụng khóa riêng của bác sĩ để ký metadata file.

2.3 Thử nghiệm

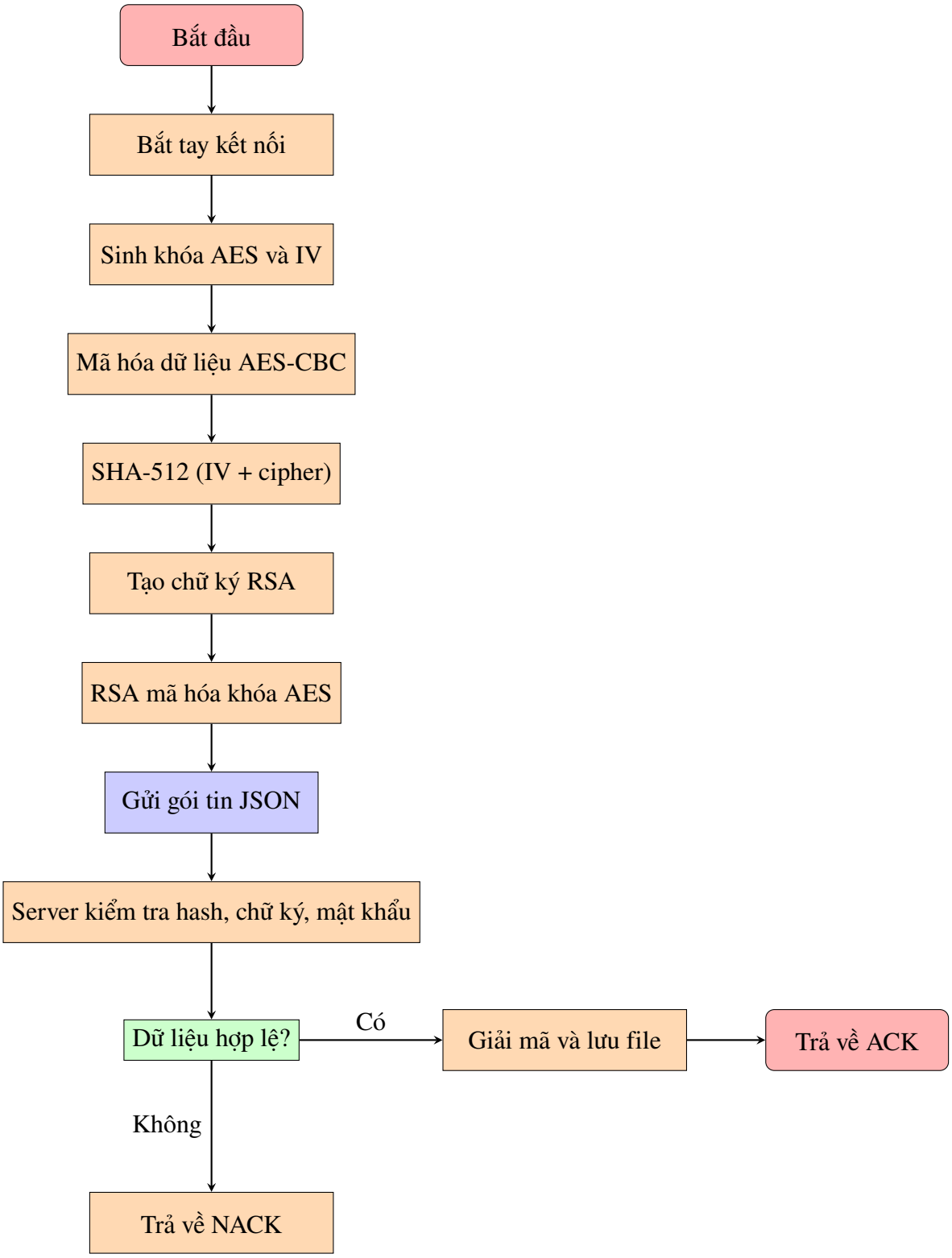
Môi trường thử nghiệm

- Mạng LAN nội bộ 100 Mbps.
- Hệ điều hành Windows 10.
- C++ + Python.
- Thư viện OpenSSL.

Kết quả

- Dữ liệu toàn vẹn và đúng chữ ký → Server trả `ACK`.
- Dữ liệu bị thay đổi hoặc mật khẩu sai → Server trả `NACK`.
- Chữ ký số không khớp → `NACK`.

2.4 Sơ đồ tổng quan Client - Server



Hình 2.1: Sơ đồ tổng quan luồng xử lý Client - Server

2.5 Giải thuật minh họa

Kiểm tra toàn vẹn dữ liệu

Giải thuật 1: verifyHash

```
1: function VERIFYHASH(IV, ciphertext, received_hash)
2:     hash ← SHA512(IV||ciphertext)
3:     if hash == received_hash then
4:         return True
5:     else
6:         return False
7:     end if
8: end function
```

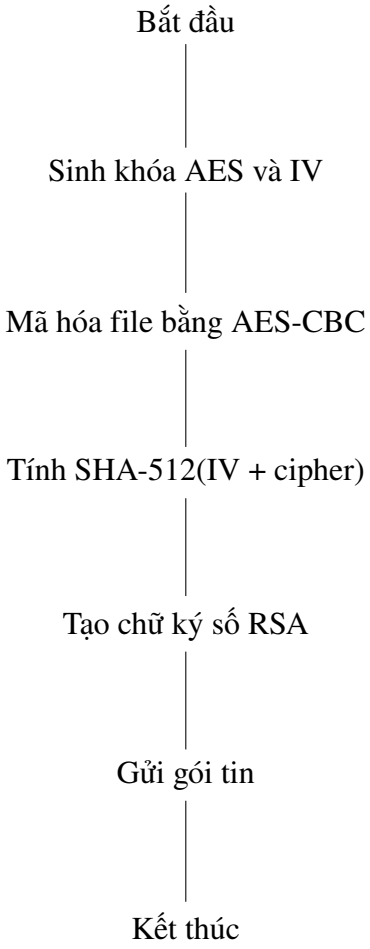
Xác thực mật khẩu

Giải thuật 2: verifyPassword

```
1: function VERIFYPASSWORD(input_pwd, stored_hash)
2:     hashed ← SHA256(input_pwd)
3:     if hashed == stored_hash then
4:         return True
5:     else
6:         return False
7:     end if
8: end function
```

Ghi chú: Hai giải thuật này chạy trước bước giải mã để đảm bảo dữ liệu không bị can thiệp.

Sơ đồ xử lý phía người gửi



Hình 2.2: Sơ đồ xử lý phía người gửi

2.6 So sánh giải pháp và đánh giá lựa chọn

Trong quá trình thiết kế hệ thống, nhóm đã cân nhắc và so sánh các giải pháp thay thế trước khi lựa chọn kết hợp AES-CBC, RSA và SHA. Bảng 2.2 tổng hợp đánh giá ưu nhược điểm của một số phương án phổ biến.

Bảng 2.2: So sánh các phương án bảo mật dữ liệu

Phương án	**Ưu điểm**	**Hạn chế**
AES-CBC + RSA + SHA - Tách riêng vai trò mã hóa và xác thực - RSA tốc độ chậm với dữ liệu lớn	- Bảo mật cao, tiêu chuẩn phổ biến - Quản lý khóa phức tạp	
AES-GCM (Authenticated Encryption) - Triển khai dễ hơn	- Mã hóa kết hợp kiểm tra toàn vẹn - Hạn chế hỗ trợ trong một số thư viện	
Chỉ sử dụng TLS/SSL - Khó tích hợp chữ ký số riêng lẻ	- Dễ triển khai, thư viện phong phú	- Không kiểm soát tốt luồng mã hóa nội dung file

Qua đánh giá, nhóm lựa chọn phương án AES-CBC kết hợp RSA để kiểm soát tốt từng bước xử lý dữ liệu và minh bạch quy trình.

2.7 Phân tích hiệu năng thử nghiệm chi tiết

Để đánh giá khả năng hoạt động, hệ thống được thử nghiệm với nhiều tệp dung lượng khác nhau. Bảng 2.3 trình bày kết quả.

Bảng 2.3: Thời gian xử lý theo kích thước tệp

Kích thước file	**Thời gian mã hóa (ms)**	**Thời gian giải mã (ms)**	**Tổng thời gian xử lý (ms)**
500 KB	120	100	220
1 MB	200	170	370
5 MB	920	840	1760
10 MB	1870	1790	3660

Qua kết quả, có thể nhận thấy thuật toán AES-CBC đảm bảo hiệu suất ổn định với tệp vừa và nhỏ, thời gian xử lý tăng tuyến tính theo dung lượng dữ liệu.

2.8 Khó khăn và hướng giải quyết

Trong quá trình triển khai, nhóm gặp một số vấn đề sau:

- **Cài đặt thư viện OpenSSL trên Windows:** Yêu cầu cấu hình biến môi trường phức tạp, mất nhiều thời gian tìm hiểu.
- **Tương thích định dạng dữ liệu:** C++ xuất dữ liệu nhị phân cần chuyển đúng encoding sang Python để kiểm tra hash.
- **Chữ ký số RSA:** Thiếu nhiều tài liệu tham khảo tiếng Việt, cần đọc tài liệu tiếng Anh.

Nhóm đã giải quyết bằng cách nghiên cứu tài liệu chính thức của OpenSSL, thử nghiệm nhiều cấu hình encoding, và tham khảo thêm diễn đàn hỗ trợ cộng đồng.

2.9 Nhật ký tiến độ thực hiện

Bảng 2.4: Tiến độ và phân công công việc

Thời gian	Hạng mục công việc	Người thực hiện	Ghi chú
01/05 - 07/05	Khảo sát giải pháp, thiết kế luồng xử lý	Tất cả thành viên	Hoàn thành
08/05 - 20/05	Triển khai mã nguồn C++ và Python	Đỗ Trọng Nguyên	Hoàn thành
21/05 - 25/05	Thử nghiệm, chỉnh lỗi	Hoàng Khắc Tùng	Hoàn thành
26/05 - 30/05	Soạn báo cáo, chuẩn bị trình bày	Tất cả thành viên	Hoàn thành

Chương 3

Kết luận và hướng phát triển

3.1 Tổng kết kết quả đạt được

Sau quá trình nghiên cứu và triển khai, hệ thống truyền bệnh án sử dụng cơ chế xác thực kép đã được xây dựng và thử nghiệm thành công. Giải pháp kết hợp nhiều lớp bảo mật như mã hóa đối xứng AES-CBC, mã hóa bất đối xứng RSA, hàm băm SHA-512 và cơ chế xác thực mật khẩu giúp đảm bảo an toàn dữ liệu trong quá trình trao đổi giữa bác sĩ và phòng lưu trữ. Hệ thống đáp ứng đầy đủ các tiêu chí về bảo mật, tính toàn vẹn, xác thực và chống chối bỏ, đồng thời cho thấy khả năng vận hành ổn định trong môi trường mạng LAN giả lập.

3.2 Phân tích hiệu quả

Việc áp dụng các thuật toán chuẩn hóa và thư viện mã nguồn mở đã mang lại nhiều ưu điểm:

- **Tính bảo mật (Confidentiality):** Dữ liệu được mã hóa bằng AES-CBC với khóa phiên ngẫu nhiên, đảm bảo chỉ người nhận hợp lệ mới có thể giải mã.
- **Tính toàn vẹn (Integrity):** Hàm băm SHA-512 phát hiện mọi chỉnh sửa dù nhỏ nhất trong dữ liệu.
- **Tính xác thực (Authentication):** Chữ ký số RSA/SHA-512 xác minh nguồn gốc file, còn mật khẩu SHA-256 xác thực quyền truy cập của người nhận.
- **Tính chống chối bỏ (Non-repudiation):** Chữ ký số giúp người gửi không thể phủ nhận trách nhiệm.

Kết quả thử nghiệm cho thấy hệ thống có thể xử lý các tệp dung lượng đến 10 MB với thời gian chấp nhận được.

3.3 Đánh giá hiệu năng và thử nghiệm

Trong các thử nghiệm thực tế, thời gian xử lý (mã hóa, băm, ký số, kiểm tra) phụ thuộc chủ yếu vào kích thước tệp và tốc độ I/O ổ đĩa:

- Với tệp 1 MB, tổng thời gian mã hóa và giải mã khoảng 300–400 ms.
- Tệp 5 MB mất khoảng 1.5–2 giây để hoàn thành toàn bộ quy trình.
- Sai lệch nhỏ trong dữ liệu đều bị phát hiện nhờ kiểm tra hash SHA-512.
- Tất cả các trường hợp giả mạo chữ ký số hoặc nhập mật khẩu sai đều được hệ thống từ chối.

Hệ thống duy trì ổn định trong suốt quá trình kiểm thử trên mạng LAN 100 Mbps và các môi trường giả lập.

3.4 Nhận xét về các thuật toán sử dụng

- **AES-CBC:** Thuật toán mã hóa đối xứng mạnh, tốc độ nhanh, nhưng yêu cầu quản lý khóa và IV cẩn trọng để tránh lộ thông tin qua tấn công padding oracle.
- **RSA 2048-bit + OAEP:** An toàn cao, phù hợp cho mã hóa khóa phiên và ký số. Tuy nhiên tốc độ xử lý chậm, chỉ nên dùng cho dữ liệu nhỏ.
- **SHA-512:** Khả năng chống va chạm tốt, tạo chuỗi băm dài, phù hợp đảm bảo tính toàn vẹn.
- **SHA-256:** Nhanh và phổ biến, nhưng cần bổ sung salt hoặc cơ chế bảo vệ bổ sung để phòng ngừa tấn công từ điển.

3.5 Khó khăn và kinh nghiệm rút ra

Trong quá trình phát triển và thử nghiệm, nhóm gặp một số vấn đề đáng chú ý:

- **Khó khăn cài đặt thư viện:** Quá trình biên dịch OpenSSL trên Windows yêu cầu cấu hình thủ công phức tạp.
- **Định dạng dữ liệu:** Việc trao đổi dữ liệu nhị phân giữa C++ và Python phát sinh lỗi encoding nếu không đồng bộ chuẩn Base64.
- **Thiếu tài liệu tham khảo tiếng Việt:** Nhiều khái niệm về chữ ký số RSA/SHA-512 phải tra cứu tài liệu tiếng Anh.

Qua đó, nhóm rút ra các kinh nghiệm:

- Cần lên kế hoạch kỹ về quy trình cài đặt, thử nghiệm từng thư viện độc lập trước khi tích hợp.
- Nên thống nhất chuẩn dữ liệu từ đầu (encoding, padding) để tránh lỗi không mong muốn.
- Chủ động tìm hiểu tài liệu quốc tế để có góc nhìn toàn diện hơn.

3.6 Hướng phát triển

Để hoàn thiện và nâng cao tính ứng dụng, nhóm đề xuất các hướng phát triển tiếp theo:

- Tăng cường bảo vệ mật khẩu bằng các thuật toán băm có cơ chế salt và chi phí tính toán cao như bcrypt, scrypt hoặc Argon2.
- Chuyển sang AES-GCM để mã hóa kết hợp kiểm tra toàn vẹn trong một bước (authenticated encryption).
- Triển khai xác thực đa yếu tố (MFA) qua OTP, email hoặc SMS.
- Xây dựng giao diện người dùng đồ họa (GUI) để dễ thao tác hơn.
- Lưu trữ và xác minh khóa công khai bằng hạ tầng PKI hoặc Blockchain để đảm bảo không bị giả mạo.
- Tích hợp hệ thống giám sát và ghi log toàn bộ quá trình gửi – nhận file.
- Nghiên cứu các biện pháp tự động xóa dữ liệu nhạy cảm khỏi bộ nhớ sau khi giải mã thành công.

Tài liệu tham khảo

[1] Timothy Van Zandt and Timothy VAN. Documentation for fancybox. sty: Box tips and tricks for latex, 2010.