TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP

KHOA ĐIỆN TỬ

BÀI TẬP LỚN AN TOÀN VÀ BẢO MẬT THÔNG TIN

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

MÔN HỌC: AN TOÀN VÀ BẢO MẬT THÔNG TIN

GIÁO VIÊN HƯỚNG DẪN	:	THS. ĐỖ DUY CỐP
HỌ VÀ TÊN SINH VIÊN	:	ĐẬU VĂN KHÁNH
LỚP	:	K58KTP.K01
MSSV	:	K225480106099

THÁI NGUYÊN - 2025

TRƯỜNG ĐHKTCN **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

KHOA ĐIỆN TỬ **Độc lập - Tự do - Hạnh phúc**

PHIẾU GIAO BÀI TẬP LỚN AN TOÀN VÀ BẢO MẬT THÔNG TIN

Sinh viên: Đậu Văn Khánh MSSV: K225480106099

Lớp: K58KTP.K01 Khoá: 2022-2027

Ngành học: Kỹ thuật phần mềm

Giáo viên hướng dẫn: ThS. Đỗ Duy Cốp

1. Tên đề tài: Bài tập An toàn và bảo mật thông tin

2. Nội dung thực hiện:

- Bài 1: Tìm hiểu các phương pháp mã hóa cổ điển
- Bài 2: Chữ ký số trong file PDF

3. Ngày giao nhiệm vụ: 22/10/2025

4.. Ngày hoàn thành nhiệm vụ: 22/11/2025

TRƯỞNG KHOA **BỘ MÔN** **GIÁO VIÊN HƯỚNG DẪN**

(Ký và ghi rõ họ tên) (Ký và ghi rõ họ tên) (Ký và ghi rõ họ tên)

TRƯỜNG ĐHKTCN **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

KHOA ĐIỆN TỬ ***Độc lập – Tự do – Hạnh phúc***

PHIẾU GHI ĐIỂM
HƯỚNG DẪN BÀI TẬP LỚN AN TOÀN VÀ BẢO MẬT THÔNG TIN

Sinh viên: Đậu Văn Khánh

MSSV: K225480106099

Lớp: K58KTP

GVHD: ThS. Đỗ Duy Cốp

Môn học: An toàn và bảo mật thông tin

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

……

……

……

……

……

……

Xếp loại: .....

Điểm: .....

Thái Nguyên, ngày….tháng…..năm 2025

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

LỜI CẢM ƠN

Em xin bày tỏ lòng biết ơn sâu sắc đến thầy Đỗ Duy Cốp, giảng viên Trường Đại học Kỹ thuật Công nghiệp Thái Nguyên, người đã tận tình hướng dẫn, chỉ bảo và đồng hành cùng em trong suốt quá trình thực hiện và hoàn thiện báo cáo bài tập lớn này. Nhờ sự giúp đỡ quý báu của thầy, em đã có thể vượt qua những khó khăn, từng bước hoàn thành tốt nội dung đề tài được giao.

Mặc dù đã cố gắng hết sức, nhưng trong quá trình thực hiện báo cáo, chắc chắn sẽ không tránh khỏi những thiếu sót nhất định. Em rất mong nhận được sự thông cảm và những góp ý quý báu từ các thầy, cô để em có thể rút kinh nghiệm, hoàn thiện bản thân hơn trong những chặng đường học tập và công tác sau này.

Em xin chân thành cảm ơn!

LỜI NÓI ĐẦU

Trong thời đại chuyển đổi số diễn ra mạnh mẽ, dữ liệu đã trở thành tài nguyên quan trọng và là yếu tố cốt lõi quyết định hiệu quả hoạt động của nhiều hệ thống thông tin. Song song với sự phát triển đó, những nguy cơ về rò rỉ dữ liệu, tấn công mạng và giả mạo thông tin ngày càng gia tăng, đặt ra yêu cầu cấp thiết về xây dựng các biện pháp bảo mật hiệu quả, đáng tin cậy. Chính vì vậy, việc nghiên cứu các kỹ thuật mã hóa, chữ ký số và các mô hình đảm bảo an toàn dữ liệu có ý nghĩa quan trọng, vừa mang tính nền tảng lý thuyết vừa có giá trị ứng dụng cao trong thực tiễn.

Bài tập lớn môn An toàn và Bảo mật thông tin giúp sinh viên tiếp cận hệ thống kiến thức cơ bản nhưng cốt lõi của lĩnh vực bảo mật. Trong bài báo cáo này, em tập trung tìm hiểu năm phương pháp mã hóa cổ điển gồm Caesar, Affine, Hoán vị, Vigenère và Playfair, qua đó hiểu rõ cách thức mã hóa – giải mã, không gian khóa và các kỹ thuật phân tích phá mã. Bên cạnh đó, phần nghiên cứu về chữ ký số trong tài liệu PDF mang lại cái nhìn sâu sắc hơn về cơ chế bảo vệ tính toàn vẹn và tính xác thực của tài liệu điện tử, đặc biệt thông qua các cấu trúc như Signature Dictionary, ByteRange hay cơ chế cập nhật gia tăng của PDF.

MỤC LỤC

BÀI TẬP 1: TÌM HIỂU CÁC PHƯƠNG PHÁP MÃ HOÁ CỔ ĐIỂN

ĐỀ BÀI:

❖ Tìm hiểu các phương pháp mã hóa cổ điển

1. Caesar
2. Affine
3. Hoán vị
4. Vigenère
5. Playfair

❖ Với mỗi phương pháp, hãy tìm hiểu:

1. Tên gọi
2. Thuật toán mã hoá, thuật toán giải mã
3. Không gian khóa
4. Cách phá mã (mà không cần khoá)
5. Cài đặt thuật toán mã hoá và giải mã bằng code C++ và bằng html+css+javascript

1.1. Phương pháp mã hóa Caesar

- Tên gọi: Mã Caesar (Caesar Cipher), còn gọi là mã dịch chuyển.
- Thuật toán mã hoá/giải mã:

  - Mã hóa: Mỗi ký tự trong bản rõ được thay thế bằng một ký tự khác ở vị trí cách nó một số cố định trong bảng chữ cái.

  - Công thức mã hóa: $C=(P+K)(mod26)$

  - Trong đó:

    + C: Ký tự trong bản mã

    + K: Khoảng dịch chuyển (khóa)

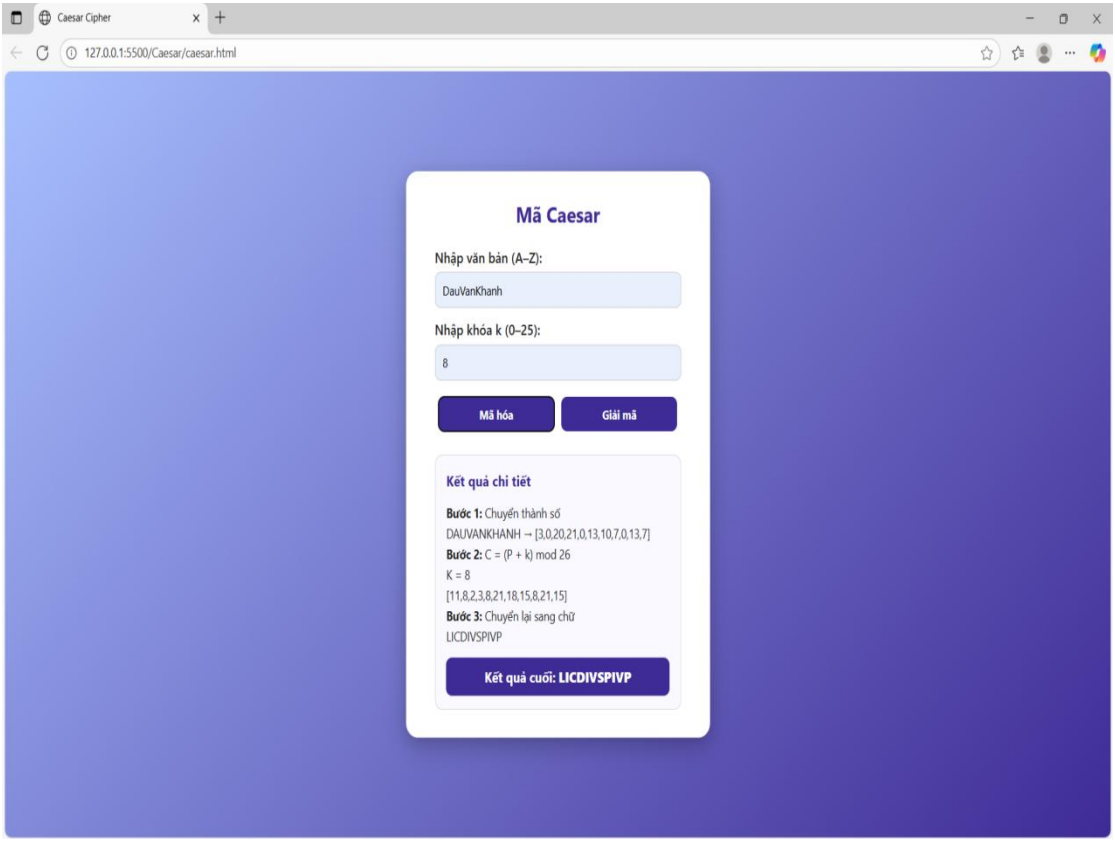
  - Giải mã: Ngược lại, mỗi ký tự trong bản mã được dịch chuyển ngược lại với cùng một khoảng cách.

  - Công thức giải mã: $P=(C-K)(mod26)$
- Không gian khóa: Không gian khóa của mã Caesar là rất nhỏ, chỉ có 25 khóa có thể (từ 1 đến 25). Vì khóa 0 không làm thay đổi bản rõ, nên nó không được tính.
- Cách phá mã (không cần khoá):

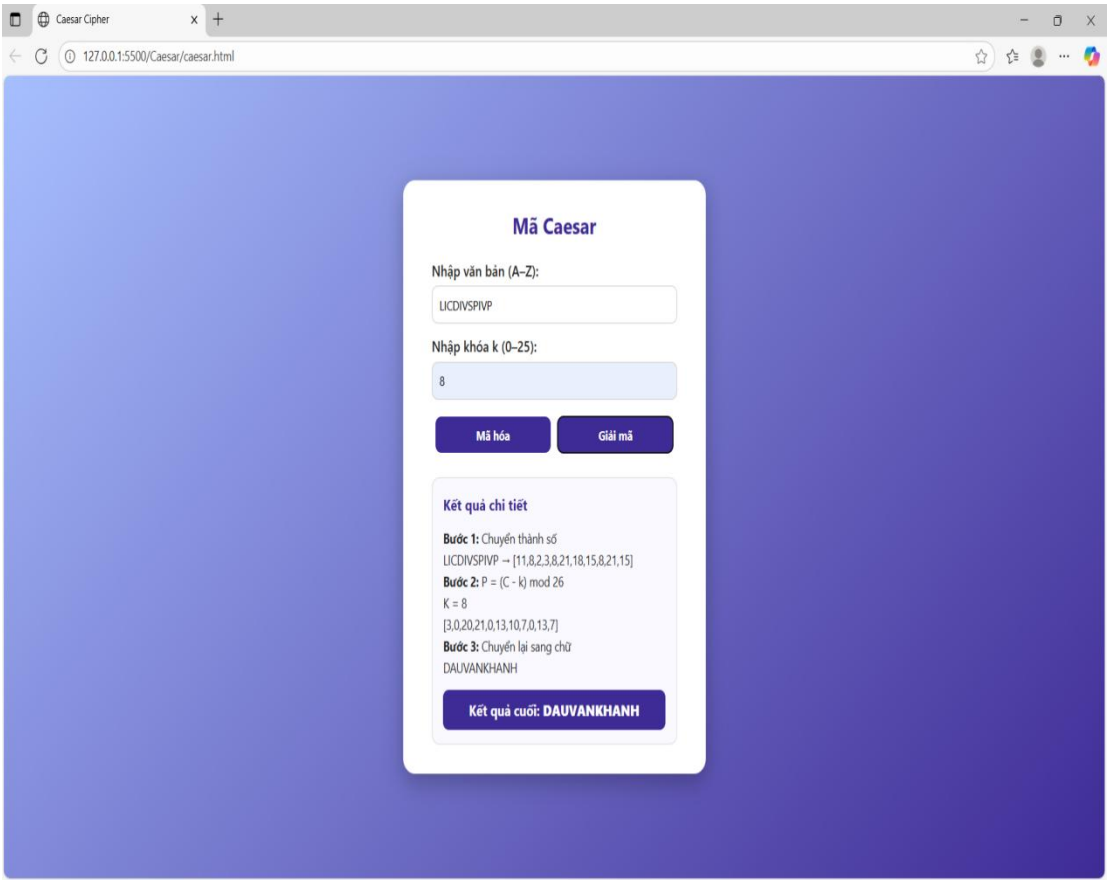
  - Brute-force: thử tất cả 26 giá trị k và đọc ra plaintext hợp lý. Thường dùng người đọc (human) hoặc kiểm tra bằng từ điển.

  - Phân tích tần suất: tìm ký tự xuất hiện nhiều nhất trong ciphertext (thường tương ứng E trong tiếng Anh) → suy k bằng cách so sánh.

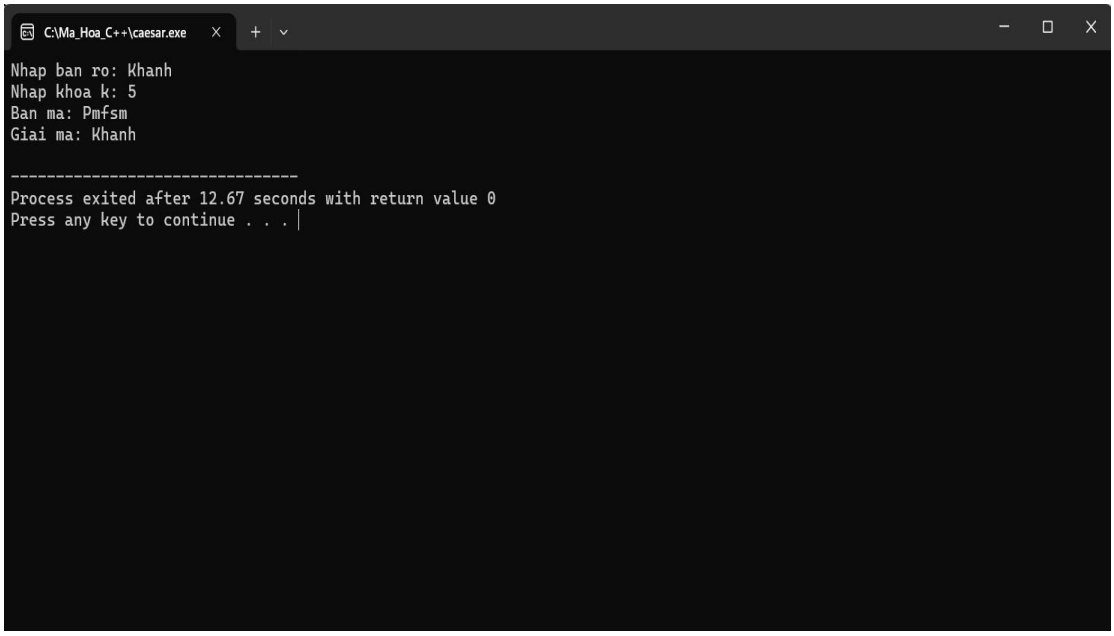
  - Kết hợp: dùng scoring (n-gram) để tự động chọn candidate tốt nhất.
- Mã hóa:


Caesar Cipher
127.0.0.1:5500/Caesar/caesar.html
Mã Caesar
Nhập văn bản (A–Z):
DauVanKhanh
Nhập khóa k (0–25):
8
Mã hóa
Giải mã
Kết quả chi tiết
Bước 1: Chuyển thành số
DAUVANKHANH → [3,0,20,21,0,13,10,7,0,13,7]
Bước 2: C = (P + k) mod 26
K = 8
[11,8,2,3,8,21,18,15,8,21,15]
Bước 3: Chuyển lại sang chữ
LICDIVSPIVP
Kết quả cuối: LICDIVSPIVP

❖ Giải mã:


Caesar Cipher
127.0.0.1:5500/Caesar/caesar.html
Mã Caesar
Nhập văn bản (A–Z):
LICDIVSPIVP
Nhập khóa k (0–25):
8
Mã hóa
Giải mã
Kết quả chi tiết
Bước 1: Chuyển thành số
LICDIVSPIVP → [11,8,2,3,8,21,18,15,8,21,15]
Bước 2: P = (C - k) mod 26
K = 8
[3,0,20,21,0,13,10,7,0,13,7]
Bước 3: Chuyển lại sang chữ
DAUVANKHANH
Kết quả cuối: DAUVANKHANH

❖ C++:



1.2. Phương pháp mã hóa Affine

❖ Tên gọi: Mã Affine

❖ Thuật toán:

- Mã hóa: Mỗi ký tự P được mã hóa thành C theo công thức: C=(aP+b)(mod26).
- Trong đó: a,b là khóa (khóa a phải là số nguyên tố cùng nhau với 26)
- Giải mã: Để giải mã, ta cần tìm nghịch đảo của a modulo 26, ký hiệu là a^−1.
  + P=a^−1(C−b)(mod26)
  + Lưu ý: a^−1 phải thỏa mãn (a×a^−1)(mod26)=1.

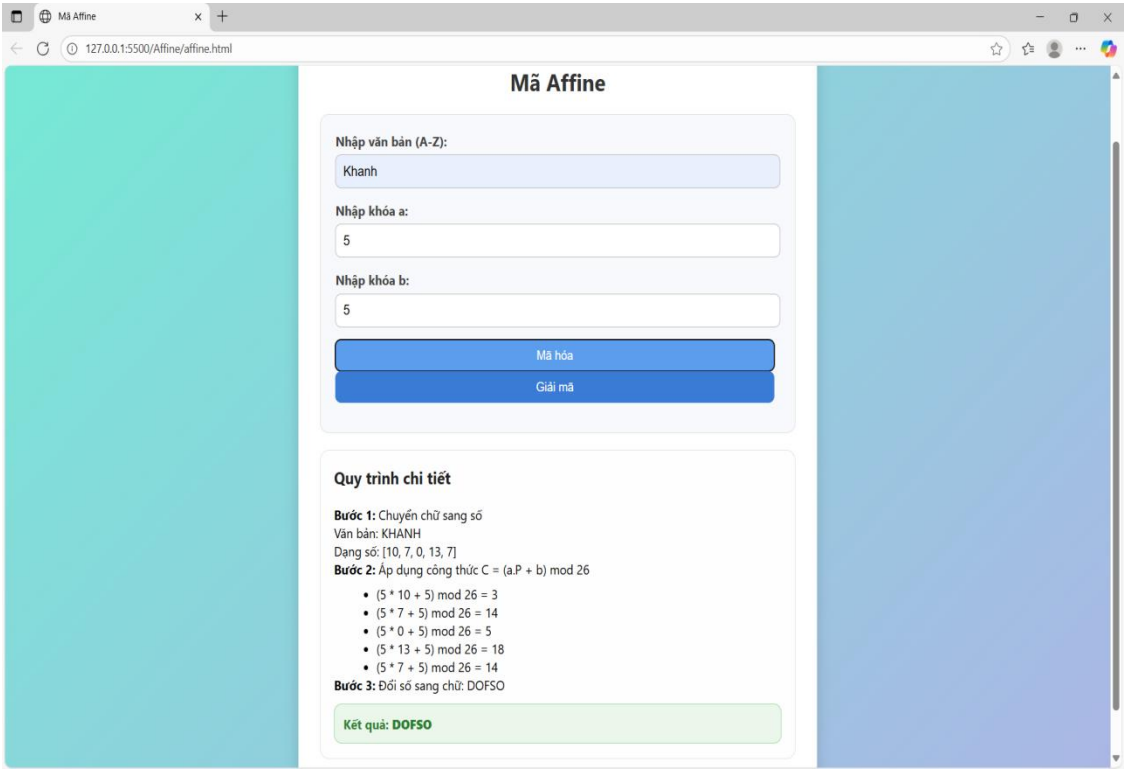
❖ Không gian khóa: Không gian khóa của mã Affine lớn hơn Caesar. Khóa a có 12 giá trị có thể (1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25) và khóa b có 26 giá trị có thể (0-25). Tổng số khóa là 12×26=312 khóa.

❖ Cách phá mã:

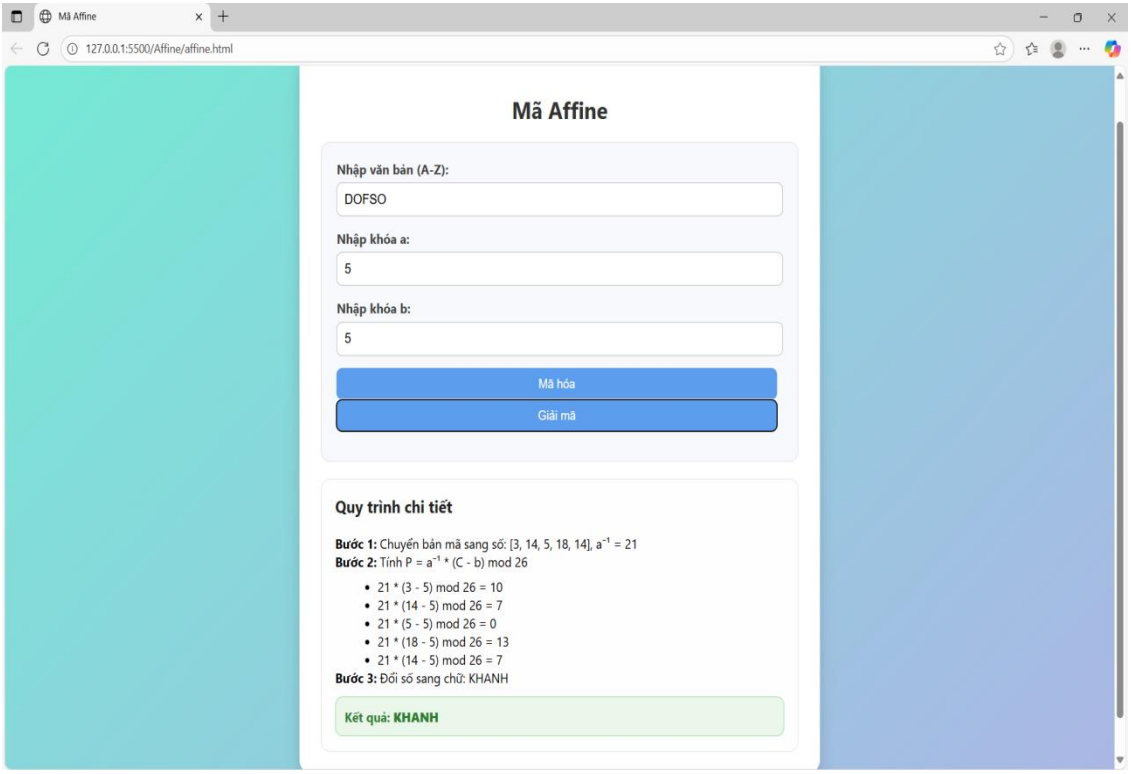
- Tìm nghịch đảo modulo: Tìm a^{-1}.

- Thử mọi khóa (bất kỳ): Thử tất cả các cặp (a, b) phù hợp với điều kiện gcd(a, m) = 1.

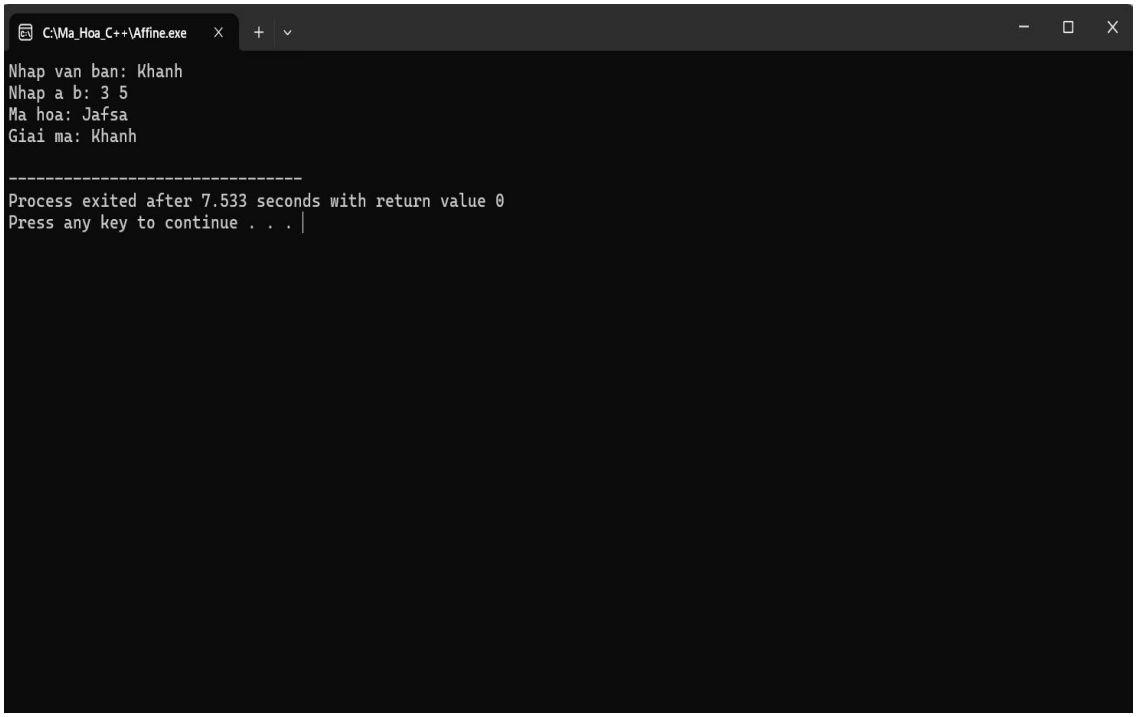
❖ Mã hóa:



❖ Giải mã:



❖ C++:



1.3. Phương pháp mã hóa hoán vị

❖ Tên gọi: Columnar Transposition (Mã hoán vị cột) — viết plaintext theo hàng vào một lưới có số cột = độ dài khóa, rồi đọc cột theo thứ tự khóa đã được sắp xếp.

❖ Thuật toán:

➢ Thuật toán mã hoá:

- Khóa: một từ (ví dụ "ZEBRA") — độ dài m.
- Các bước:

+ Loại bỏ/giữ spaces theo cách bạn muốn (thường bỏ spaces để đơn giản).

+ Viết plaintext theo hàng vào bảng có m cột (độ dài khóa). Nếu cần, padding ký tự (ví dụ 'X') để đầy hàng cuối.

+ Gán thứ tự cho các cột bằng cách sắp chữ cái khóa theo bảng chữ cái, nếu trùng chữ thì dùng chỉ số xuất hiện.

+ Đọc ciphertext theo cột theo thứ tự tăng dần của thứ tự gán (tức cột có chữ nhỏ nhất trước...).

➢ Thuật toán giải mã:

- Biết độ dài khóa m → bạn biết số cột. Từ chiều dài ciphertext tính số hàng rows = ceil(len(cipher)/m).
- Dựa vào thứ tự khóa, phân chia ciphertext thành từng cột theo thứ tự đó (mảng cột có rows ký tự mỗi cột, có thể cột cuối thiếu ký tự nếu padding không đầy).
- Sau đó dựng lại bảng hàng xột và đọc theo hàng để lấy plaintext (loại padding nếu có).

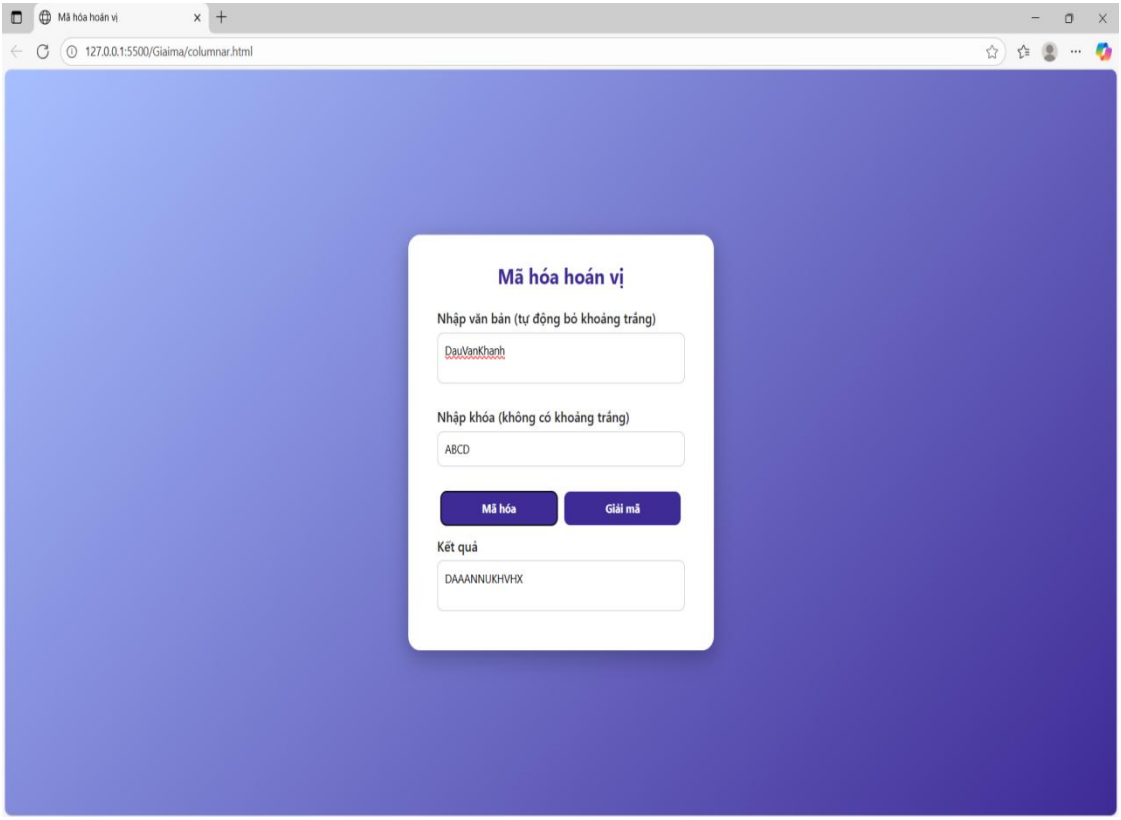
❖ Không gian khóa:

Nếu khóa dài m và ký tự khác nhau, có m! hoán vị khả dĩ — không gian tăng rất nhanh. Nhưng nếu khóa là từ tự nhiên thì không gian thực tế nhỏ hơn.

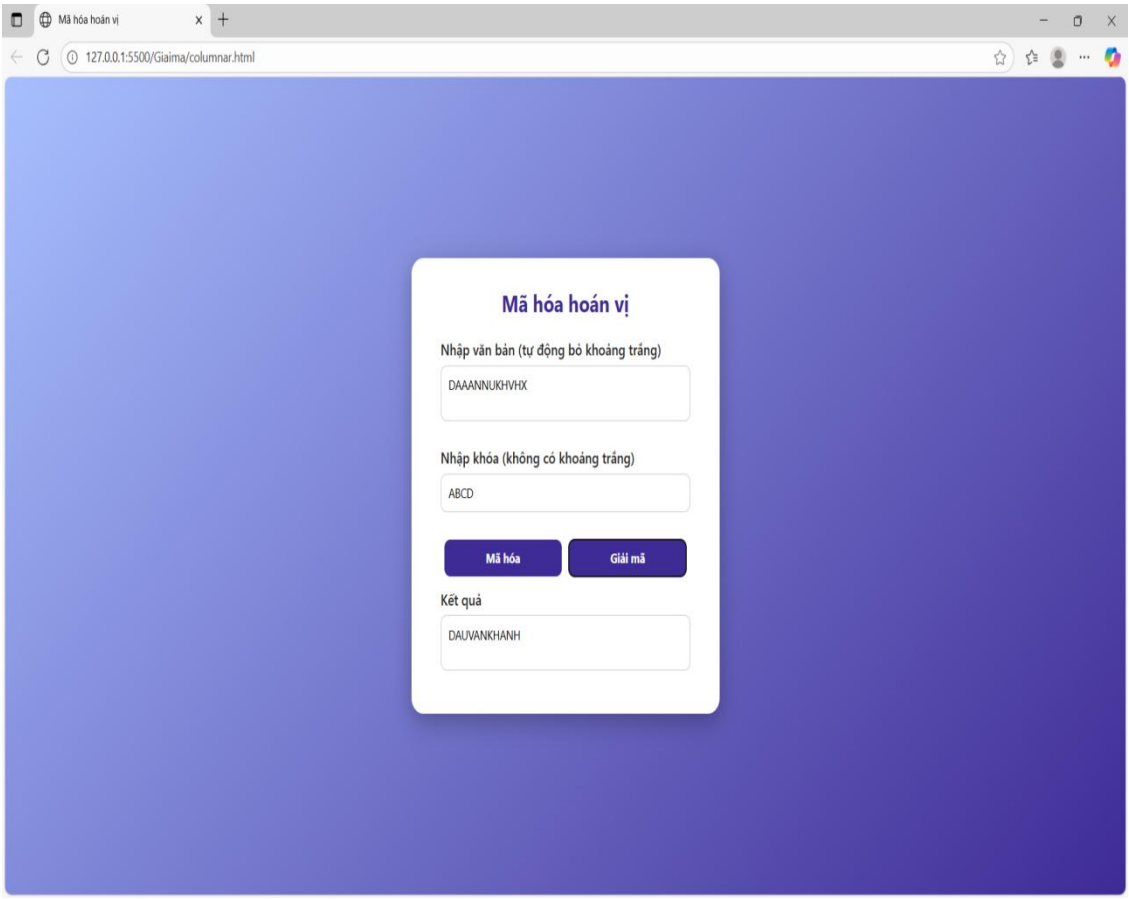
❖ Cách phá mã (không cần khoá):

- Brute-force: thử tất cả m! hoán vị — chỉ khả thi khi m nhỏ (ví dụ m ≤ 8).
- Kết hợp với scoring n-gram: dùng thuật toán tìm kiếm trên không gian hoán vị, tính điểm bằng n-gram (bigrams/trigrams) để chọn hoán vị hợp lý.
- Known-plaintext / crib: nếu biết đoạn plaintext xuất hiện ở đâu, dùng để tìm hoán vị phù hợp.
- Phân tích cấu trúc: dựa vào vị trí khoảng trắng/điểm, tần suất chữ, độ dài từ để suy.

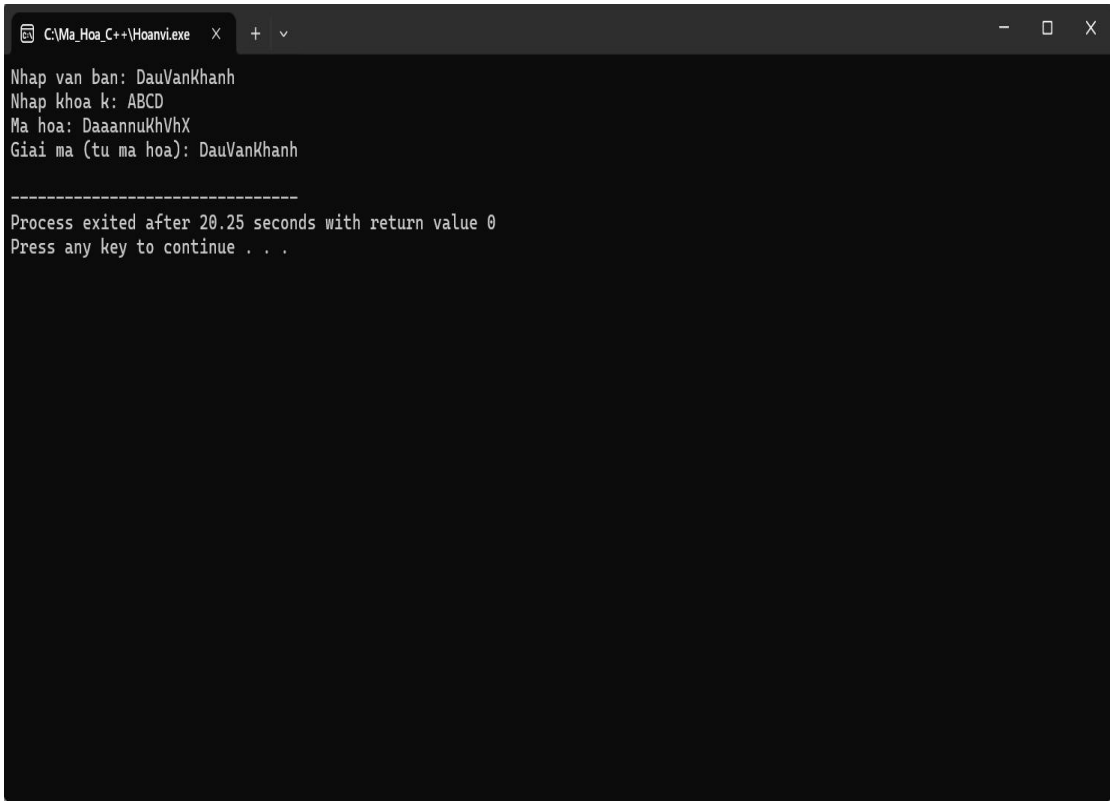
❖ Mã hóa:


Mã hóa hoán vị
127.0.0.1:5500/Giaima/columnar.html
Mã hóa hoán vị
Nhập văn bản (tự động bỏ khoảng trắng)
DauVanKhanh
Nhập khóa (không có khoảng trắng)
ABCD
Mã hóa
Giải mã
Kết quả
DAAANNUKHVHX

❖ Giải mã:


Mã hóa hoán vị
127.0.0.1:5500/Giaima/columnar.html
Mã hóa hoán vị
Nhập văn bản (tự động bỏ khoảng trắng)
DAAANNUKHVHX
Nhập khóa (không có khoảng trắng)
ABCD
Mã hóa
Giải mã
Kết quả
DAUVANKHANH

❖ C++:



1.4. Phương pháp mã hóa Vigenère

❖ Tên gọi: Vigenère cipher — polyalphabetic substitution dùng chuỗi khóa lặp lại để đổi bảng Caesar theo từng vị trí.

❖ Thuật toán:

➢ Thuật toán mã hoá:

- Khóa: chuỗi ký tự K dài m.
- Công thức:
  + Với plaintext ký tự Pi(0...25) và key ký tự Kj với j=i mod m: Ci = (Pi + Kj) mod 26.
- Các bước:
  + Chuẩn hoá: thường dùng chữ hoa/không dấu.
  + Lặp lại key để khớp chiều dài plaintext.
  + Với từng ký tự, cộng giá trị key rồi mod 26.

+ Thuật toán giải mã: Ci = (Pi + Kj) mod 26

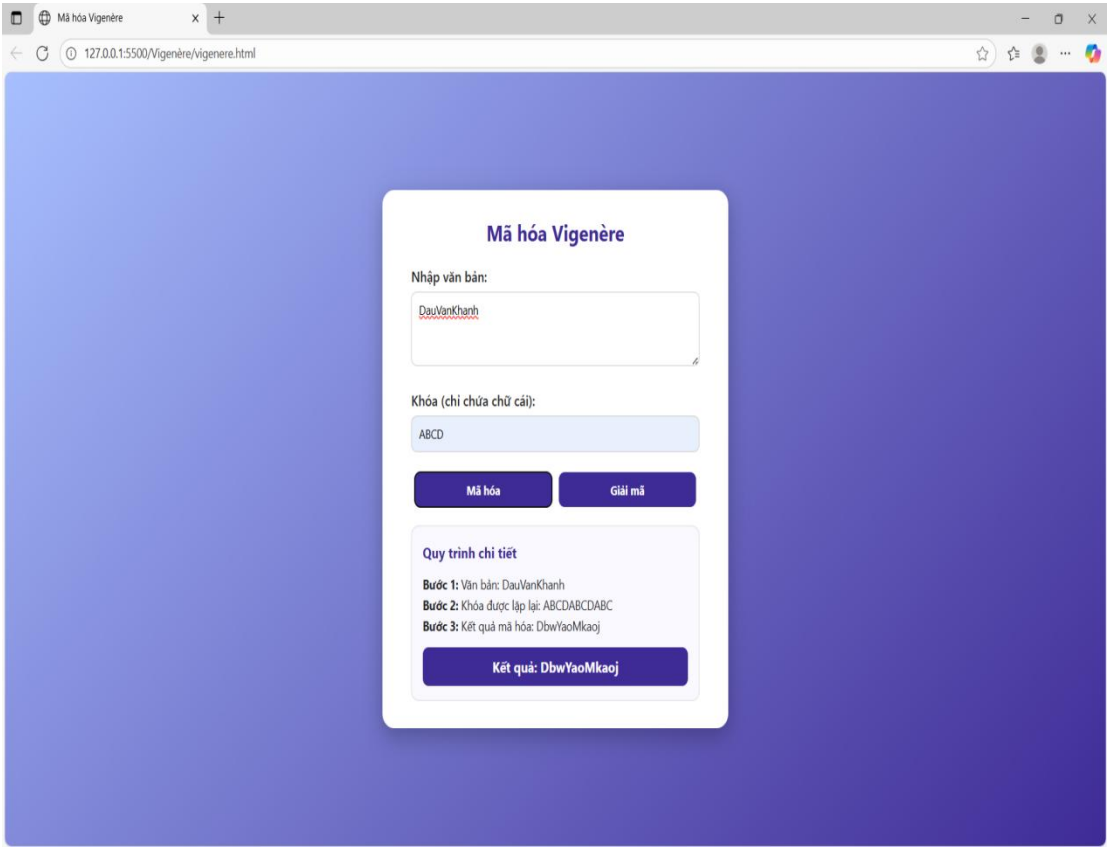
❖ Không gian khóa:

Nếu key độ dài m và mỗi ký tự có 26 khả năng: 26^m. Tuy nhiên, nếu key lấy từ từ ngôn ngữ tự nhiên, không gian hiệu dụng nhỏ hơn.

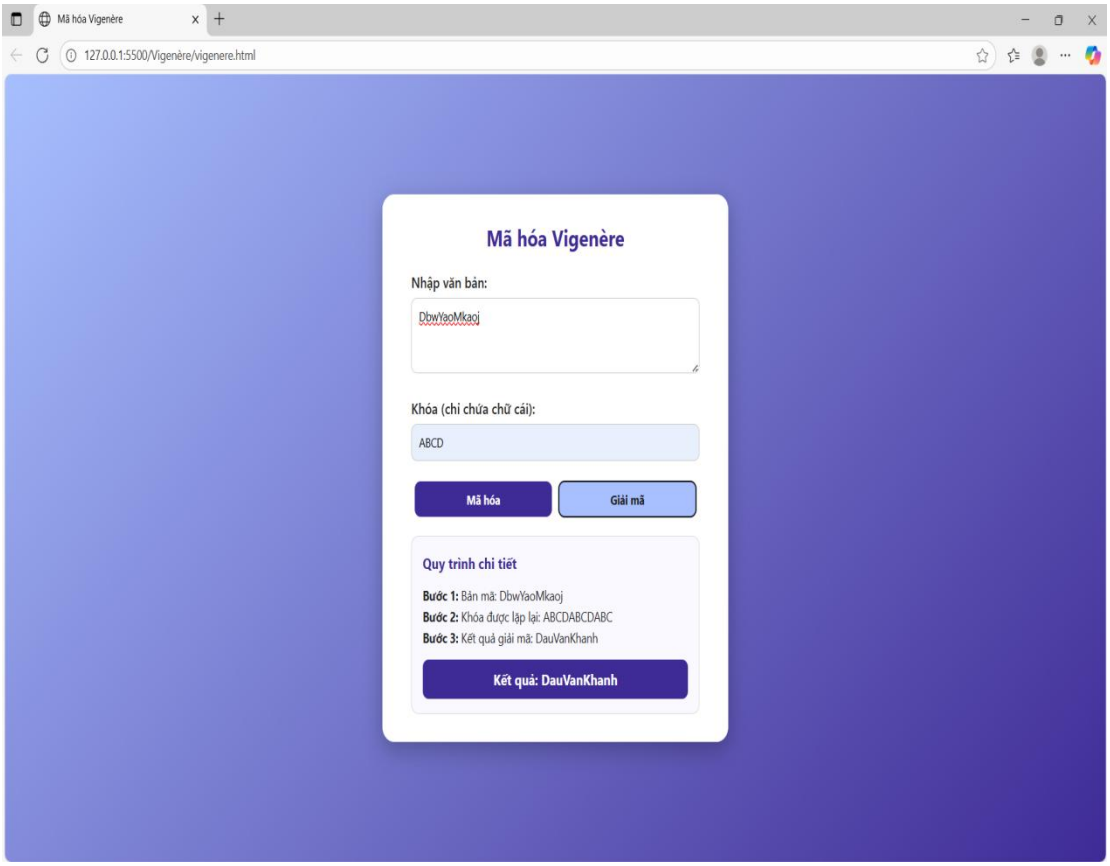
❖ Cách phá mã (không cần khoá):

- Kasiski examination: tìm các chuỗi trùng lặp (tri-gram, 4-gram) trong ciphertext, đo khoảng cách giữa các lần lặp → ước đoán độ dài key m (ước số chung của các khoảng cách).
- Friedman test (Index of Coincidence): ước lượng độ dài key bằng tính IC (chỉ số trùng hợp).
- Khi biết m: chia ciphertext thành m dãy (các ký tự tương ứng cùng vị trí modulo m), mỗi dãy là Caesar cipher → áp dụng phân tích tần suất trên từng dãy để tìm shift của từng dãy → xây lại key.
- Kết hợp: sử dụng scoring n-gram để chọn độ dài key và key tốt nhất.

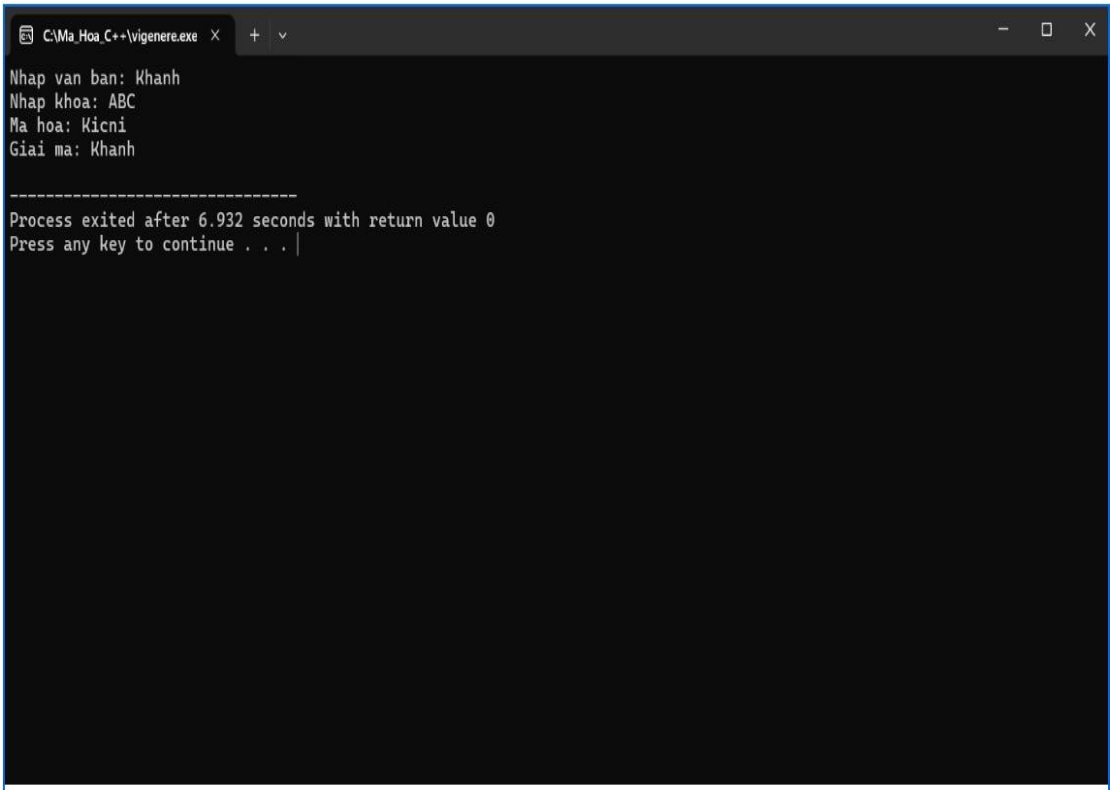
❖ Mã hóa:



❖ Giải mã:


Mã hóa Vigenère
127.0.0.1:5500/Vigenère/vigenere.html
Mã hóa Vigenère
Nhập văn bản:
DbwYaoMkaoj
Khóa (chỉ chứa chữ cái):
ABCD
Mã hóa
Giải mã
Quy trình chi tiết
Bước 1: Bản mã: DbwYaoMkaoj
Bước 2: Khóa được lặp lại: ABCDABCDABC
Bước 3: Kết quả giải mã: DauVanKhanh
Kết quả: DauVanKhanh

❖ C++:


C:\Ma_Hoa_C++\vigenere.exe
Nhap van ban: Khanh
Nhap khoa: ABC
Ma hoa: Kicni
Giai ma: Khanh

Process exited after 6.932 seconds with return value 0
Press any key to continue . . .

1.5. Phương pháp mã hóa Playfair

❖ Tên gọi: Playfair cipher — substitution cipher làm việc theo cặp ký tự (digraph), sử dụng một ma trận 5×5 (tổng 25 chữ cái, thường gộp I và J).

❖ Thuật toán:

➢ Thuật toán mã hóa:

- Tạo key square 5×5:
  + Viết khóa (loại chữ trùng, gộp J → I) theo thứ tự xuất hiện.
  + Sau đó điền các chữ cái còn lại (A..Z, bỏ J) theo thứ tự.
- Chuẩn bị plaintext:
  + Loại bỏ ký tự không phải chữ (tuỳ chọn), chuyển sang chữ hoa.
  + Gộp J thành I.
  + Chia plaintext thành digraphs (cặp 2 ký tự):
    - Nếu trong cặp 2 ký tự giống nhau (ví dụ "AA"), chèn X (hoặc Q) giữa, rồi tiếp tục phân chia.
    - Nếu chuỗi lẻ → thêm X vào cuối.
- Mã hoá mỗi digraph (A,B):
  + Nếu A và B ở cùng hàng: thay mỗi chữ bằng chữ bên phải của nó (vòng lại).
  + Nếu A và B ở cùng cột: thay mỗi chữ bằng chữ bên dưới của nó (vòng lại).
  + Nếu khác hàng & cột: thay mỗi chữ bằng chữ cùng hàng nhưng lấy cột của chữ kia (tạo hình chữ nhật). Nối các digraph đã biến đổi thành ciphertext.

➢ Thuật toán giải mã:

- Tương tự thuật toán mã hóa nhưng:
  + Nếu cùng hàng: thay bằng chữ bên trái.

+ Nếu cùng cột: thay bằng chữ bên trên.

+ Nếu khác: đảo lại các cột như lúc mã hoá.

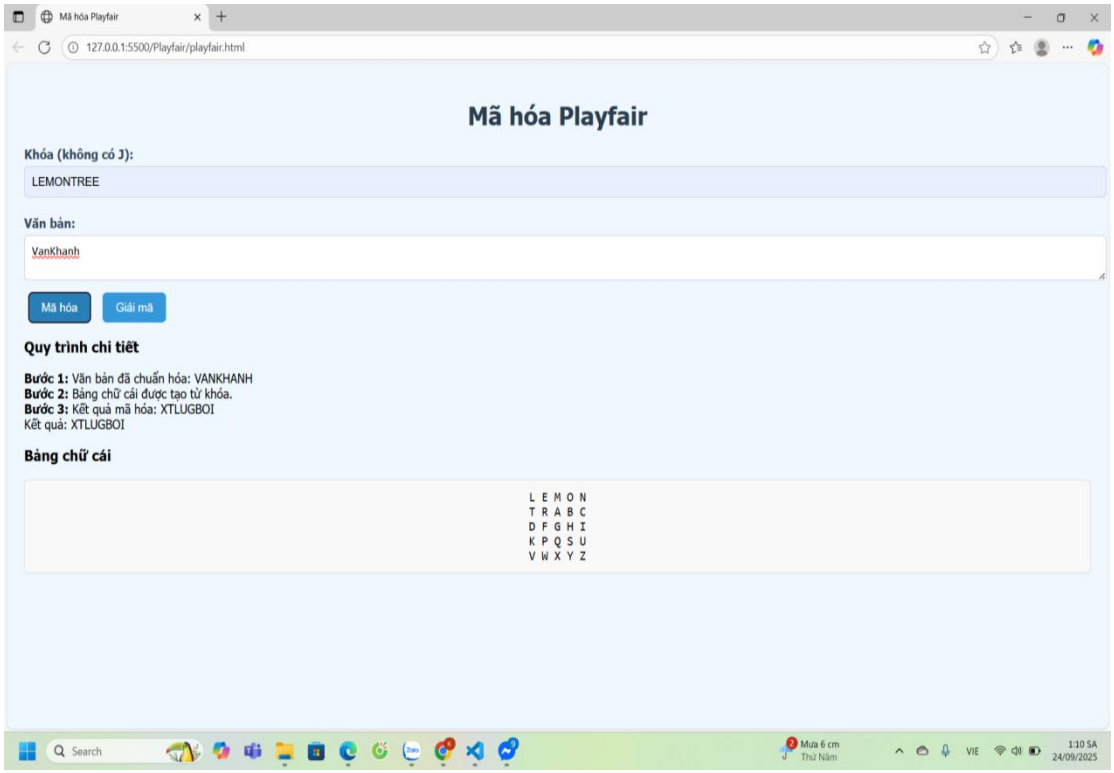
❖ Không gian khóa:

- Không gian lớn (rất nhiều ma trận 5×5 khả dĩ), nhưng thực tế khóa thường là chuỗi từ ngôn ngữ → không gian thực lựa chọn nhỏ hơn. Không brute-force toàn bộ 25! được.

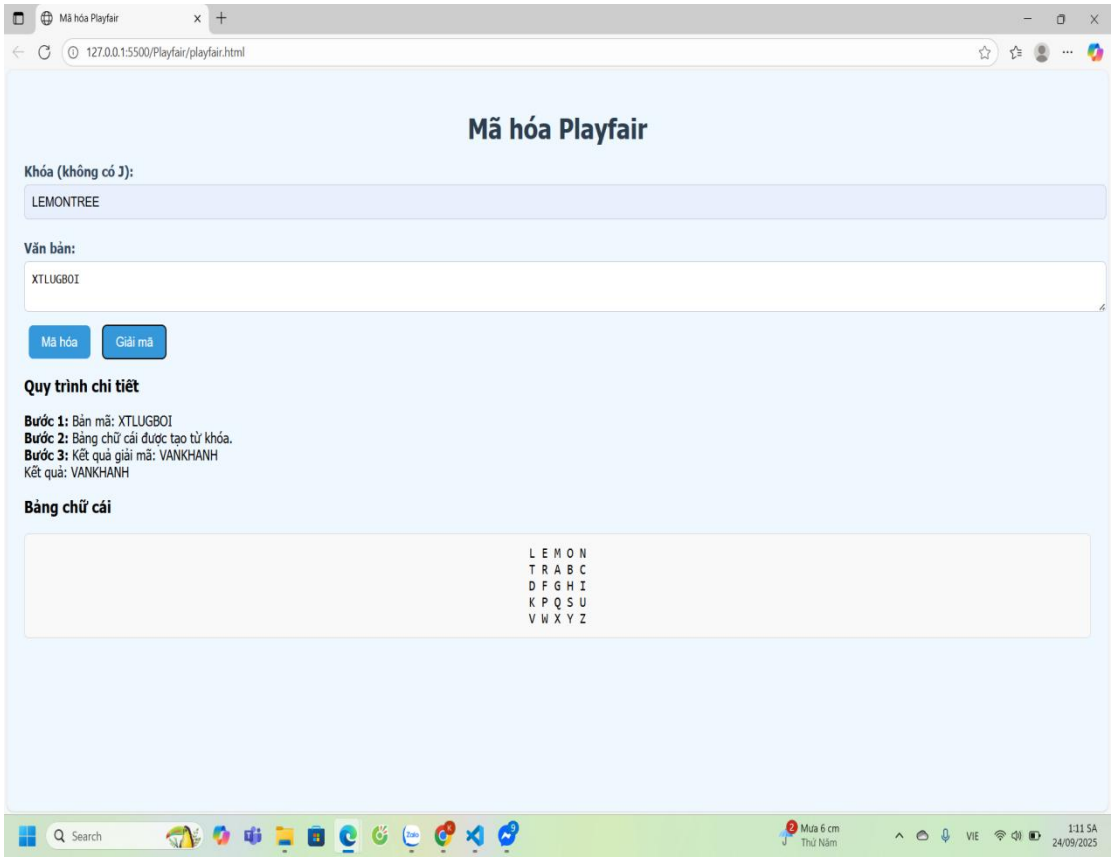
❖ Cách phá mã:

- Phân tích digraph: so sánh tần suất digraphs, bigram scoring.
- Heuristic search: hill-climbing, simulated annealing, genetic algorithm trên không gian các ma trận 5×5 — tối ưu scoring n-gram (English fitness) để tìm ma trận khả dĩ.
- Known-plaintext: nếu biết vài cặp digraph → rút ra cấu trúc key square.
- Vì Playfair mã hóa theo digraph nên phân tích tần suất đơn ký tự ít hữu dụng hơn; phải dùng digraph/quadgram.

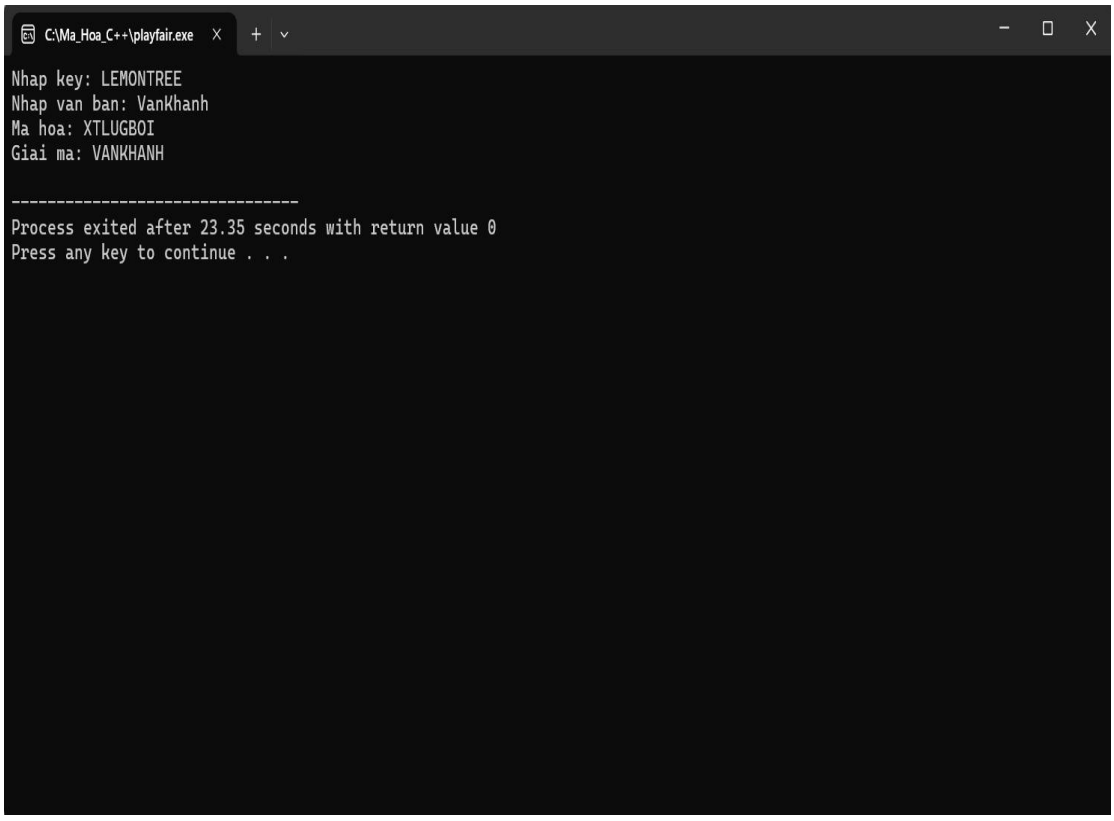
❖ Mã hóa:



❖ Giải mã:


Mã hóa Playfair
127.0.0.1:5500/Playfair/playfair.html
Mã hóa Playfair
Khóa (không có J):
LEMONTREE
Văn bản:
XTLUGBOI
Mã hóa
Giải mã
Quy trình chi tiết
Bước 1: Bản mã: XTLUGBOI
Bước 2: Bảng chữ cái được tạo từ khóa.
Bước 3: Kết quả giải mã: VANKHANH
Kết quả: VANKHANH
Bảng chữ cái
L E M O N
T R A B C
D F G H I
K P Q S U
V W X Y Z
Search
Mưa 6 cm
Thứ Năm
VIE
1:11 SA
24/09/2025

❖ C++:


C:\Ma_Hoa_C++\playfair.exe
Nhap key: LEMONTREE
Nhap van ban: VanKhanh
Ma hoa: XTLUGBOI
Giai ma: VANKHANH

Process exited after 23.35 seconds with return value 0
Press any key to continue . . .

BÀI TẬP 2: CHỮ KÝ SỐ TRONG FILE PDF

ĐỀ BÀI:

1) Cấu trúc PDF liên quan chữ ký (Nghiên cứu)

- Mô tả ngắn gọn: Catalog, Pages tree, Page object, Resources, Content streams, XObject, AcroForm, Signature field (widget), Signature dictionary (/Sig), /ByteRange, /Contents, incremental updates, và DSS (theo PAdES).
- Liệt kê object refs quan trọng và giải thích vai trò của từng object trong lưu/truy xuất chữ ký.
- Đầu ra: 1 trang tóm tắt + sơ đồ object (ví dụ: Catalog → Pages → Page → /Contents
; Catalog → /AcroForm → SigField → SigDict).

2) Thời gian ký được lưu ở đâu?

- Nêu tất cả vị trí có thể lưu thông tin thời gian:

+ /M trong Signature dictionary (dạng text, không có giá trị pháp lý).

+ Timestamp token (RFC 3161) trong PKCS#7 (attribute timeStampToken).

+ Document timestamp object (PAdES).

+ DSS (Document Security Store) nếu có lưu timestamp và dữ liệu xác minh.

- Giải thích khác biệt giữa thông tin thời gian /M và timestamp RFC3161.

3) Các bước tạo và lưu chữ ký trong PDF (đã có private RSA)

- Viết script/code thực hiện tuần tự:

1. Chuẩn bị file PDF gốc.
2. Tạo Signature field (AcroForm), reserve vùng /Contents (8192 bytes).
3. Xác định /ByteRange (loại trừ vùng /Contents khỏi hash).
4. Tính hash (SHA-256/512) trên vùng ByteRange.

5. Tạo PKCS#7/CMS detached hoặc CAdES:

- Include messageDigest, signingTime, contentType.

- Include certificate chain.

- (Tùy chọn) thêm RFC3161 timestamp token.

6. Chèn blob DER PKCS#7 vào /Contents (hex/binary) đúng offset.

7. Ghi incremental update.

8. (LTV) Cập nhật DSS với Certs, OCSPs, CRLs, VRI.

- Phải nêu rõ: hash alg, RSA padding, key size, vị trí lưu trong PKCS#7.

- Đầu ra: mã nguồn, file PDF gốc, file PDF đã ký.

4) Các bước xác thực chữ ký trên PDF đã ký

- Các bước kiểm tra:

1. Đọc Signature dictionary: /Contents, /ByteRange.

2. Tách PKCS#7, kiểm tra định dạng.

3. Tính hash và so sánh messageDigest.

4. Verify signature bằng public key trong cert.

5. Kiểm tra chain → root trusted CA.

6. Kiểm tra OCSP/CRL.

7. Kiểm tra timestamp token.

8. Kiểm tra incremental update (phát hiện sửa đổi).

- Nộp kèm script verify + log kiểm thử.

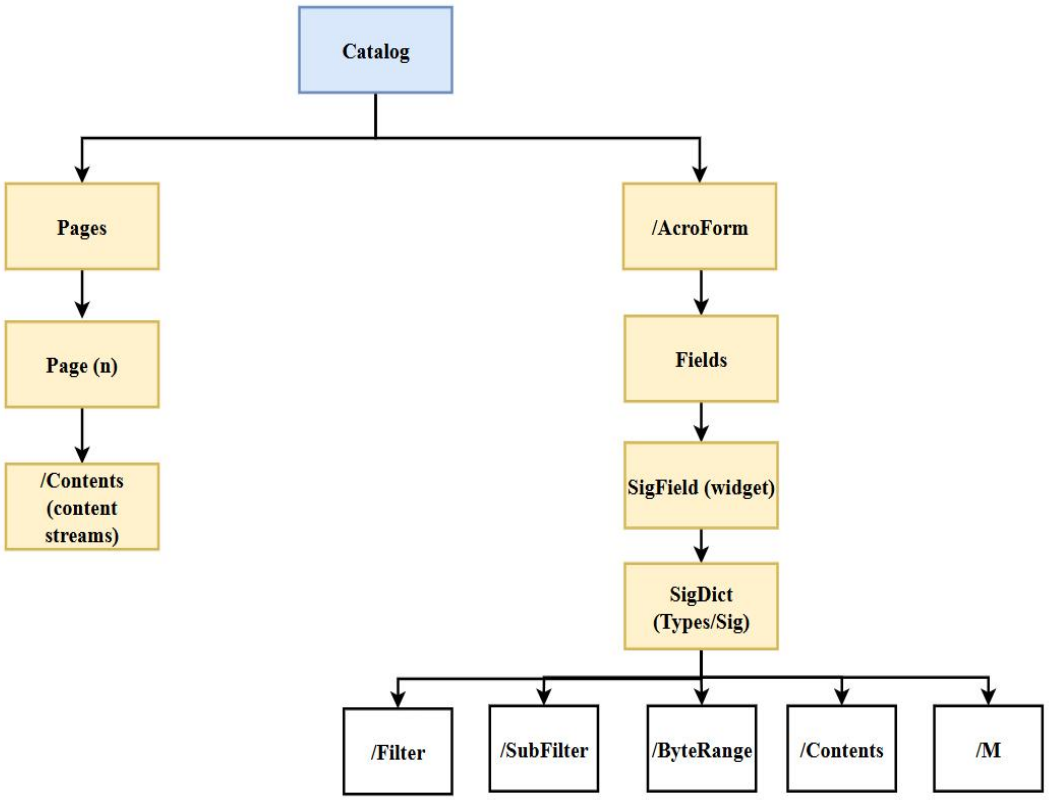
BÀI LÀM

2.1. Cấu trúc PDF liên quan chữ ký:

❖ Mô tả ngắn gọn các object:

- Catalog: root object của tài liệu. Thường tham chiếu tới /Pages và /AcroForm.
- Pages tree / Page object: chứa /Contents (content streams) và resources.
- Resources / Content streams / XObject: nội dung hiển thị.
- AcroForm: form-level dictionary; chứa Fields (form fields) và có thể có /SigFlags.
- Signature field (widget): một AcroForm field loại Sig (field dictionary) và widget annotation trên trang để hiển thị khung chữ ký.
- Signature dictionary (/Sig): Đối tượng chứa thông tin chữ ký, gồm: /Type /Sig, /Filter, /SubFilter (kiểu chữ ký), /ByteRange, /Contents (blob chữ ký PKCS#7), /M (thời gian ký), /Name, /Location.
- /ByteRange: mảng bốn số [start1 length1 start2 length2] xác định hai vùng dữ liệu trong file được hash (vùng giữa chứa /Contents placeholder).
- /Contents: chỗ chứa blob chữ ký (DER PKCS#7), thường được reserve kích thước cố định (ví dụ 8192 bytes) trong incremental update.
- Incremental updates: PDF có thể bổ sung object mới và xref bổ sung, cho phép chèn chữ ký mà không sửa nội dung cũ - cơ chế này cho phép phát hiện sửa đổi sau khi ký.
- DSS (Document Security Store): Lưu thông tin xác thực dài hạn (LTV) như chứng thư, OCSP/CRL và timestamp tokens.

❖ Sơ đồ object:



❖ Object refs quan trọng & vai trò:

- /AcroForm (Root obj): chứa danh sách field — cần để trình đọc biết chỗ signature fields.

- SigField (Field obj): định danh trường chữ ký (tên, vị trí hiển thị); widget annotation trên trang liên hệ tới form field này.

- SigDict (Signature dictionary): chứa metadata chữ ký và blob PKCS#7 trong /Contents — đây là object chính để lưu chữ ký.

- /ByteRange: thiết lập vùng file để compute digest (loại trừ vùng /Contents).

- Incremental update xref + trailer: nếu có thay đổi sau chữ ký, xref/trailer mới sẽ khác → phát hiện sửa đổi.

2.2. Thời gian ký được lưu ở đâu?

❖ Vị trí có thể lưu thông tin thời gian:

- /M trong Signature dictionary (dạng text, ví dụ D:20251026...). Không có giá trị pháp lý — chỉ là metadata.

- Timestamp token (RFC 3161) được nhúng trong PKCS#7/CMS (attribute timeStampToken hoặc separate attribute id-aa-signatureTimeStampToken trong CAdES). Timestamp RFC3161 do một TSA ký trên digest giúp chứng thực thời điểm.

- Document timestamp object (PAdES) — PDF có thể có đối tượng timestamp document-level (khác với signature field của signer) theo PAdES.

- DSS (Document Security Store): lưu trữ timestamp tokens (.tsr), chứng thư, OCSP/CRL responses phục vụ LTV.

❖ Khác biệt chính /M vs RFC3161 timestamp:

- /M: chỉ là chuỗi định dạng ngày giờ do signer ghi vào dictionary; có thể bị giả mạo (không được ký độc lập). Không đủ cho chứng thực thời điểm.

- RFC3161 timestamp: do một Time Stamping Authority (TSA) ký trên digest của PKCS#7/CMS (hoặc của dữ liệu), do đó là bằng chứng thời điểm độc lập và đáng tin cậy (nếu tin tưởng TSA).

2.3. Rủi ro bảo mật chữ ký số trong PDF

❖ **Tổng quan ngắn:** Chữ ký số trên PDF có thể rất an toàn nếu triển khai theo chuẩn (ví dụ PAdES) và thực hiện kiểm tra xác thực đầy đủ. Tuy nhiên có nhiều lớp rủi ro: từ việc sửa đổi nội dung PDF, dùng cập nhật gia tăng (incremental updates) để che thay đổi, đến sự yếu kém hoặc bị lộ của chứng thư và khoá riêng. Dưới đây tóm tắt các rủi ro chính, cách phát hiện và biện pháp giảm nhẹ.

❖ **Rủi ro bảo mật và giải pháp:**

- Incremental update lạm dụng: Cơ chế cập nhật gia tăng có thể bị lợi dụng để chèn thay đổi mà không phá chữ ký, khiến tài liệu bị chỉnh sửa nhưng vẫn trông hợp lệ.

→ Giải pháp: hạn chế incremental updates; kiểm tra modification_level và lịch sử revision.

- Metadata bị thay đổi: Metadata như tác giả, thời gian tạo, thời gian sửa, tiêu đề… nằm ngoài nội dung trực tiếp nên đôi khi không được bao phủ bởi chữ ký. Điều này cho phép kẻ xấu thay đổi metadata mà không làm chữ ký bị phát hiện sai.

  → Giải pháp: ký cả nội dung và metadata; tuân thủ chuẩn PAdES.

- Thuật toán mật mã yếu: Một số hệ thống vẫn dùng thuật toán cũ như SHA-1 hoặc khóa RSA 1024-bit, vốn không còn an toàn. Điều này tạo cơ hội cho việc tấn công va chạm hoặc dò khóa.

  → Giải pháp: dùng SHA-256/ SHA-3, RSA ≥ 2048-bit hoặc ECC; áp dụng policy ký chuẩn.

- Tampering nội dung PDF: Tin tặc có thể thao tác trực tiếp vào cấu trúc PDF như sửa vùng /Contents (chứa chữ ký) hoặc ByteRange để che khu vực đã bị chỉnh sửa. Nếu phần mềm xem PDF không kiểm tra đầy đủ, tampering có thể không bị phát hiện.

  → Giải pháp: kiểm tra toàn bộ byte-range của tài liệu; sử dụng trình xác thực tuân thủ PAdES để đảm bảo kiểm tra đầy đủ mọi vùng nội dung.

- Replay / incremental-update attack: Tái sử dụng chữ ký hoặc timestamp cũ để gắn lên tài liệu đã bị chỉnh sửa.

  → Giải pháp: kiểm tra revision; xác thực timestamp RFC 3161; dùng PAdES-DSS.

- Time repudiation: Timestamp giả hoặc không đáng tin làm giảm giá trị pháp lý của chữ ký.

  → Giải pháp: sử dụng TSA uy tín; lưu và xác thực TimeStampToken (TST).

- Chứng thư hết hạn/thu hồi: Không kiểm tra CRL/OCSP khiến chữ ký vẫn hiển thị hợp lệ dù chứng thư đã thu hồi hoặc hết hạn.

  → Giải pháp: kiểm tra CRL/OCSP; nhúng thông tin xác thực theo PAdES-DSS (LTV).

- Lộ khóa riêng / side-channel: Khóa ký bị đánh cắp cho phép ký giả mạo tài liệu hợp lệ.

  → Giải pháp: dùng HSM/TPM hoặc USB Token an toàn; kiểm soát truy cập và nhật ký.

- Vấn đề vận hành & cấu hình: Cấu hình sai, thiếu logging hoặc không kiểm thử bảo mật dẫn đến rủi ro tiềm ẩn.

  → Giải pháp: tuân thủ PAdES; audit, logging và kiểm thử bảo mật định kỳ.

2.4. Kết luận:

Bài tập giúp em hiểu và thực hành quy trình tạo, nhúng và xác thực chữ ký số trong file PDF theo chuẩn PDF/PAdES, qua đó nắm rõ cấu trúc tài liệu, vị trí lưu chữ ký và thời gian ký, cùng cách sử dụng RSA, SHA-256 và PKCS#7 để đảm bảo tính toàn vẹn, xác thực và chống giả mạo.

Chữ ký số PDF duy trì tính toàn vẹn thông qua các thành phần AcroForm, SigDict và ByteRange, còn thông tin thời gian được lưu bằng thuộc tính /M hoặc token RFC3161. Để tăng độ tin cậy và ngăn ngừa nguy cơ sửa đổi hay giả mạo chứng thư, việc mở rộng xác minh bằng DSS và chuẩn PAdES nâng cao là cần thiết. Thử nghiệm các tệp original.pdf, signed.pdf, tampered.pdf cho thấy quy trình ký và kiểm chứng hoạt động chính xác, minh chứng cho tính hiệu quả của phương pháp.

KẾT LUẬN

Qua quá trình thực hiện bài tập lớn, em đã hiểu rõ nền tảng của hai lĩnh vực quan trọng trong bảo mật thông tin: mật mã cổ điển và chữ ký số trong tài liệu PDF. Các phương pháp mã hóa như Caesar, Affine, Hoán vị, Vigenère và Playfair đã giúp em nắm được tư duy cơ bản về thiết kế thuật toán, không gian khóa và kỹ thuật phân tích, tạo tiền đề cho việc tiếp cận các hệ mật mã hiện đại.

Phần tìm hiểu về chữ ký số mang lại góc nhìn toàn diện hơn về cơ chế đảm bảo tính toàn vẹn và tính xác thực của tài liệu điện tử. Thông qua việc nghiên cứu cấu trúc PDF, vai trò của các đối tượng như Signature Dictionary, ByteRange, Contents và cơ chế incremental update, em đã hiểu rõ quy trình hình thành chữ ký số cũng như cách thức kiểm tra, phát hiện thay đổi. Việc tiếp cận chuẩn PKCS#7/CMS, RFC3161 và PAdES giúp em nhận thức sâu sắc hơn về giá trị pháp lý và mức độ an toàn của chữ ký số trong thực tiễn.

Bài tập không chỉ củng cố kiến thức lý thuyết mà còn hỗ trợ em rèn luyện kỹ năng thực hành, tư duy bảo mật và khả năng phân tích kỹ thuật. Đây là nền tảng quan trọng để em tiếp tục nghiên cứu sâu hơn về an toàn thông tin và các hệ thống bảo mật hiện đại.

TÀI LIỆU THAM KHẢO

1. William Stallings, "Cryptography and Network Security: Principles and Practice", 8th Edition, Pearson, 2023.
2. Bài giảng Mật mã học cơ bản – Caesar, Affine, Playfair, Vigenère, Bộ môn Toán – Tin, Đại học Sư phạm Hà Nội, 2020.
3. Giáo trình An toàn và bảo mật thông tin, học viện Công nghệ Bưu chính viễn thông.
4. Bài giảng Chữ ký số & PKI, Học viện Công nghệ Bưu chính Viễn thông (PTIT), 2022.
5. Bernd Zipperer, "Digital Signatures in PDF Documents", iText Software, 2021. (Tài liệu hướng dẫn cấu trúc chữ ký trong PDF).
6. Paar, C. & Pelzl, J., "Understanding Cryptography", Springer, 2010. (Mã hóa cổ điển, Vigenère, Affine, Playfair).

MÃ QR GITHUB

BÀI TẬP 1

BÀI TẬP 2