BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI

BÀI TẬP LỚN

MÔN: AN TOÀN VÀ BẢO MẬT THÔNG TIN

TÊN ĐỀ TÀI BTL

Tìm hiểu về chữ ký điện tử Elgamal và viết ứng dụng minh họa

CBHD: ***ThS. Trần Phương Nhung***

Nhóm: 13

Thành viên nhóm: Ngô Văn Sáng - 2018603391

Hoàng Minh Quý - 2018601167

Nguyễn Nhật Quang - 2020608641

Vũ Hồng Phương – 2020601638

Hà Nội – Năm 2021

Chương 1. TỔNG QUAN 1
I. TỔNG QUÁT VỀ ĐỀ TÀI NGHIÊN CỨU 1
II. NỘI DUNG NGHIÊN CỨU 1
1. Chữ kí điện tử, chữ kí điện tử Elgamal 1
1.1. Chữ kí điện tử 1
1.2. Sơ đồ chữ ký Elgamal 2
2. Tìm hiểu về hàm băm SHA. 2
3. Tìm hiểu phương pháp mã hóa bất đối xứng ứng dụng trong chữ ký điện tử. 2
4. Triển khai cụ thể với ngôn ngữ 2
III. CÁC KIẾN THỨC 3
1. Hướng nghiên cứu đề tài 3
2. Đối tượng nghiên cứu 3
3. Phương pháp nghiên cứu đề tài 3
IV. CHỦ ĐỀ NGHIÊN CỨU 3
Chương 2. KẾT QUẢ NGHIÊN CỨU 4
I. GIỚI THIỆU 4
II. NỘI DUNG THUẬT TOÁN 5
1. Tạo khóa 5
2. Mã hóa 5
3. Giải mã 5
III. THIẾT KẾ, CÀI ĐẶT CHƯƠNG TRÌNH ĐỀ MÔ THUẬT TOÁN 6
1. Giao diện chương trình demo 6
1.1. Giao diện màn hình Tạo khóa 6
1.2. Giao diện màn hình kí văn bản 7
1.3. Giao diện màn hình Xác nhận chữ kí 7
IV. CÀI ĐẶT VÀ TRIỂN KHAI 8
1. Giới thiệu về PHP 8
2. Giới thiệu về C# 9
3. Giới thiệu về C++ 9
4. Giới thiệu về Python 10
V. THỰC HIỆN BÀI TOÁN 11
1. Phân công công việc 11

2. Hoàng Minh Quý - Tìm hiểu về chữ ký điện tử, chữ ký điện tử Elgamal. 11
2.1. Chữ kí điện tử, chữ kí điện tử Elgamal 11
2.1.1. Chữ kí điện tử 11
2.1.2. Các thành phần của chữ ký điện tử: 11
2.1.3. Các thành phần của sơ đồ chữ ký điện tử. 12
2.1.4. Chức năng của chữ ký điện tử. 13
2.1.5. Các ứng dụng về chữ ký điện tử trên thế giới. 13
2.1.6. Sơ đồ chữ ký Elgamal 13
2.2. Viết chương trình demo với ngôn ngữ Python 15
2.2.1. Cài đặt và cấu hình các biến khởi tạo và trả về: 15
2.2.2. Thuật toán ký lên văn bản: 15
2.2.3. Thuật toán kiểm tra chữ ký. 16
2.2.4. Giao diện để ký: 17
2.2.5. Giao diện để kiểm tra chữ ký: 19
3. Nguyễn Nhật Quang -Tìm hiểu phương pháp mã hóa bất đối xứng ứng dụng trong chữ ký điện tử 22
3.1. Mật mã đối xứng là gì 22
3.2. Cách hoạt động của mật mã đối xứng 22
3.3. Ứng dụng của mật mã đối xứng 23
3.4. Lợi ích của mật mã đối xứng 24
3.5. So sánh mật mã đối xứng và bất đối xứng 25
3.6. Viết chương trình demo với ngôn ngữ PHP. 25
3.6.1. Demo chương trình. 25
4. Vũ Hồng Phương - Tìm hiểu về hàm băm SHA256 29
4.1. Khái niệm. 29
4.2. Đặc tính của hàm băm. 30
4.3. Hàm băm SHA (Secure Hash Algorithm) 31
4.3.1. Miêu tả SHA 31
4.3.2. Tính bảo mật trong SHA: 33
4.4. Một số ứng dụng của hàm băm. 34
4.5. Viết chương trình Demo với ngôn ngữ C++ 35
5. Ngô Văn Sáng – Ứng dụng xây dựng chương trình bằng ngôn ngữ C#.. 37

5.1. Ý tưởng và thuật toán 37
5.2. Giao diện sử dụng 41
Chương 3. PHẦN KIẾN THỨC LĨNH HỘI VÀ BÀI HỌC KINH NGHIỆM .. 45
I. NỘI DUNG ĐÃ THỰC HIỆN 45
II. HƯỚNG PHÁT TRIỂN. 46
Lời cảm ơn 49

LỜI NÓI ĐẦU

Vấn đề đảm bảo an ninh, an toàn thông tin dữ liệu là nội dung nghiên cứu thiết thực, là chủ đề luôn được các cấp, các ngành quan tâm trong lĩnh vực công nghệ thông tin. Nhu cầu đảm bảo an ninh thông tin dữ liệu trên mạng máy tính là cấp thiết trong các hoạt động kinh tế xã hội, đặc biệt là đối với các mạng máy tính chuyên dùng phục vụ công tác an ninh, quốc phòng, đối ngoại của các cơ quan Đảng, Nhà nước... Thực tế ứng dụng công nghệ thông tin trong các lĩnh vực liên quan đến an ninh chính trị, quốc phòng luôn gặp phải những rủi ro đột nhập trái phép, tấn công, lấy cắp thông tin

Xuất phát từ nhu cầu bảo mật trong chữ ký điện tử trên các văn bản điện tử nhóm em

Tìm hiểu về chữ ký điện tử và viết ứng dụng minh họa sẽ tìm hiểu vấn đề nêu trên và cải đặt chương trình ký số minh họa. Những vấn để nhóm chúng em tìm hiểu và viết ứng dụng minh họa bao gồm: Tìm hiểu về chữ ký điện tử, tìm hiểu phương pháp mà hóa bất đối xứng ứng dụng trong chữ ký điện tử, tìm hiểu về làm băm SHA, ứng dụng xây dựng chương trình.

Chương 1. TỔNG QUAN

I. TỔNG QUÁT VỀ ĐỀ TÀI NGHIÊN CỨU

Kể từ khi con người phát minh ra chữ viết, các chữ ký thường luôn được sử dụng hàng ngày, chẳng hạn như ký một biên nhận trên một bức thư nhận tiền từ ngân hàng, ký hợp đồng hay một văn bản bất kỳ nào đó. Chữ ký viết tay thông thường trên tài liệu thường được dùng để xác định người ký nó.

Sơ đồ chữ ký điện tử là một phương pháp ký một văn bản hay lưu bức điện dưới dạng điện tử. Chẳng hạn một bức điện có chữ ký được lưu hành trên mạng máy tính. Chữ ký điện tử từ khi ra đời đã có nhiều ứng dụng rộng rãi trong các giao dịch thương mại, từ việc xác minh chữ ký cho đến các thẻ tín dụng, các sơ đồ định danh và các sơ đồ chia sẻ bí mật… Sau đây, chúng ta sẽ tìm hiểu một số sơ đồ chữ ký quan trọng.

Qua đề tài trên chúng ta sẽ biết được thế nào là chữ ký điện tử, hệ chữ ký Elgamal, vai trò và ứng dụng của nó thông qua một số chương trình demo.

II. NỘI DUNG NGHIÊN CỨU

1. Chữ kí điện tử, chữ kí điện tử Elgamal

1.1. Chữ kí điện tử

- Chữ kí số (Digital Signature) là một định dạng điện tử được tạo ra bởi máy có hiệu quả và có hiệu lực như là các chữ ký tay.
- Là thông tin đi kèm theo dữ liệu nhằm mục đích xác nhận người chủ của dữ liệu đấy.
- Dựa trên kỹ thuật sử dụng mã hóa công khai: Mỗi người dùng phải có một cặp khóa gồm khóa công khai và khỏa bí mật.

- Giải thuật kiểm tra chữ ký số (Digital Signature verification algorithm) là một phương pháp xác minh tính xác thực của chữ ký số, có nghĩa là nó thực sự được tạo ra bởi 1 bên chỉ định.

1.2. Sơ đồ chữ ký Elgamal

- Là hệ chữ ký dựa trên sự phức tạp của bài toán logarit rời rạc và được trình bày bởi Taher Elgamal vào năm 1985
- Thuật toán sử dụng một cặp public key và private key. Private key được dùng để tạo ra chữ ký điện tử và người dùng có thể xác nhận dựa trên public key.

2. Tìm hiểu về hàm băm SHA.

- Giới thiệu.
- Các khái niệm.
- Ứng dụng.
- Thuật toán SHA.

3. Tìm hiểu phương pháp mã hóa bất đối xứng ứng dụng trong chữ ký điện tử.

- Mật mã đối xứng là gì?
- Cách hoạt động của mật mã đối xứng.
- Ứng dụng của mật mã đối xứng.
- Lợi ích của mật mã đối xứng.
- So sánh mật mã đối xứng và bất đối xứng.

4. Triển khai cụ thể với ngôn ngữ

Sử dụng các ngôn ngữ lập trình:

- PHP
- C#
- C++
- Python

III. CÁC KIẾN THỨC

1. Hướng nghiên cứu đề tài

- Nghiên cứu các giải thuật mã hóa khóa công khai.
- Nghiên cứu về chữ ký điện tử, tim hiểu về hàm băm và các giải thuật về hàm băm.
- Cài đặt thử nghiệm một giải thuật sinh chữ ký điện tử.

2. Đối tượng nghiên cứu

- Một số thuật toàn sinh khóa công khai khóa bí mật
- Một số hàm bầm thường sử dụng hiện nay
- Chữ ký điện tử.

3. Phương pháp nghiên cứu đề tài

- Tìm hiểu dựa trên cơ sở lý thuyết các thuật toán hay sinh khóa công khai đã có từ trước so sánh để thấy những ưu điểm, những hạn chế của từng thuật toán.
- Từ cơ sở đó có thể cải tiến, hoặc triển khai ứng dụng cài đặt một giải thuật tối ưu nhất vào thực tiễn.
- Trong quá trình triển khai ứng dụng nêu lên những hạn chế của đề tài và những khó khăn trong thực hiện đề tài.
- Đề xuất hướng phát triển của đề tài trong thời gian tới.

IV. CHỦ ĐỀ NGHIÊN CỨU

- Tìm hiểu về chữ ký điện tử Elgamal và viết ứng dụng minh họa
- Thuộc lĩnh vực hoạt động an toàn và bảo mật.

Chương 2. KẾT QUẢ NGHIÊN CỨU

I. GIỚI THIỆU

- Tên đề tài nghiên cứu: Tìm hiểu về chữ kí điện tử Elgamal và viết ứng dụng minh họa
- Các bước thực hiện đề tài gồm:
    - Tìm hiểu về chữ ký điện tử, chữ ký điện tử Elgamal.
    - Tìm hiểu phương pháp mã hóa bất đối xứng trong chữ ký điện tử.
    - Tìm hiểu về hàm băm SHA.
    - Ứng dụng xây dựng chương trình.
    - Demo chương trình.
- Hình thức sản phẩm: Sản phẩm ứng dụng + sản phẩm lý thuyết.
- Kết quả đạt được: Hoàn thành về nghiên cứu và demo sản phẩm ở 4 ngôn ngữ khác nhau.

II. NỘI DUNG THUẬT TOÁN

1. Tạo khóa

Mấu chốt cơ bản của việc sinh khóa trong Elgamal là tìm được bộ 3 số tự nhiên p, α và a với p là một số nguyên tố, α là một phân tử nguyên thủy của p và a là một số ngẫu nhiên.

Cụ thể, khóa của Elgamal được sinh như sau:

- Chọn một số nguyên tố p và phân tử nguyên tử α đủ lớn.
- Chọn một số nguyên a bất kỳ và giữ bí mật.
- Tính $\beta = \alpha^a \ mod \ p$
- Khóa công khai sẽ là {p, α, β} và khóa bí mật sẽ là {a}

2. Mã hóa

Giả sử Bob muốn gửi đoạn thông tin M cho Alice để ký. Đầu tiên Alice sẽ mã hóa M thành một chuỗi số và tính sig_k (x, k) = (γ, δ). Trong đó

- k là một số nguyên dương ngẫu nhiên và là số nguyên tố cùng nhau với p – 1.
- γ = αk mod p
- δ = (x – a* γ) k-1 mod (p - 1).

Lúc này Alice lưu chữ ký dưới dạng {M, γ, δ}

3. Giải mã

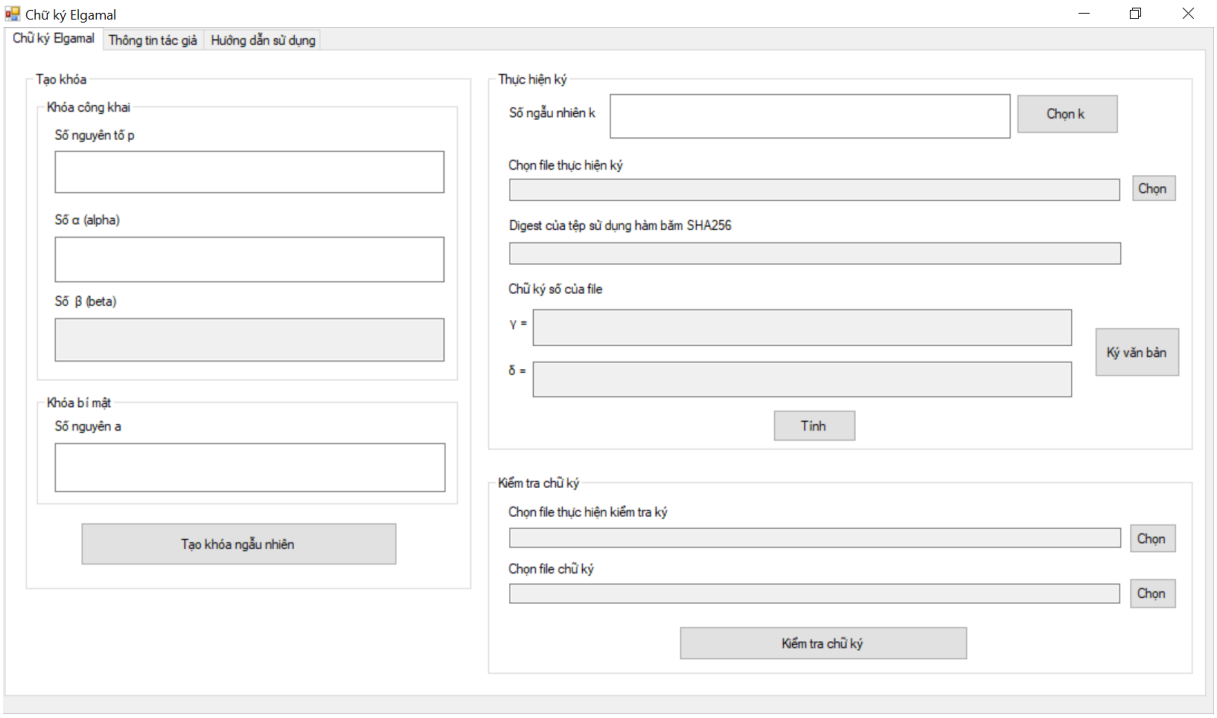
Bob có thể biết được Alice ký bằng cách sử dụng khóa công khai như sau:

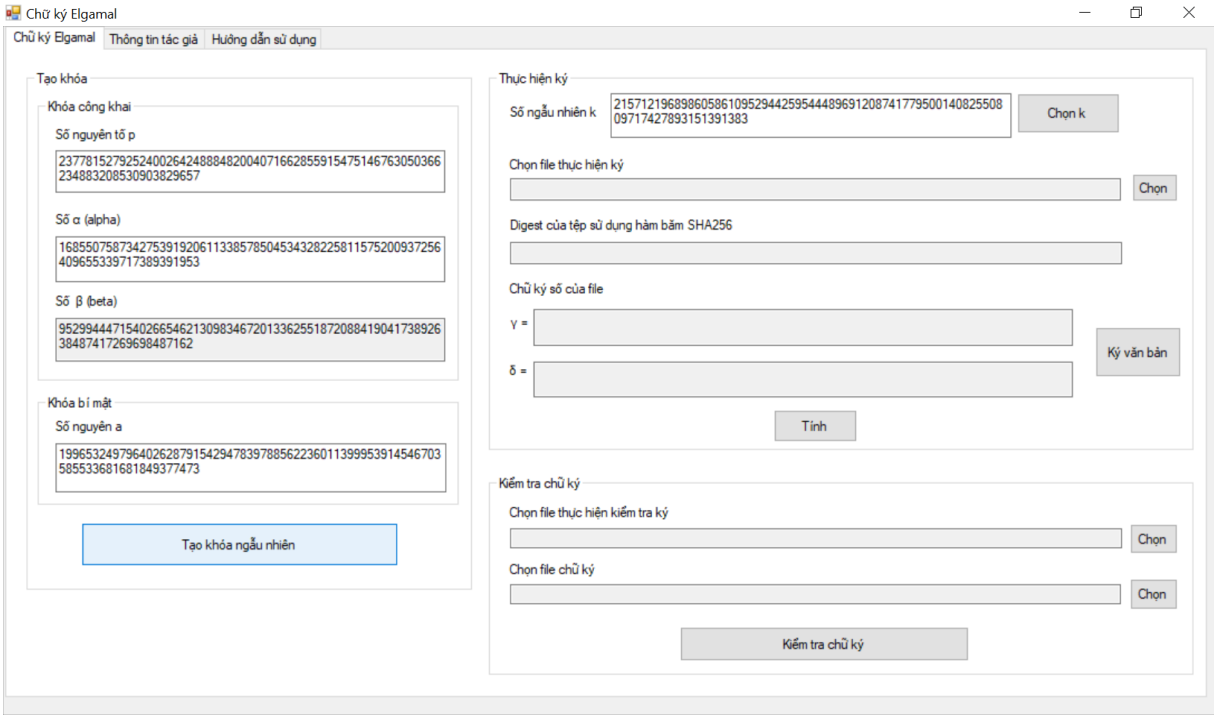
- Bob tính $\beta^\gamma \gamma^\delta$ và α^x (mod p).
- Nếu $\beta^\gamma \gamma^\delta \equiv \alpha^x$ (mod p) thì chữ ký thỏa mãn.

III. THIẾT KẾ, CÀI ĐẶT CHƯƠNG TRÌNH ĐỂ MÔ THUẬT TOÁN

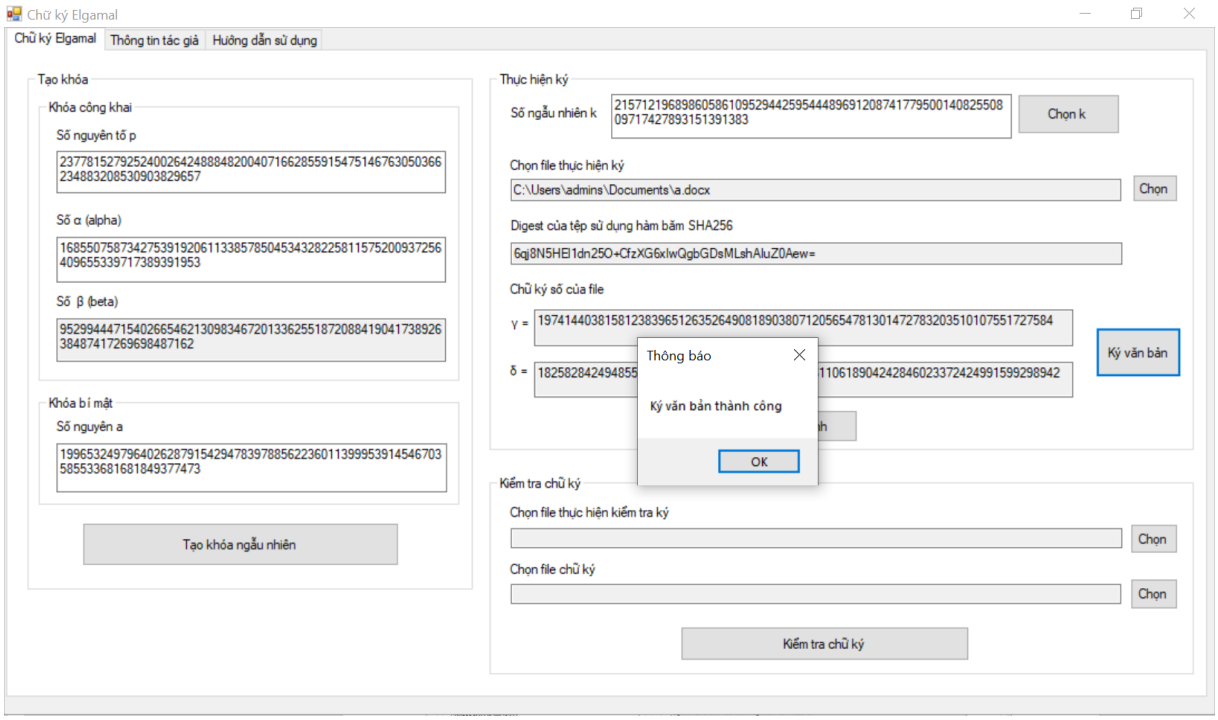
1. Giao diện chương trình demo

1.1. Giao diện màn hình Tạo khóa

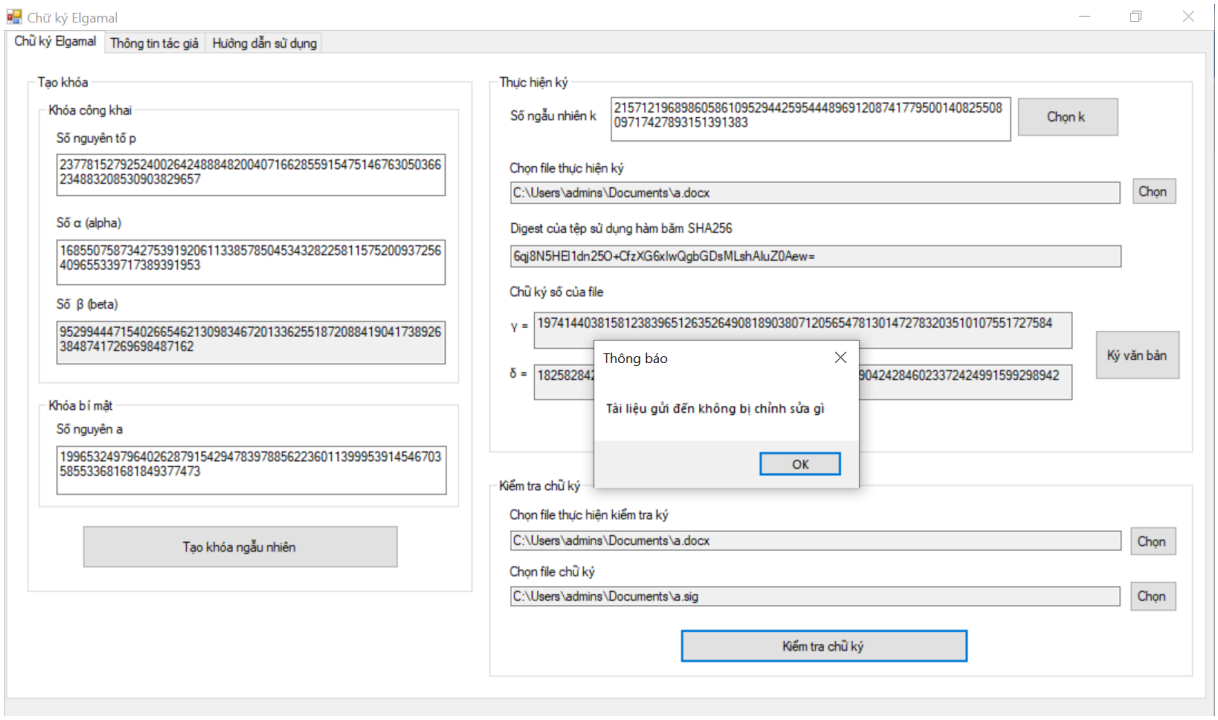


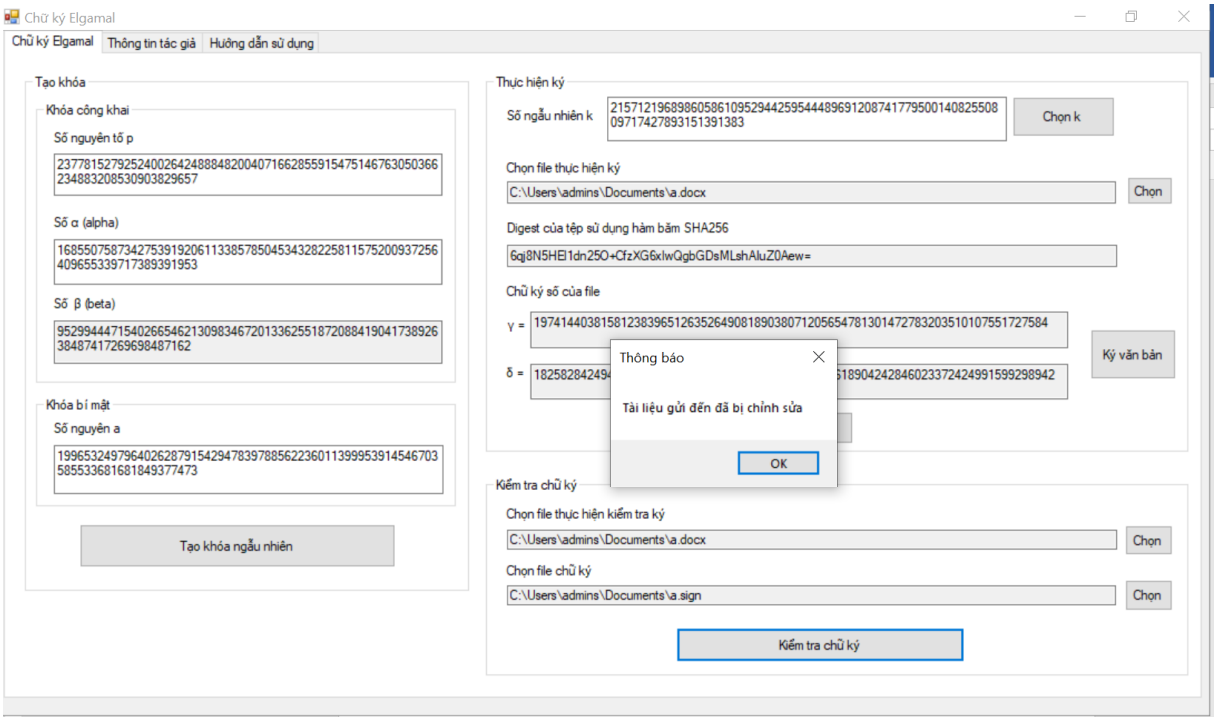


1.2. Giao diện màn hình kí văn bản



1.3. Giao diện màn hình Xác nhận chữ kí





IV. CÀI ĐẶT VÀ TRIỂN KHAI

1. Giới thiệu về PHP

PHP là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động.

PHP được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sản phẩm gia dụng, và có tên là Oak.

PHP được phát hành năm 1994, đến năm 2010 được Oracle mua lại từ Sun MicroSystem.

Đặc điểm của ngôn ngữ PHP.

- Độc lập phần cứng và hệ điều hành.
- Ngôn ngữ thông dịch.
- Cơ chế thùng rác tự động.
- Đa luồng.

- Tính an toàn và bảo mật.

2. Giới thiệu về C#

C# (hay C sharp) là một ngôn ngữ lập trình đơn giản, được phát triển bởi đội ngũ kỹ sư của Microsoft vào năm 2000. C# là ngôn ngữ lập trình hiện đại, hướng đối tượng và được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java.

Đặc trưng của ngôn ngữ C#:

- Đơn giản.
- Hiện đại.
- Là ngôn ngữ lập trình hướng đối tượng.
- Ít từ khóa.

3. Giới thiệu về C++

C++ là một loại ngôn ngữ lập trình bậc trung (middle-level). Đây là ngôn ngữ lập trình đa năng được tạo ra bởi Bjarne Stroustrup.

Từ thập niên 1990, C++ đã trở thành một trong những ngôn ngữ thương mại ưa thích và phổ biến của lập trình viên.

C++ là một phiên bản mở rộng của ngôn ngữ lập trình C. Những bản cập nhật gần đây nhất là C++ 14 và C++ 17, và sắp tới là C++ 20 (đang trong quá trình phát triển), đã và đang mang đến những tính năng hỗ trợ rất lớn cho lập trình viên C++.

Đặc điểm của C++

- Ngôn ngữ lập trình bậc trung.
- Đơn giản và hiệu quả.
- Hỗ trợ đa nền tảng.
- Lập trình hướng đối tượng.
- Con trỏ.

4. Giới thiệu về Python

Python đã được hình thành vào cuối những năm 1980.

Được thực hiện vào tháng 12 năm 1989 bởi Guido van Rossum tại Centrum Wiskunde & Informatica (CWI) ở Hà Lan.

Là một kế thừa cho ngôn ngữ ABC (tự lấy cảm hứng từ SETL) có khả năng xử lý ngoại lệ và giao tiếp với Hệ điều hành Amoeba.

Van Rossum là tác giả chính của Python, và vai trò trung tâm của ông trong việc quyết định hướng phát triển của Python.

Đặc điểm của Python:

- Python là một ngôn ngữ thông dịch, nghĩa là Python không cần phải được biên dịch trước khi nó được chạy.
- Miễn phí, mã nguồn mở.
- Khả năng di động linh hoạt.
- Khả năng mở rộng và có thể nhưng.
- Ngôn ngữ thông dịch cấp cao (khi chạy code Python, nó sẽ tự động chuyển đổi code sang ngôn ngữ máy tính có thể hiểu).
- Thư viện tiêu chuẩn lớn để giải quyết những tác vụ phổ biến.
- Hướng đối tượng (giúp giải quyết vấn đề phức tạp 1 cách trực quan).

V. THỰC HIỆN BÀI TOÁN

1. Phân công công việc

Tên sinh viên	Tên công việc
Hoàng Minh Quý	Tìm hiểu về chữ ký điện tử, chữ ký điện tử RSA.
Nguyễn Nhật Quang	Tìm hiểu phương pháp mã hóa bất đối xứng ứng dụng trong chữ ký điện tử.
Vũ Hồng Phương	Tìm hiểu về hàm băm SHA.
Ngô Văn Sáng	Ứng dụng xây dựng chương trình.

2. Hoàng Minh Quý - Tìm hiểu về chữ ký điện tử, chữ ký điện tử Elgamal.

2.1. Chữ kí điện tử, chữ kí điện tử Elgamal

2.1.1. Chữ kí điện tử

- Là một định dạng điện tử được tạo ra bởi máy có hiệu quả và có hiệu lực như là các chữ ký tay.
- Là thông tin đi kèm theo dữ liệu nhằm mục đích xác nhận người chủ của dữ liệu đấy.
- Giải thuật kiểm tra chữ ký số (Digital Signature verification algorithm) là một phương pháp xác minh tính xác thực của chữ ký số, có nghĩa là nó thực sự được tạo ra bởi 1 bên chỉ định.
- Dựa trên kỹ thuật sử dụng mã hóa công khai: Mỗi người dùng phải có một cặp khóa gồm khóa công khai và khỏa bí mật.

2.1.2. Các thành phần của chữ ký điện tử:

- Chữ ký số điện tử bao gồm 3 thành phần: thuật toán tạo ra khóa, hàm tạo chữ ký và hàm kiểm tra chữ ký.

- Hàm tạo ra chữ ký là hàm tính toán chữ ký trên cơ sở khóa mật và dữ liệu cần ký
- Hàm kiểm tra chữ ký là hàm kiểm tra xem chữ ký đã cho có đúng với khóa công công cộng không. Khóa này mọi người có quyền truy cập cho nên mọi người đều có thể kiểm tra được chữ ký.

2.1.3. Các thành phần của sơ đồ chữ ký điện tử.

Sơ đồ chữ ký bao gồm các thành phần sau:

- Không gian bản rõ M.
- Không gian chữ ký S.
- Không gian khóa K để tạo nên chữ ký, không gian khóa K' để kiểm tra chữ ký.
- Thuật toán hiệu quả để tạo nên khóa Gen: $N \alpha \ K \times K'$, ở đây K và K' tương ứng với không gian khóa mật và khóa công cộng.
- Thuật toán tạo chữ ký Sign: $M \times K \alpha \ S$.
- Thuật toán kiểm tra chữ ký Verify: $M \times K \times K' \alpha \ \{True, False\}$.

Đối với bất kỳ khóa tạo chữ ký $sk \in K$ và bất kỳ bản tin lệnh ký bức điện được ký hiệu:

$$s \leftarrow Sign_{sk}(m).$$

Biểu thức này được đọc như sau: s - là chữ ký của bản tin m được tạo ra nhờ thuật toán Sign và khóa mật sk.

Đối với bất kỳ khóa mật của chữ ký $sk \in K$, tương ứng với khóa công cộng để kiểm tra chữ ký là $pk \in K'$, bất kỳ bản tin $m \in M$ và chữ ký $s \in S$ cần thỏa mãn điều kiện sau:

$$Verify_{pk(m,s)} = \begin{cases} True, if & s = Sign_{sk(m)} \\ False, if & s \neq Sign_{sk(m)} \end{cases}$$

Bởi vì tài liệu cần ký thường có chiều dài khá dài. Một biện pháp để ký là chia tài liệu ra các đoạn nhỏ và sau đó ký lên từng đoạn và ghép lại. Nhưng phương pháp có nhược điểm là chữ ký lớn, thứ hai là ký chậm vì hàm ký là các hàm mũ,

thứ ba là chữ ký có thể bị đảo loạn các vị trí không đảm tính nguyên vẹn của tài liệu. Chính vì điều đó mà khi ký thì người ta ký lên giá trị hàm hash của tài liệu, vì giá trị của hàm hash luôn cho chiều dài xác định.

2.1.4. Chức năng của chữ ký điện tử.

- Xác thực được nguồn gốc tài liệu: Tuy thuộc vào từng bản tin mà có thể thêm các thông tin nhận dạng, như tên tác giả, nhản thời gian…vv.
- Tính toàn vẹn tài liệu. Vì khi có một sự thay bất kỳ vô tình hay cố ý lên bức điện thì gía trị hàm hash sẽ bị thay đổi và kết quả kiểm tra bức điện sẽ không đúng.
- Chống từ chối bức điện. Vì chỉ có chủ của bức điện mới có khóa mật để ký bức điện.

2.1.5. Các ứng dụng về chữ ký điện tử trên thế giới.

- Annature.
- DocuSign.
- FuseSign.
- ….

2.1.6. Sơ đồ chữ ký Elgamal

- Là hệ chữ ký dựa trên sự phức tạp của bài toán logarit rời rạc và được trình bày bởi Taher Elgamal vào năm 1985.
- Thuật toán sử dụng một cặp public key và private key. Private key được dùng để tạo ra chữ ký điện tử và người dùng có thể xác nhận dựa trên public key.

➢ Mô tả

❖ Cho p là một số nguyên tố như là bài toán logarit rời rạc trong Zp, $\alpha \in Zp^*$ là một phần tử nguyên và $P = Zp^*$, $A = (Zp^*) * Zp-1$, và định nghĩa:
$K = \{(p, \alpha, a, \beta): \beta \equiv \alpha a \pmod p\}$

Trong đó giá trị p, α và β là công khai, còn a là bí mật.

Với K = {p, α, a, β} và chọn một số ngẫu nhiên k ∈ Zp-1*, định nghĩa:

sigk (x, k) = (γ, δ)

Trong đó: $\gamma = \alpha^k \bmod p$

δ = (x – a* γ) k-1 mod (p - 1).

Với x, $\gamma \in Z_p^*$ và $\delta \in Z_{p-1}$, định nghĩa:

Ver (x, c, δ) = TRUE ⇔ $\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$.

Nếu chữ ký là đúng thì việc xác nhận thành công khi:

$$\beta^\gamma \gamma^\delta \equiv \alpha^{a\gamma} \alpha^{k\delta} \pmod p$$

$$\equiv \alpha^x \pmod p.$$

Trong đó: aγ + kδ ≡ x (mod p-1).

B sẽ tính toán chữ ký bằng việc sử dụng cả giá trị bí mật a (một phần của khóa) và số bí mật ngẫu nhiên k (giá trị để ký bức điện). Việc xác minh có thể thực hiện được chỉ với các thông tin được công khai.

➢ Ví dụ:

Chúng ta chọn p = 467, α = 2, a = 127. Ta tính: $\beta = \alpha^a \bmod p = 2127 \bmod 467 = 132$.

Bây giờ B muốn ký lên bức điện x = 100 và anh ta chọn một giá trị ngẫu nhiên k = 213 (chú ý là UCLN (213, 466) = 1 và 213-1 = mod 466 = 431). Sau đó tính:

$\gamma = 2^{213} \bmod 467 = 29$.

δ = (100 – 127*29)431 mod 466 = 51.

Bất cứ ai cũng có thể kiểm tra chữ ký này bằng cách tính:

$132^{29}29^{51} \equiv 189 \pmod{467}$.

$2^{100} \equiv 189 \pmod{467}$.

2.2. Viết chương trình demo với ngôn ngữ Python

2.2.1. Cài đặt và cấu hình các biến khởi tạo và trả về:

Các tham biến cần thiết của thuật toán như p, α, β, khóa bí mật a, khóa công khai được lưu dưới dạng sau:

```
p = 65675690019899337996609165991 1
alpha = 1402
a = 8920889 #primitive root - using caculator.py to render
beta = builtins.pow(alpha, a, mod=p)
privateKey = {
    "a": a
}
publicKey = {
    "p": p,
    "alpha": alpha,
    "beta": beta
}
```

Sau khi ký thì chữ ký được lưu dưới dạng sau:

elgamal.py {} anchor.pdf0.json ×

{} anchor.pdf0.json > ...


```
{
    "rKey": 518720597204153479838942970644,
    "sKey": 628325064558675029646625333819,
    "pubKey": {
        "p": 65675690019899337996609165991 1,
        "alpha": 1402,
        "beta": 171559504267430146499453101270
    }
}
```


2.2.2. Thuật toán ký lên văn bản:

Cần truyền vào văn bản đã được hash sha256 và base sang decimal (việc lấy doc và hash sha256 sẽ được đề cập chi tiết đến xử lý UI), private key, public key.

Lúc này cần render ra hệ số k ngẫu nhiên thỏa mãn k và p – 1 là hai số nguyên tố cũng nhau. (ở đây để tiện tính toán số lớn, code sẽ ngẫu nhiên k trong khoảng p/2 đến p -1)

Tính rKey = $a^k \ mod \ p$ và sKey = $k^{-1}(m - a * rKey) \ mod \ p$ với m là giá trị của doc khi được hash.

Trả về bằng chứng (rKey, sKey).

```
def SigningTheMessage(message, privateKey, publicKey):
    p = publicKey["p"]
    a = privateKey["a"]
    alpha = publicKey["alpha"]
    #hard code
    # k = 31
    halfofP = (p-1)/2
    k = random.randint(halfofP, p-1)
    print("k value is: " + str(k))
    while(gcd(k, p - 1) != 1):
        k = random.randint(halfofP, p-1)
    reverse_k_mod_p = builtins.pow(k, -1, mod=p-1)
    rKey = builtins.pow(alpha, k, p)
    #return the proof which prove User signed this message
    return{
        "rKey": rKey,
        "sKey": reverse_k_mod_p*(message - a * rKey) % (p-1),
        "pubKey": publicKey
    }
```

2.2.3. Thuật toán kiểm tra chữ ký.

Cần truyền vào văn bản đã được hash sha256 và base sang decimal, public key, bằng chứng ký văn bản.

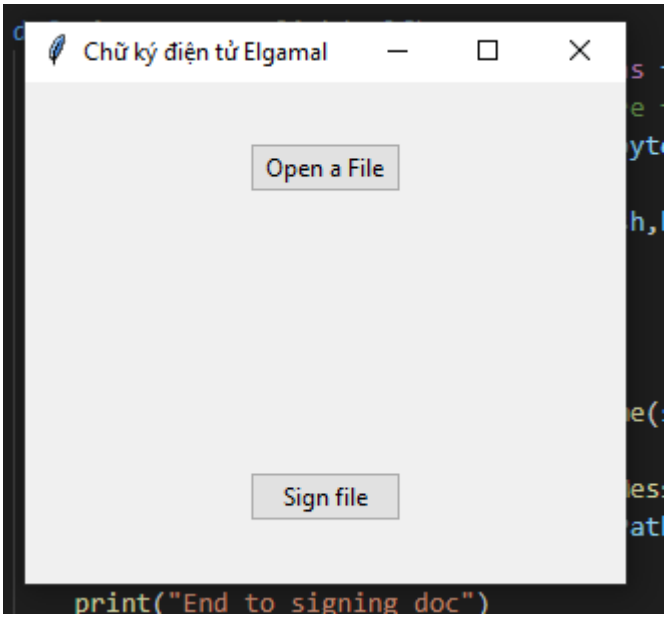
Để biết được chữ ký trên có thuộc văn bản được truyền vào hay không chúng ta kiểm tra biểu thức sau có thỏa mãn không:

$$\beta^r r^s \equiv a^m mod \ p$$

$$Với \ r = rKey, s = sKey, m = hash(doc)$$

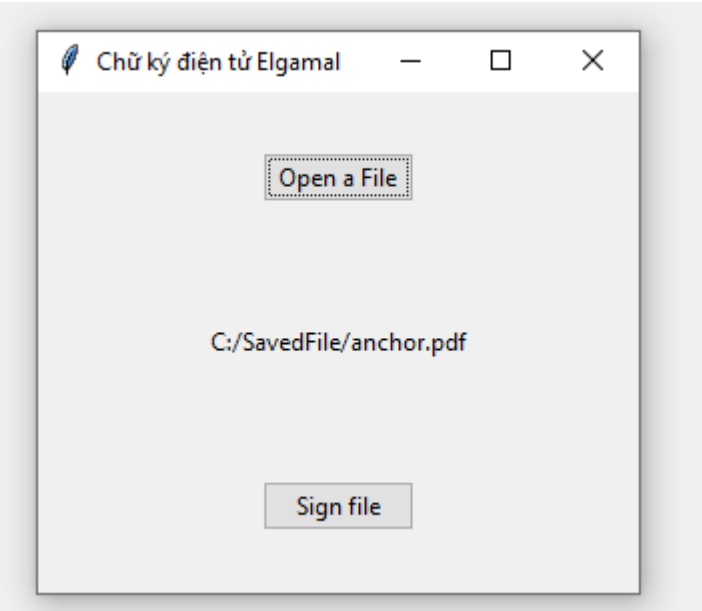
```python
#Verify Signature with public key
def VerifyTheSignature(message, proof, publicKey):
    rKey = proof["rKey"]
    sKey = proof["sKey"]
    p = publicKey["p"]
    alpha = publicKey["alpha"]
    beta = publicKey["beta"]

    firstParaInFirstVerify = builtins.pow(beta, rKey, mod=p)
    secondParaInFirstVerify = builtins.pow(rKey, sKey, mod=p)
    firstVerify = firstParaInFirstVerify * secondParaInFirstVerify % p
    secondVerify = builtins.pow(alpha, message, mod=p)
    if(firstVerify == secondVerify):
        return True
    return False
```

2.2.4. Giao diện để ký:



Các tham số của publicKey và privateKey sẽ được hard code trực tiếp vào code.

Ấn Open a File để chọn 1 file muốn ký.

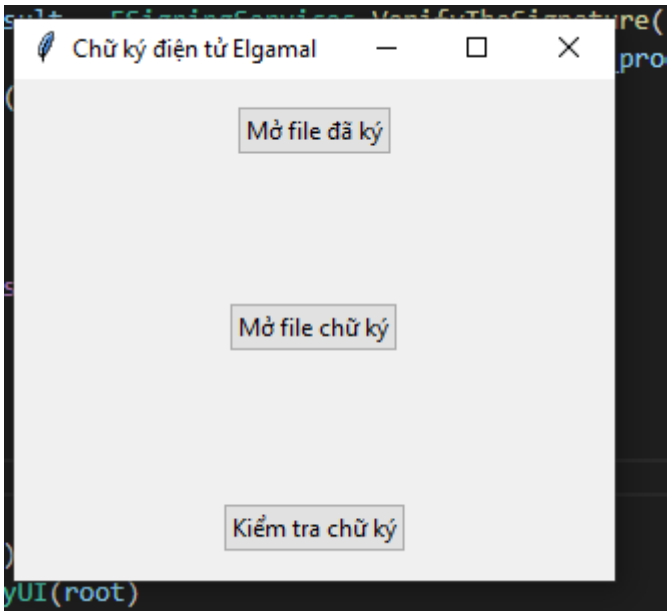


Ấn Sign file để ký, lúc này chương trình sẽ hash file doc theo sha256 và chuyển sang hệ cơ số 10. Sau đó gọi hàm ký lên văn bản để ký và lưu chữ ký vào file.

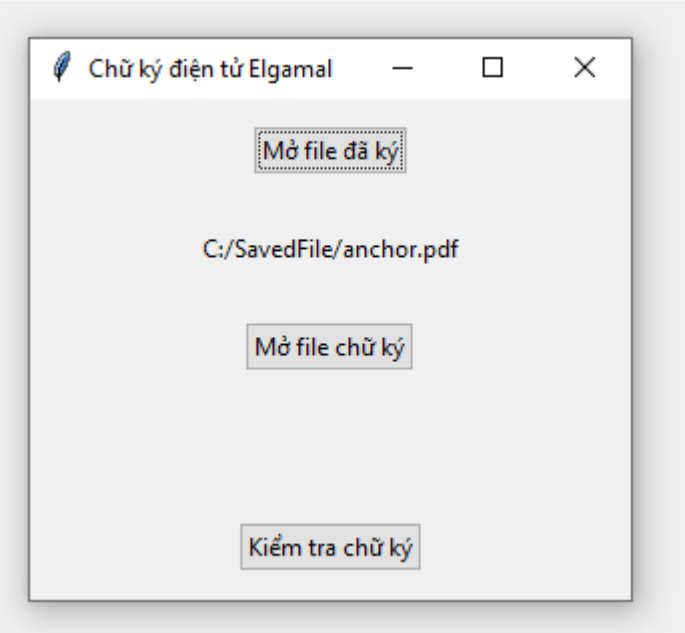
Code:

```
def signButtonOnClick(self):
    with open(self.docPath.get(),"rb") as file:
        bytes = file.read() # read entire file as bytes
        readable_hash = hashlib.sha256(bytes).hexdigest()
        #to decimal for calculated
        hashInDecimal = int(readable_hash,base=16)
    print(readable_hash)
    print(hashInDecimal)
    print("Start to signing doc")
    offset = 0
    while(os.path.isfile(os.path.basename(self.docPath.get()) + str(offset) +'.txt')):
        offset+=1
    proof = ESigningServices.SigningTheMessage(hashInDecimal,self.privateKey,self.publicKey)
    with open(os.path.basename(self.docPath.get()) + str(offset) +'.txt', 'w') as json_file:
        json.dump(proof, json_file)
    print("End to signing doc")
    showinfo(
            title='Thông báo',
            message="Ký thành công"
    )
```

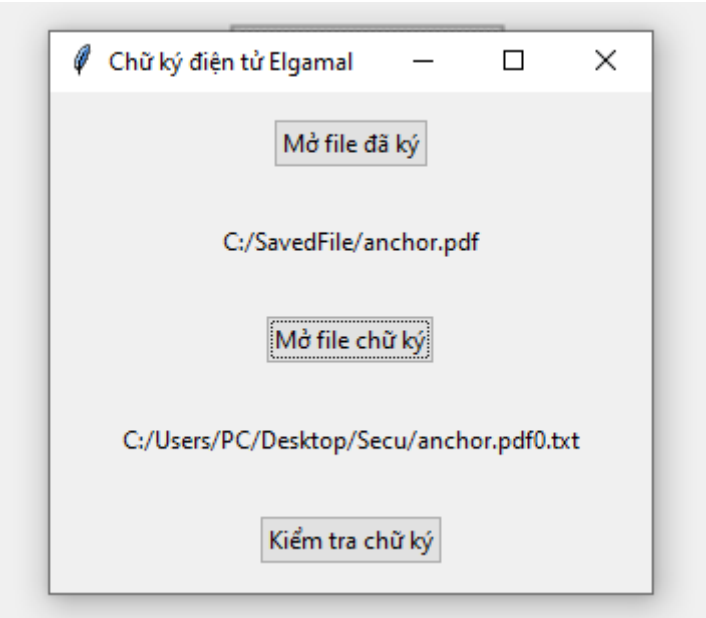
2.2.5. Giao diện để kiểm tra chữ ký:



Ấn Mở File đã ký để chọn file đã ký.

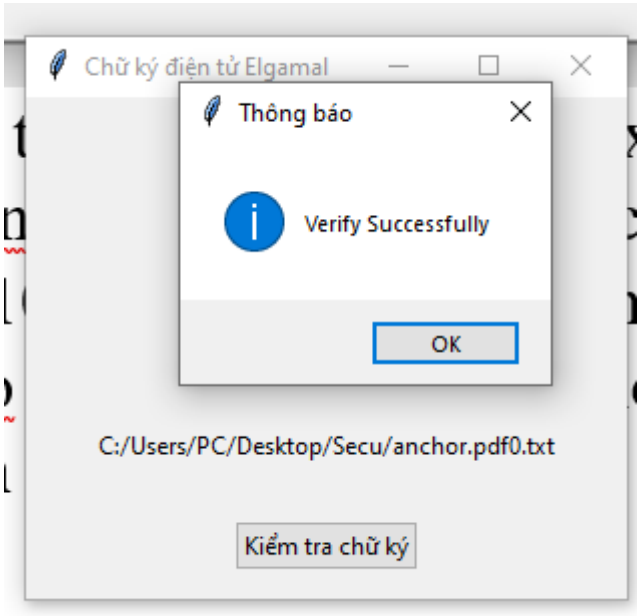


Ấn Mở File chữ ký để chọn file chữ ký tương ứng.

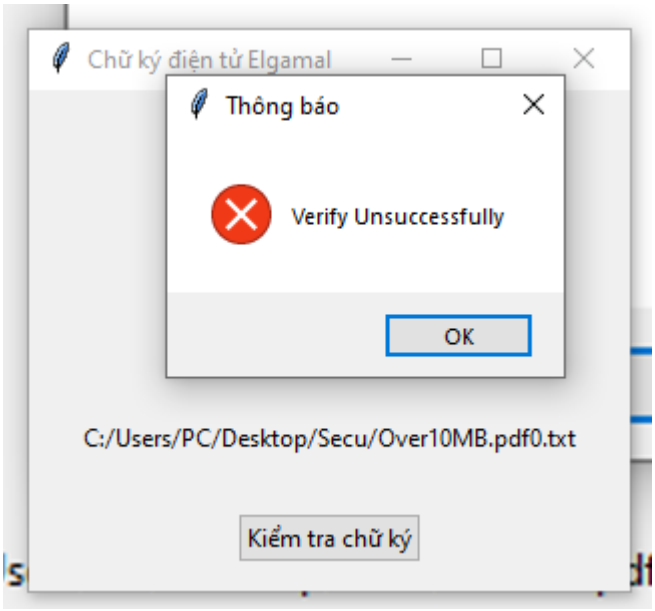


Ấn Kiểm tra chữ ký để kiểm tra xem đúng người ký chưa. Lúc đấy chương trình sẽ hash file doc theo sha256 và chuyển sang hệ cơ số 10. Sau đó gọi hàm kiểm tra chữ ký để kiểm tra và thông báo kết quả thông qua dialog.

- Nếu kiểm tra thành công:



- Nếu kiểm tra không thành công:



Code của phần xử lý:

```
def verifyButtonAfterClick(self):
    print("Start Hashing Doc")
    with open(self.docPath.get(), "rb") as file:
        bytes = file.read()  # read entire file as bytes
        readable_hash = hashlib.sha256(bytes).hexdigest()
        #to decimal for calculated
        hashInDecimal = int(readable_hash, base=16)
    print(readable_hash)
    print(hashInDecimal)
    with open(self.proof_Of_Signing_Path.get()) as file:
        data_of_proof = json.load(file)
    result = ESigningServices.VerifyTheSignature(
        hashInDecimal, data_of_proof, data_of_proof["pubKey"])
    if(result is True):
        showinfo(
            title='Thông báo',
            message="Verify Successfully"
        )
    else:
        showerror(
            title='Thông báo',
            message="Verify Unsuccessfully"
        )
```

3. Nguyễn Nhật Quang -Tìm hiểu phương pháp mã hóa bất đối xứng ứng dụng trong chữ ký điện tử

- **Tổng quan về các hệ mã hóa.**

3.1. Mật mã đối xứng là gì

Hệ thống mã hóa bất đối xứng (Asymmetric cryptosystem) là hệ thống mã hóa sử dụng một khóa công khai (public key) và một khóa bí mật (private key) cho quá trình mã hóa và giải mã.

Hệ thống mã hóa bất đối xứng còn được gọi là hệ thống mã hóa khóa công khai (public-key cryptosystem)

Kỹ thuật mã hóa bất đối xứng:

- Kỹ thuật mã hóa bất đối xứng phổ biến: RSA.
- RSA: tên được đặt theo tên 3 nhà phát minh ra giải thuật Rivest, Shamir và Adleman.
- Thuật toán sử dụng 2 khóa có quan hệ toán học với nhau: khóa công khai và khóa bí mật.
- Khóa công khai được công bố rộng rãi cho mọi người và được dùng để mã hóa. Khóa bí mật dùng để giải mã.

Nhiều giao thức dựa trên mật mã không đối xứng, bao gồm giao thức bảo mật lớp truyền tải (TLS) và giao thức lớp cổng bảo mật (SSL), giúp HTTPS khả thi. Quá trình mã hóa cũng được sử dụng trong các chương trình phần mềm - chẳng hạn như trình duyệt - cần thiết lập kết nối an toàn qua mạng không an toàn như Internet hoặc cần xác thực chữ ký điện tử.

3.2. Cách hoạt động của mật mã đối xứng

Mã hóa không đối xứng sử dụng một cặp khóa liên quan đến toán học để mã hóa và giải mã: khóa công khai và khóa riêng tư. Nếu khóa công khai được sử dụng để mã hóa, thì khóa cá nhân liên quan được sử dụng để giải mã; nếu khóa

riêng tư được sử dụng để mã hóa, thì khóa công khai liên quan được sử dụng để giải mã.

Hai người tham gia vào quy trình mã hóa bất đối xứng là người gửi và người nhận; mỗi khóa có cặp khóa công khai và khóa riêng. Đầu tiên, người gửi nhận được khóa công khai của người nhận. Tiếp theo, bản rõ - hoặc văn bản thông thường, có thể đọc được - được mã hóa bởi người gửi bằng khóa công khai của người nhận; điều này tạo ra bản mã. Sau đó, bản mã được gửi đến người nhận, người sẽ giải mã bản mã bằng khóa riêng của họ và trả nó về bản rõ dễ đọc.

Do tính chất một chiều của chức năng mã hóa, một người gửi không thể đọc tin nhắn của người gửi khác, mặc dù mỗi người đều có khóa công khai của người nhận.

3.3. Ứng dụng của mật mã đối xứng

Mật mã không đối xứng thường được sử dụng để xác thực dữ liệu bằng chữ ký số. Chữ ký điện tử là một kỹ thuật toán học được sử dụng để xác nhận tính xác thực và tính toàn vẹn của một thông điệp, phần mềm hoặc tài liệu kỹ thuật số. Nó là dạng kỹ thuật số tương đương với chữ ký viết tay hoặc con dấu có đóng dấu.

Dựa trên mật mã không đối xứng, chữ ký điện tử có thể cung cấp bằng chứng đảm bảo về nguồn gốc, danh tính và trạng thái của một tài liệu, giao dịch hoặc thông điệp điện tử, cũng như xác nhận sự đồng ý của người ký.

Mật mã không đối xứng cũng có thể được áp dụng cho các hệ thống trong đó nhiều người dùng có thể cần mã hóa và giải mã các thông điệp, bao gồm:
Email được mã hóa - một khóa công khai có thể được sử dụng để mã hóa một tin nhắn và một khóa riêng tư có thể được sử dụng để giải mã nó.

Các giao thức mật mã SSL / TSL - thiết lập các liên kết được mã hóa giữa các trang web và trình duyệt cũng sử dụng mã hóa không đối xứng.

3.4. Lợi ích của mật mã đối xứng

Các lợi ích của mật mã không đối xứng bao gồm:

- Vấn đề phân phối khóa được loại bỏ vì không cần trao đổi khóa.
- Bảo mật được tăng cường vì các khóa riêng tư không bao giờ phải truyền hoặc tiết lộ cho bất kỳ ai. Đây là quy trình mã hóa an toàn nhất vì người dùng không bao giờ bị yêu cầu tiết lộ hoặc chia sẻ khóa riêng tư của họ, do đó làm giảm khả năng tội phạm mạng phát hiện ra khóa riêng của người dùng trong quá trình truyền.
- Việc sử dụng chữ ký điện tử được kích hoạt để người nhận có thể xác minh rằng thư đến từ một người gửi cụ thể.
- Nó cho phép không từ chối để người gửi không thể từ chối gửi tin nhắn.

Hạn chế bao gồm:

- Đó là một quá trình chậm so với mật mã đối xứng, vì vậy nó không thích hợp để giải mã hàng loạt tin nhắn.
- Nếu một cá nhân mất khóa cá nhân của mình, anh ta không thể giải mã các tin nhắn mà anh ta nhận được.
- Vì khóa công khai không được xác thực, không ai thực sự biết liệu khóa công khai có thuộc về người được chỉ định hay không. Do đó, người dùng phải xác minh rằng khóa công khai của họ thuộc về họ.
- Nếu một tin tặc xác định khóa cá nhân của một người, kẻ tấn công có thể đọc tất cả các tin nhắn của cá nhân đó.

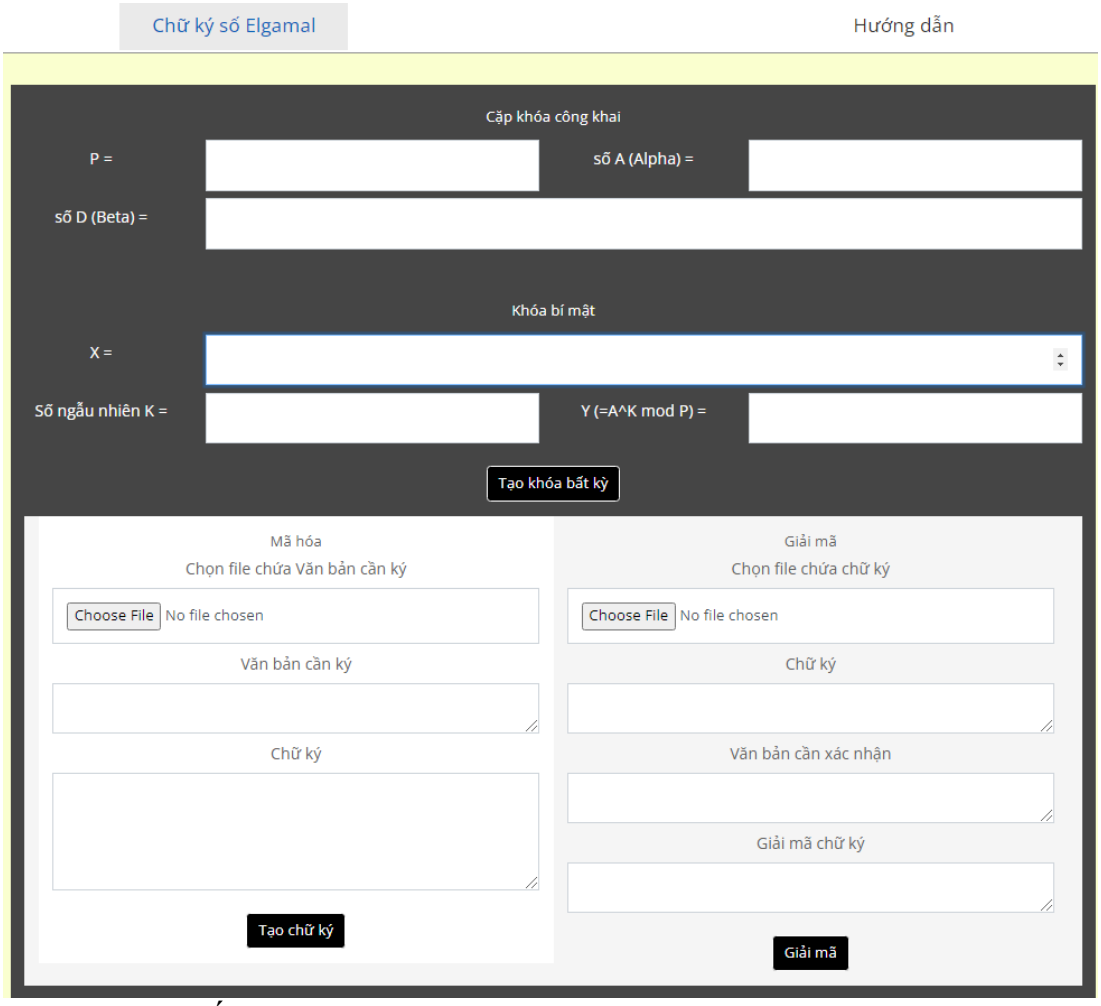
3.5. So sánh mật mã đối xứng và bất đối xứng

Mã hóa đối xứng	Mã hóa không đối xứng
Mã hóa đối xứng bao gồm một khóa để mã hóa và giải mã.	Mã hóa không đối xứng bao gồm hai khóa mật mã được gọi là khóa công khai và khóa cá nhân.
Mã hóa đối xứng nhanh hơn rất nhiều so với phương pháp không đối xứng.	Vì mã hóa không đối xứng kết hợp hai khóa riêng biệt, quá trình này bị chậm lại đáng kể.
RC4	RSA
AES	Diffie-Hellman
DES	ECC
3DES	ElGamal
QUAD	DSA

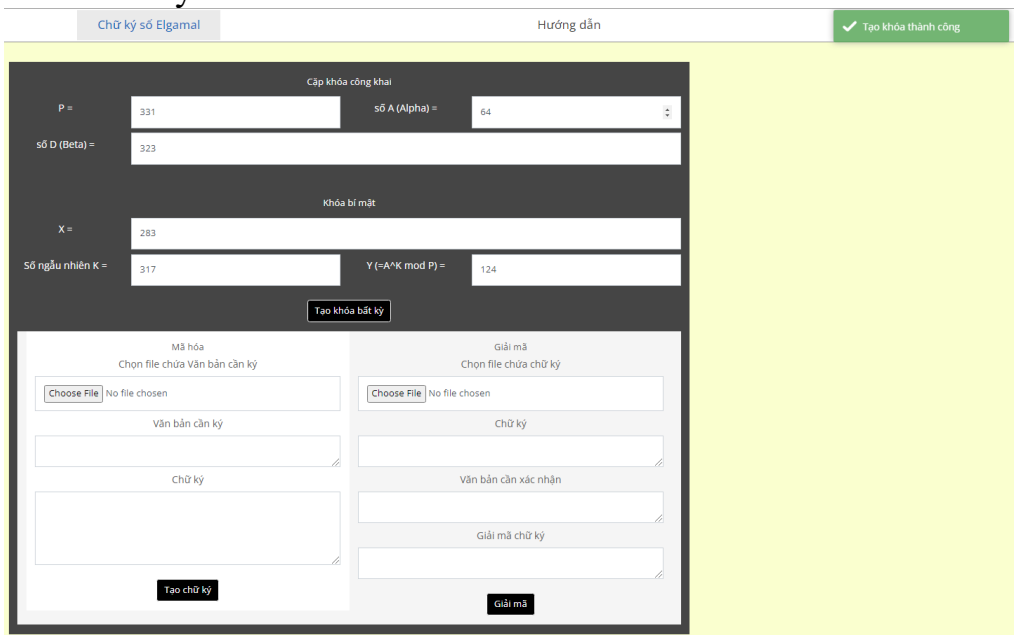
3.6. Viết chương trình demo với ngôn ngữ PHP.

3.6.1. Demo chương trình.

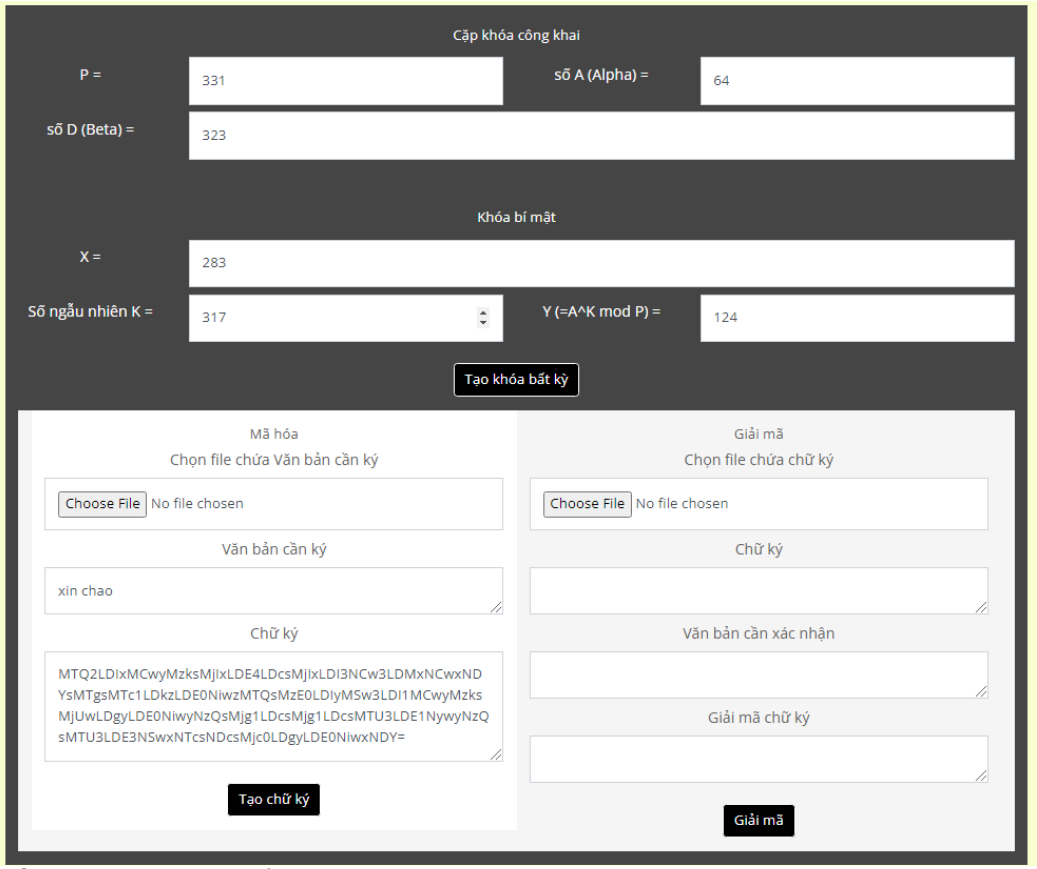
- Giao diện chương trình:



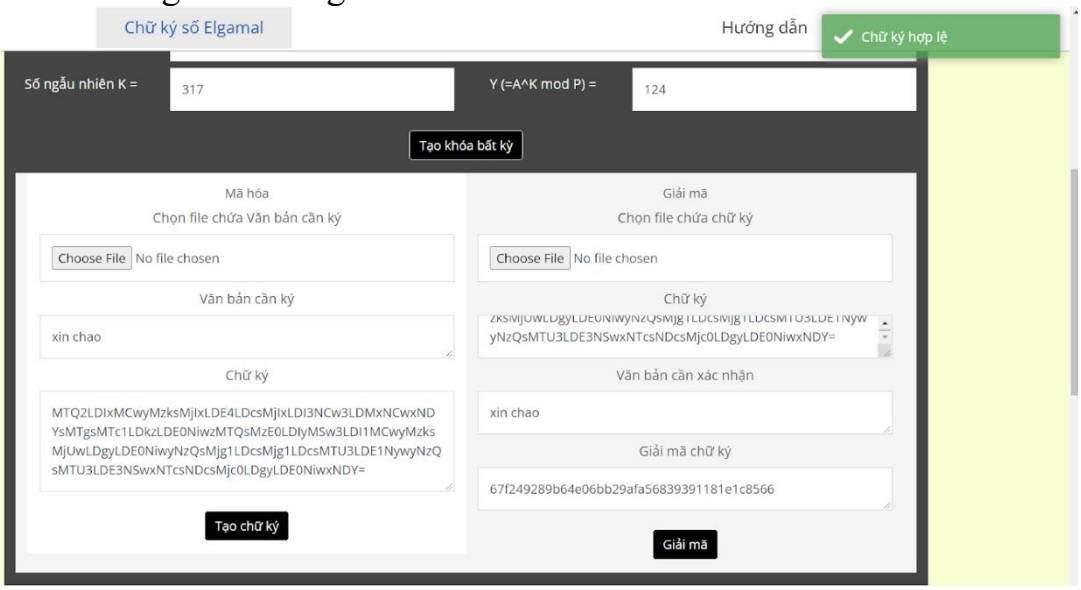
- Tạo khóa bất kỳ



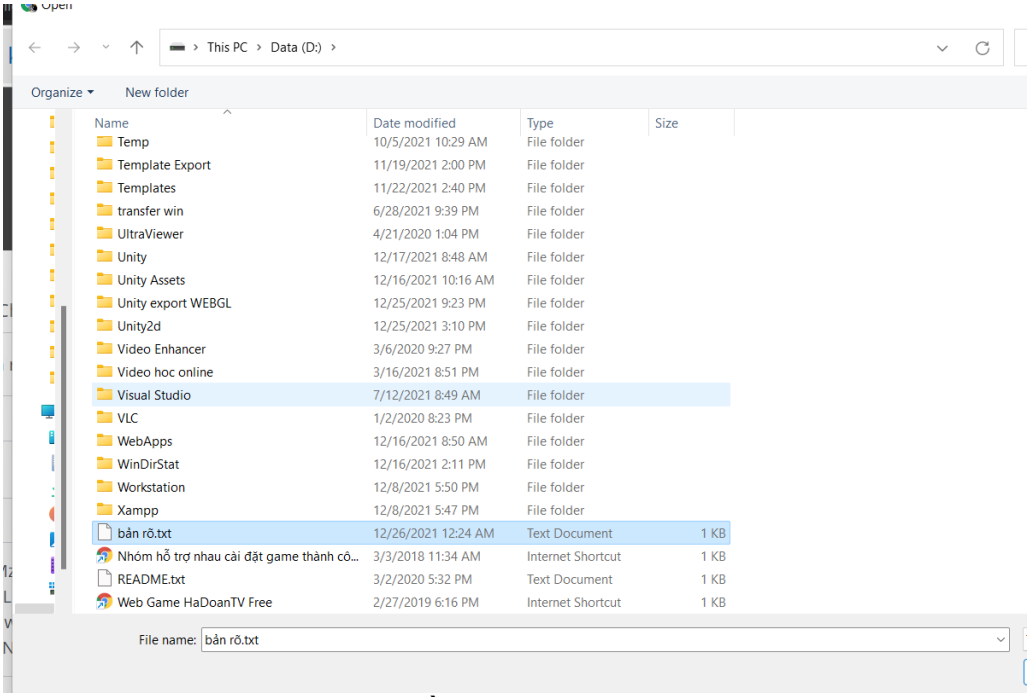
- Nhập văn bản:
  - Gõ văn bản cần mã hóa.
  - Ấn nút mã hóa và hiển thị được bản mã hóa.



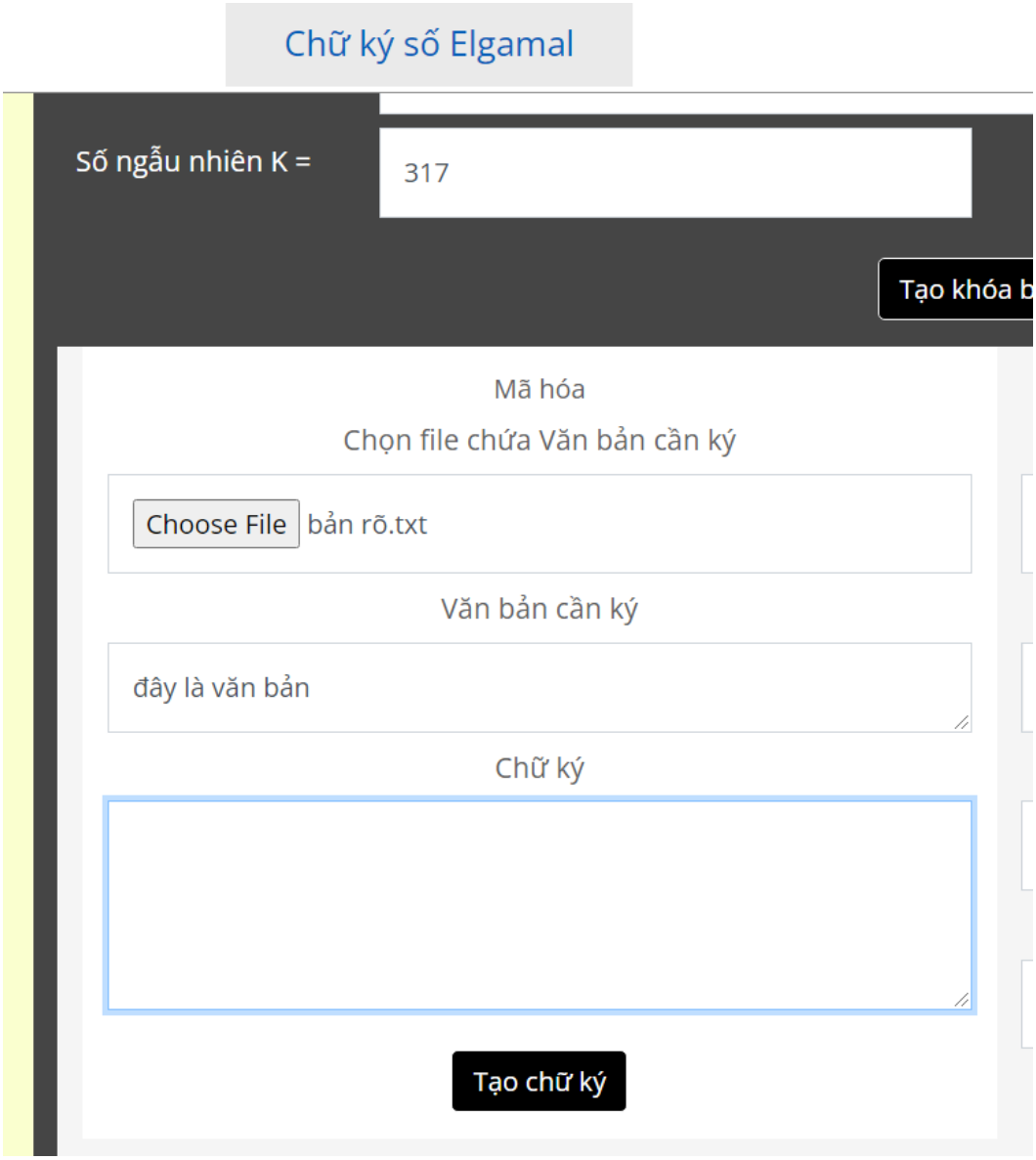
- Ấn nút giải mã để giải mã bản mã hóa



- Nhập File
  - Nhấn nút Choose File.
  - Hiển thị giao diện explore để chọn file.



- Chọn file txt chứa văn bản cần mã hóa:

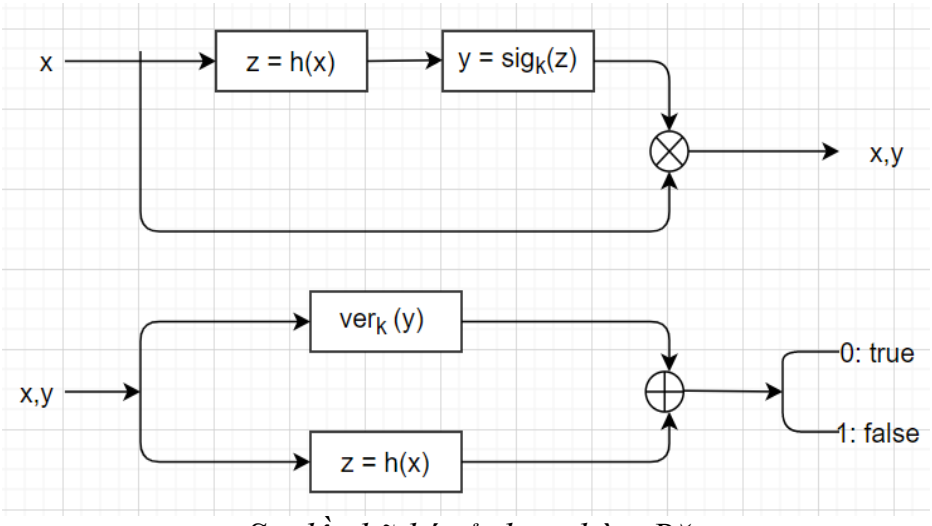


4. Vũ Hồng Phương - Tìm hiểu về hàm băm SHA256

4.1. Khái niệm.

Một hàm Băm H sẽ lấy ở đầu vào một thông tin X có kích thước biến thiên và sinh kết quả là một chuỗi có độ dài cố định, được gọi là cốt của bức điện (message digest).

Ví dụ như khi B muốn ký một bức điện x (độ dài bất kỳ), đầu tiên anh ta tính cốt của bức điện z = h (x) (độ dài cố định) và sau đó ký y = $sig_k(z)$. Anh ta phát cặp (x, y) lên kênh truyền, bây giờ việc kiểm tra có thể thực hiện bằng việc tính lại cốt của bức điện z = h (x), sau đó kiểm tra ver_k (z, y) có bằng true hay không.



Sơ đồ chữ ký sử dụng hàm Băm

4.2. Đặc tính của hàm băm.

Một vấn đề cần bản tiếp là tính đụng độ của hàm Băm. Theo nguyên lý Diricle: "nếu có n + 1 con thỏ được bỏ vảo n cái chuồng thì phải tồn tại ít nhất một cái chuồng mà trong đó có ít nhất là hai con thỏ ở chung". Rõ ràng với không gian giá trị Băm nhỏ hơn rất nhiều so với không gian tin về mặt kích thước thì chắc chắn sẽ tồn tại đụng độ, nghĩa là có hai tin $x \neq x'$ mà giá trị Băm của chúng là giống nhau, tức $h(x) = h(x')$.

Sau đây chúng ta sẽ xét các dạng tấn công có thể có, từ đó rút ra các tính chất của hàm băm:

Dạng tấn công thứ nhất là người C bắt đầu với một bức điện được ký có giá trị (x, y), trong đó $y = sig_k(h(x))$ (cặp (x, y) có thể là bất kỳ bức điện trước đó mà B đã ký). Sau đó, C tính $z = h(x)$ và cố gắng tìm $x' \neq x$ để $h(x') = h(x)$. Nếu C làm được điều này thì cặp (x', y) sẽ là một bức điện được ký có giá trị (một bức điện giả mạo có giá trị). Để ngăn cản việc này, hàm Băm h phải thỏa mãn tính chất sau:

Tính chất 1:

Một hàm băm h có tính phi đụng độ cao khi với một bức điện x cho trước, không tìm ra một bức điện $x' \neq x$ sao cho $h(x') = h(x)$.

Một dạng tấn công khác mà người C có thể làm là: đầu tiên anh ta tìm 2 bức điện $x \neq x'$ sao cho $h(x) = h(x')$. Sau đó C đưa bức điện x cho B và thuyết phục B

ký vào cốt bức điện h(x), vì vậy anh ta tìm được y. Như vậy cặp (x', y) là một cặp chữ ký giả có giá trị. Điều này là nguyên nhân mà việc thiết kế hàm Băm phải thỏa mãn tính chất 2 nhưu sau:

Tính chất 2:

Một hàm băm h có tính đụng độ cao khi không thể tìm ra những bức điện x và x' sao cho x' ≠ x và h(x') = h(x).

Dạng tấn công thứ 3 là chọn một giá trị cốt z ngẫu nhiên. Người C sẽ tính một chữ ký với một giá trị ngẫu nhiên z, sau đó anh ta tìm một bức điện x sao cho z = h(x). Nếu anh ta làm được điều này thì cặp (x, y) là cặp chữ ký giả có giá trị. Như vậy một tính chất nữa mà h cần thỏa mãn là tính một chiều:

Tính chất 3:

Một hàm Băm h có tính một chiều khi với cốt của một bức điện z cho trước không thể tìm được một bức điện x sao cho h(x) = z.

4.3. Hàm băm SHA (Secure Hash Algorithm)

SHA được thiết kế dựa trên những nguyên tắc của MD4/MD, tạo ra 160-bit giá trị Băm.

4.3.1. Miêu tả SHA

Cũng giống như với MD5, bức điện được cộng thêm một bit 1 và các bit 0 ở cuối bức điện để bức điện có thể chia hết cho 512. SHA sử dụng 5 thanh ghi dịch:

A = 0x67452301

B = 0xefcdab89

C = 0x98badcfe

D = 0x10325476

E = 0xc3d2e1f0

Bức điện được chia ra thành nhiều khối 512-bit. Ta cũng đặt là a, b, c, d, e thay cho A, B, C, D, E đối với khối 512-bit đầu tiên của bức điện. SHA có bốn vòng lặp chính trong 5 giá trị a, b, c, d và e, sau đó cũng được cộng và dịch như trong MD5.

SHA xác lập bốn hàm phi tuyến như sau:

$f_t(X, Y, Z) = (X \wedge Y) \vee ((\neg X) \wedge Z)$ với $0 <= t <= 19$

$f_t(X, Y, Z) = X \oplus Y \oplus Z$ với $20 <= t <= 39$

$f_t(X, Y, Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$ với $40 <= t <= 59$

$f_t(X, Y, Z) = X \oplus Y \oplus Z$ với $60 <= t <= 79$.

Bốn hằng số sử dụng trong thuật toán là:

$K_t = 2^{1/2}/4 = 0x5a827999$ với $0 <= t <= 19$

$K_t = 3^{1/2}/4 = 0x6ed9eba1$ với $20 <= t <= 39$

$K_t = 5^{1/2}/4 = 0x8f1bbcdc$ vưới $40 <= t <= 59$

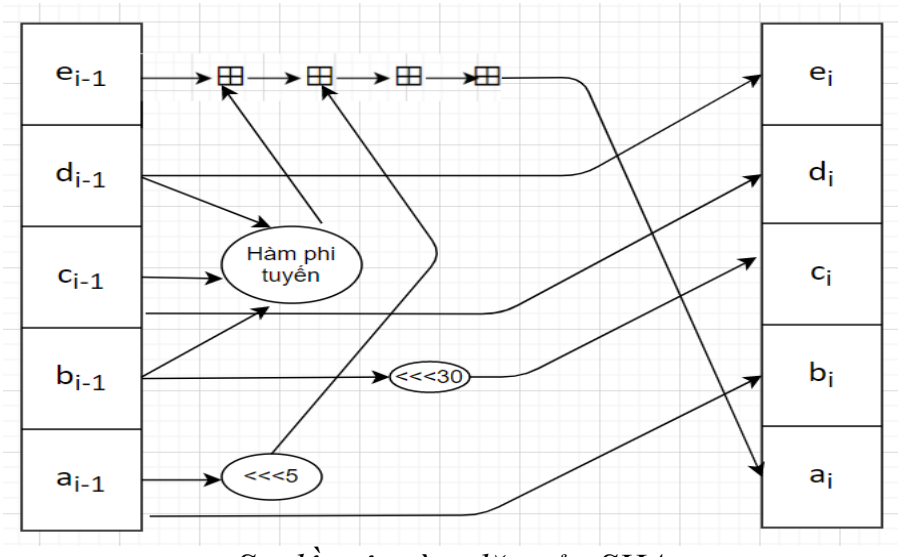
$K_t = 10^{1/2}/4 = 0xca62c1d6$ với $60 <= t <= 79$.

Các khối bức điện được mở rộng từ 16word đến 32-bit (M0 đến M15) thành 80 word 32-bit (W_0 đến W_{79}) bằng việc sử dụng thuật toán mở rộng:

$W_t = M_t$ với $0 <= t <= 15$

$W_t = (W_{t-3} \oplus W_{t-8} \oplus W_{t-16})$ với $16 <= t <= 79$.

Ta có thể miêu tả một vòng lặp của SHA như sau:



Sơ đồ một vòng lặp của SHA

Nếu gọi W_t là biểu diễn của khối con thứ t của bức điện được mở rộng, và <<<s là biểu diễn dịch trái s bit, thì vòng lặp chính của SHA như sau:

a = A, b = B, c = C, d = D, e = E,

for t = 0 to 79

{

TEMP = (a <<<5) + f_t(b, c, d) + e + W_t + K_b

e = d,

d = c,

c = b <<<30,

b = a,

a = TEMP,

}

A = A + a, B = B + b, C = C + c, D = D + d, E = E + e,

Thuật toán tiếp tục với khối 512-bit tiếp theo cho tới khi hết bức điện, và kết quả sau cùng trong 4 thanh ghi A, B, C, D và E chính là hàm Băm SHA 160-bit.

4.3.2. Tính bảo mật trong SHA:

Để hiểu rõ hơn về tính bảo mật của SHA, ta hãy so sánh SHA với MD5 để có thể tìm ra những điểm khác nhau của hai hàm Băm này:

- MD5 và SHA đều cộng thêm các bit "giả" để tạo thành những khối chia hết cho 512- bit, nhưng SHA sử dụng cùng một hàm phi tuyến f cho cả bốn vòng.
- MD5 sử dụng mỗi hằng số duy nhất cho mỗi bước biến đổi, SHA sử dụng mỗi hằng số cho mỗi vòng biến đổi, hằng số dịch này là một số nguyên tố đối với độ lớn của word (giống với MD4).
- Trong hàm phi tuyến thứ 2 của MD5 có sự cải tiến so với MD4, SHA thì sử dụng lại hàm phi tuyến của MD4, tức (X ∧ Y) ∨ (X ∧ Z) ∨ (Y ∧ Z).
- Trong MD5 với mỗi bước được cộng kết quả của bước trước đó. Sự khác biệt đối với SHA là cột thứ 5 được cộng (không phải b, c hay d như trong MD5), điều này làm cho phương pháp tấn công của Boer-Bosselaers đối với SHA bị thất bại (den Boer và Bosselaers là hai người đã phá thành công 2 vòng cuối trong MD4).

Cho đến nay, chưa có công bố nào được đưa ra trong việc tấn công SHA, bởi vì độ dài của hàm Băm SHA là 160-bit, nó có thể chống lại phương pháp tấn

công bằng vét cạn (kể cả Birthday attack) tốt hơn so với hàm Băm MD5 128-bit.

4.4. Một số ứng dụng của hàm băm.

Ứng dụng chính của hàm Băm là sử dụng với hệ chữ ký điện tử, trong đó thay vì ký trực tiếp lên các văn bản, thông điệp (mà trong đa số trường hợp là rất lớn, tốc độ chậm) người ta sẽ ký lên giá trị băm đại diện cho toàn bộ văn bản đó. Điều này đặc biệt quan trọng và hiệu quả bởi vì chúng ta biết rằng các hệ chữ ký điện tử đều làm việc với các phép tính số học số lớn nên bản thân chúng đã tương đối chậm, việc sử sụng giá trị băm thay cho toàn bộ văn bản là giải pháp toàn diện khắc phục được yếu điểm này của các hệ chữ ký điện tử. Ngoài việc sử dụng với các hệ chữ ký điện tử hàm Băm còn được sử dụng vào các mục đích khác như: xác thực hóa thông điệp, xác thực hóa người dùng.

Đối với các ứng dụng không cần giữ bí mật thông điệp mà chỉ cần đảm bảo thông điệp không bị thay đổi trên đường truyên người ta sẽ sử dụng hàm băm cho mục đích xác thực tính nguyên vẹn của thông điệp đó. Chẳng hạn chúng ta có một phần mềm mã nguồn mở ở dạng setup muốn phân phối cho người dùng, rõ ràng việc gửi phầm mềm đó tới máy tính của người dùng là không cần phải mã hóa, tuy nhiên nếu như phần mềm đó (khi đó phần mềm chính là thông điệp). Người dùng sẽ download cả phần mềm và giá trị băm nhận được, sau đó tiền hành băm lại, đối sánh hai giá trị này nếu khớp nhau thì có thể đảm bảo phần mềm không bị sửa đổi trên đường truyền. Hiện nay đa số các phần mềm mã nguồn mở đều được phân phối theo cách này.

Trong các hệ thống yêu cầu có xác thực người dùng như các hệ quản trị cơ sở dữ liệu, hệ điều hành, các ứng dụng web, ứng dụng dạng desktop application, để lưu mật khẩu người dùng người ta cũng sử dụng các hàm Băm hoặc các hệ mã trong các vai trò của hàm Băm (không sử dụng khóa). Khi đó mỗi tài khoản của người dùng thay vì lưu dưới dạng tên truy cập (user name) và mật khẩu (password) sẽ được lưu dưới dạng: tên người dùng, giá trị băm của mật khẩu. Khi một người dùng đăng nhập vào hệ thống, hệ thống sẽ lấy tên truy cập, mật khẩu họ nhập vào,

kiểm tra xem có tên truy cập nào như vậy hay không. Nếu có sẽ tiến hành băm giá trị mật khẩu do người dùng nhập vào.

4.5. Viết chương trình Demo với ngôn ngữ C++

- Bước 1: Nhập các thông số cần thiết để tạo khóa thỏa mãn:
  - Số nguyên tố p lớn và một phân tử nguyên thủy α.
  - Số nguyên bí mật a và tính $\beta \equiv \alpha z \pmod p$.
  - p, α và β được công khai và a được giữ kín.

```
void input(long long &p, long long &alpha, long long &a, long long &beta)
{
    cout << "Enter p: ";
    cin >> p;
    cout << "Enter alpha: ";
    cin >> alpha;
    cout << "Enter a: ";
    cin >> a;
    beta = power(alpha, a, p);
}
```

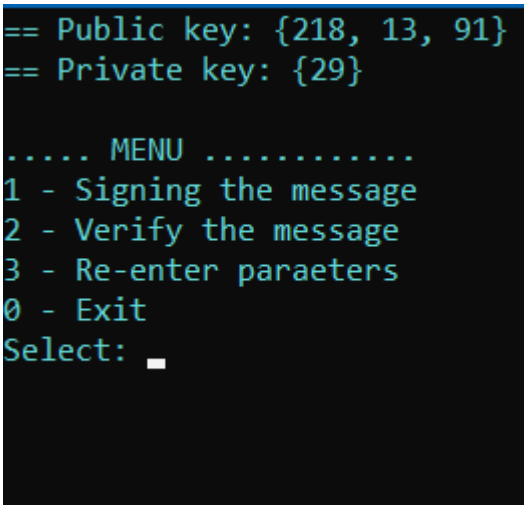
- Bước 2: Ký văn bản.
  m được đọc từ file mã hoá cần ký
  Random k ngẫu nhiêu sao cho GCD (k, p - 1) = 1.
  Tính $r \equiv \alpha^k \pmod p$.
  Tính $s \equiv k^{-1} (m - zr) \pmod{p - 1}$.
  r và s được lưu vào file lưu chữ ký.

```
long long rev_k = modInverse(k, p - 1);
long long r = power(alpha, k, p);
long long s = (rev_k * (m - a * r)) % (p - 1);

cout << "\n The digital signature of message: {"
<< r << ", " << s << "}\n";

string nameSigFile;
cout << "\n Save signature to: ";
cin >> nameSigFile;

ofstream outf(nameSigFile);
outf << r << "\n"
     << s;
cout << "\n Successfully save to " << nameSigFile
<< "\n";
outf.close();
```

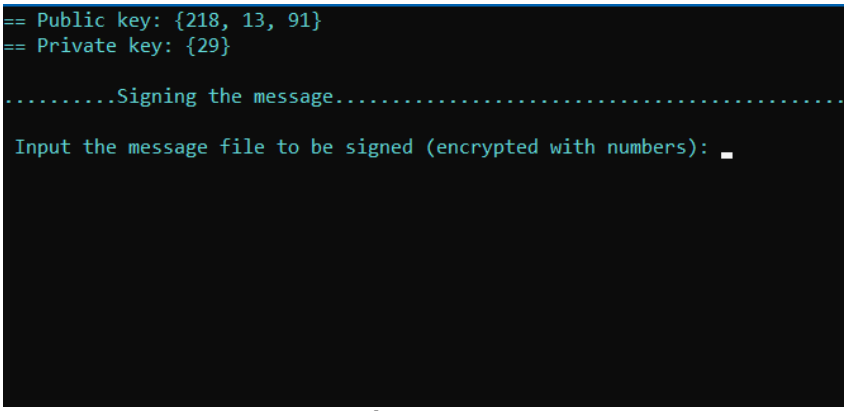
- Bước 3: Xác minh văn bản.
  m được đọc từ file mã hoá cần xác minh
  r và s được đọc từ file chữ ký
  so sánh $v_1 \equiv (\beta^r) * (r^s) \pmod p$ và $v_2 \equiv (\alpha^m) \pmod p$.

```
            v1 = (power(beta, r, p) * power(r, s, p)) % p;
            v2 = power(alpha, m, p);

            inf.close();
            flag = true;
        }
    } while (!flag);

    if (v1 == v2)
        cout << "\n True verification!";
    else
        cout << "\n False verification!";
```

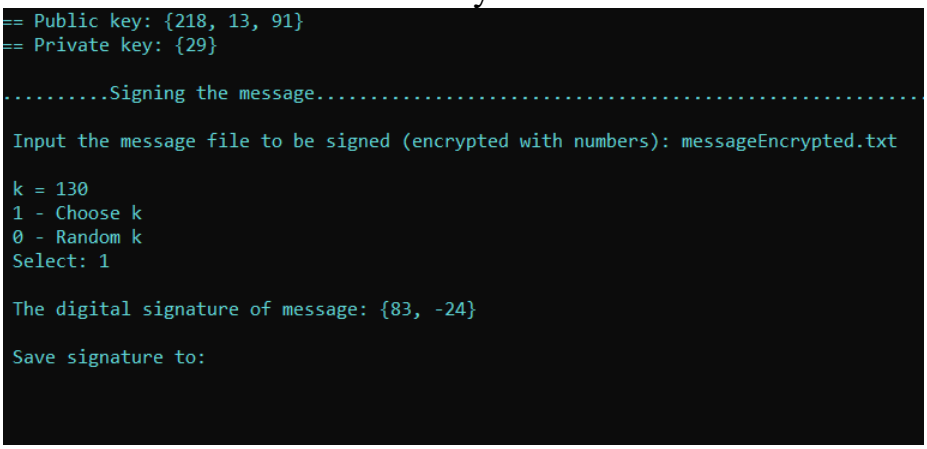
- Menu



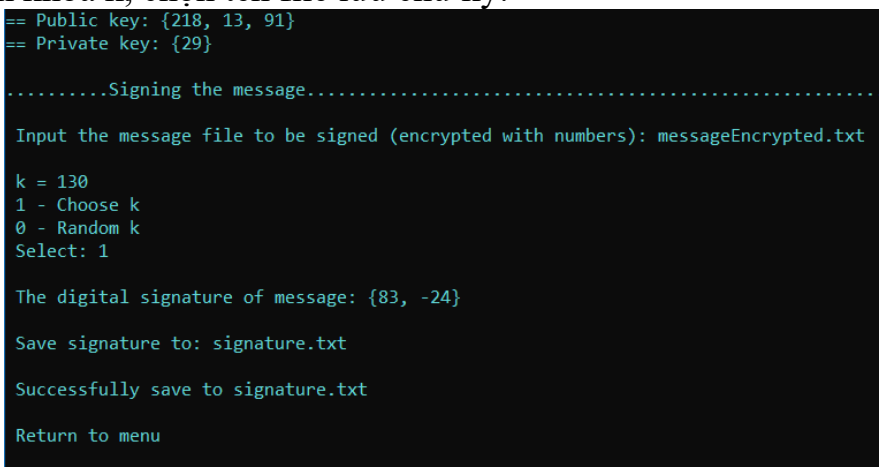
- Ký văn bản.



- Chon file văn bản đã mã hóa cần ký.

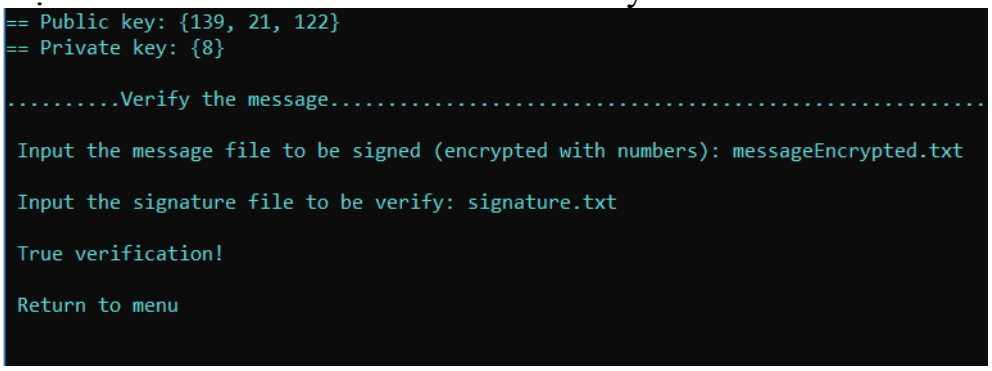


- Chon khóa k, chọn tên file lưu chữ ký.



- Xác minh văn bản

- Chọn file văn bản mã hoá và file lưu chữ ký



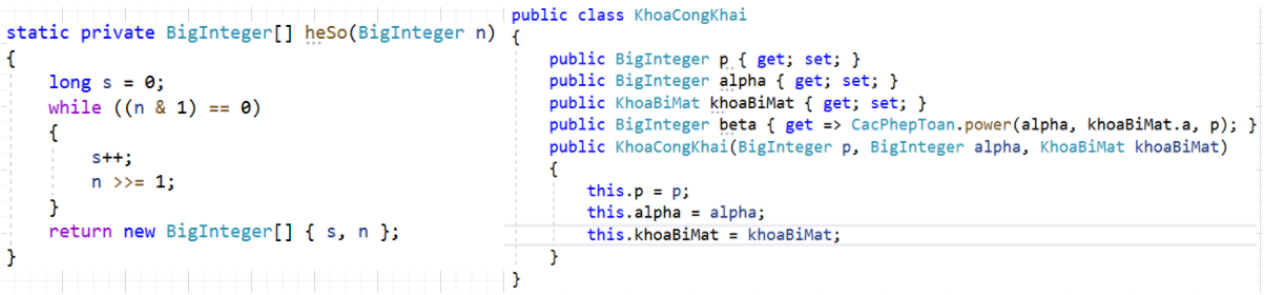
5. Ngô Văn Sáng – Ứng dụng xây dựng chương trình bằng ngôn ngữ C#.

5.1. Ý tưởng và thuật toán

- Vì tính bảo mật trong các giao dịch thương mại, từ việc xác minh chữ ký cho đến các thẻ tín dụng, các sơ đồ định danh và các sơ đồ chia sẻ bí mật nên phầm mềm tạo chữ ký Elgamal được xây dựng để dảo bảo tính an toàn trong quá trình chuyển file.
- Phần mềm cần tự tạo khóa công khai (p, α, β) và khóa bí mật (a).

```
static private BigInteger[] heSo(BigInteger n)
{
    long s = 0;
    while ((n & 1) == 0)
    {
        s++;
        n >>= 1;
    }
    return new BigInteger[] { s, n };
}
```

```
public class KhoaBiMat
{
    public BigInteger a;
    public KhoaBiMat(BigInteger a)
    {
        this.a = a;
    }
}
```



- Sau khi tạo khóa ngẫu nhiên ta tải file cần thực hiện ký, file sẽ hiển thị trong txtFile.

```
private void btnFileTHKy_Click(object sender, EventArgs e)
{
    if (ofdFileTHKy.ShowDialog() == DialogResult.OK)
    {
        string fileName = ofdFileTHKy.FileName;
        txtFile.Text = fileName;

        using (Stream stream = new FileStream(fileName, FileMode.Open))
        {
            SHA256Managed sha256Managed = new SHA256Managed();
            byte[] hashBytes = sha256Managed.ComputeHash(stream);
            string digest = Convert.ToBase64String(hashBytes);
            txtDigestSignToFile.Text = digest.ToString();
        }
    }
}
```

- Digest của tệp sử dụng hàm băm SHA256, tự động băm khi tải file cần thực hiện ký lên (ảnh trên).
- Tinh chữ ký là gamma, delta khi ấn vào button Tính toán sẽ cho ra kết quả ở hai ô textbox là txtGamma và txtDelta

```
private void btnTinhToan_Click(object sender, EventArgs e)
{
    if (kiemTraHopLe())
    {
        var p = BigInteger.Parse(txtPrimeP.Text);
        var alpha = BigInteger.Parse(txtAlpha.Text);
        var a = BigInteger.Parse(txtSoNguyena.Text);
        var k = BigInteger.Parse(txtRandomK.Text);
        quanlykhoaElgammal = new QuanLyKhoa(p, alpha, a);
        txtBeta.Text = quanlykhoaElgammal.khoaCongKhai.beta.ToString();


        chuKyElgammal.gamma = CacPhepToan.power(p, alpha, a);
        string digest = txtDigestSignToFile.Text;
        txtDigestSignToFile.Text = digest;
        var hashBytes = Convert.FromBase64String(digest);

        var x = new BigInteger(hashBytes.Concat(new byte[] { 0 }).ToArray());
        chuKyElgammal.gamma = CacPhepToan.power(alpha, k, p);
        chuKyElgammal.delta = ((x - a * chuKyElgammal.gamma) * CacPhepToan.inverseModulo(k, p - 1)) % (p - 1);
        if (chuKyElgammal.delta < 0) chuKyElgammal.delta += p - 1;
        txtGamma.Text = chuKyElgammal.gamma.ToString();
        txtDelta.Text = chuKyElgammal.delta.ToString();
        btnKyVB.Enabled = true;
    }
}
```

- Sau khi có chữ ký thì ký lên văn bản bằng button Ký văn bản và tự động lưu file chữ ký khi ký xong.

```
private void btnKyVB_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(txtGamma.Text) || string.IsNullOrEmpty(txtDelta.Text))
    {
        MessageBox.Show("Cần nhập đủ thông tin", "Thông báo");
        return;
    }
    try
    {
        string path = Path.ChangeExtension(ofdFileTHKy.FileName, "sig");
        if (File.Exists(path))
        {
            if(MessageBox.Show("File: "+Path.GetFileName(path)+ "đã có. Có muốn ghi đè?","File đã có", MessageBoxButtons
            {
                return;
            }
        }
        using (StreamWriter streamWrite = File.CreateText(path))
        {
            streamWrite.WriteLine(chuKyElgammal.gamma);
            streamWrite.WriteLine(chuKyElgammal.delta);
        }
        MessageBox.Show("Ký văn bản thành công");
    }
    catch
    {
        MessageBox.Show("Lỗi", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

- Sau khi ký thành công văn bản thì tải văn bản đã ký và cần xác nhận lên:

```
private void btnMoFileTHKiemTraKy_Click(object sender, EventArgs e)
{
    if (ofdMoFileTHKy.ShowDialog() == DialogResult.OK)
    {
        txtFileTHKiemTraKy.Text = ofdMoFileTHKy.FileName;
        string pathSignatureCheckAuto = Path.ChangeExtension(ofdMoFileTHKy.FileName, "sig");
        if (File.Exists(pathSignatureCheckAuto))
        {
            ofdMoFileChuKy.FileName = pathSignatureCheckAuto;
            txtFileChuKy.Text = pathSignatureCheckAuto;
        }
    }
}
```

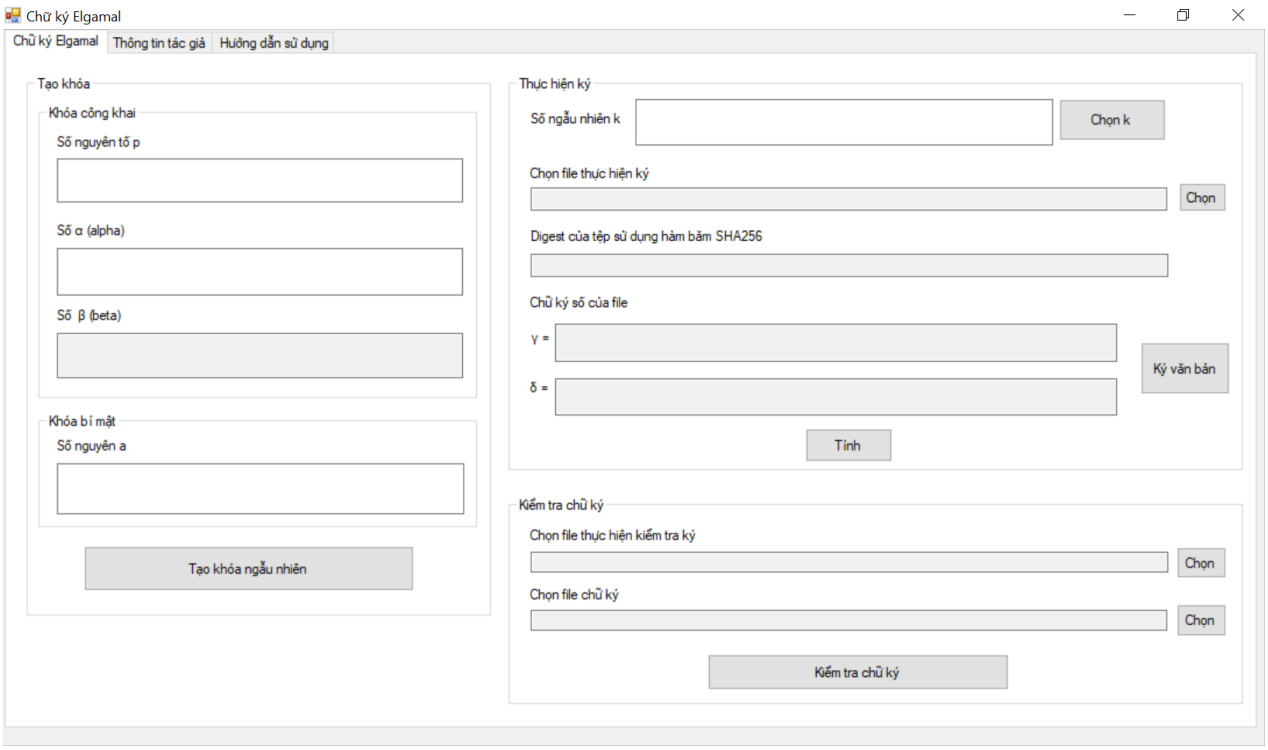
- Và tải file chữ ký kèm theo đã được ký ở văn bản:

```
private void btnMoFileChuKy_Click(object sender, EventArgs e)
{
    if (ofdMoFileChuKy.ShowDialog() == DialogResult.OK)
    {
        txtFileChuKy.Text = ofdMoFileChuKy.FileName;
    }
}
```

- Xác nhận chữ ký (Sử dụng hàm Băm SHA có trong visual studio)

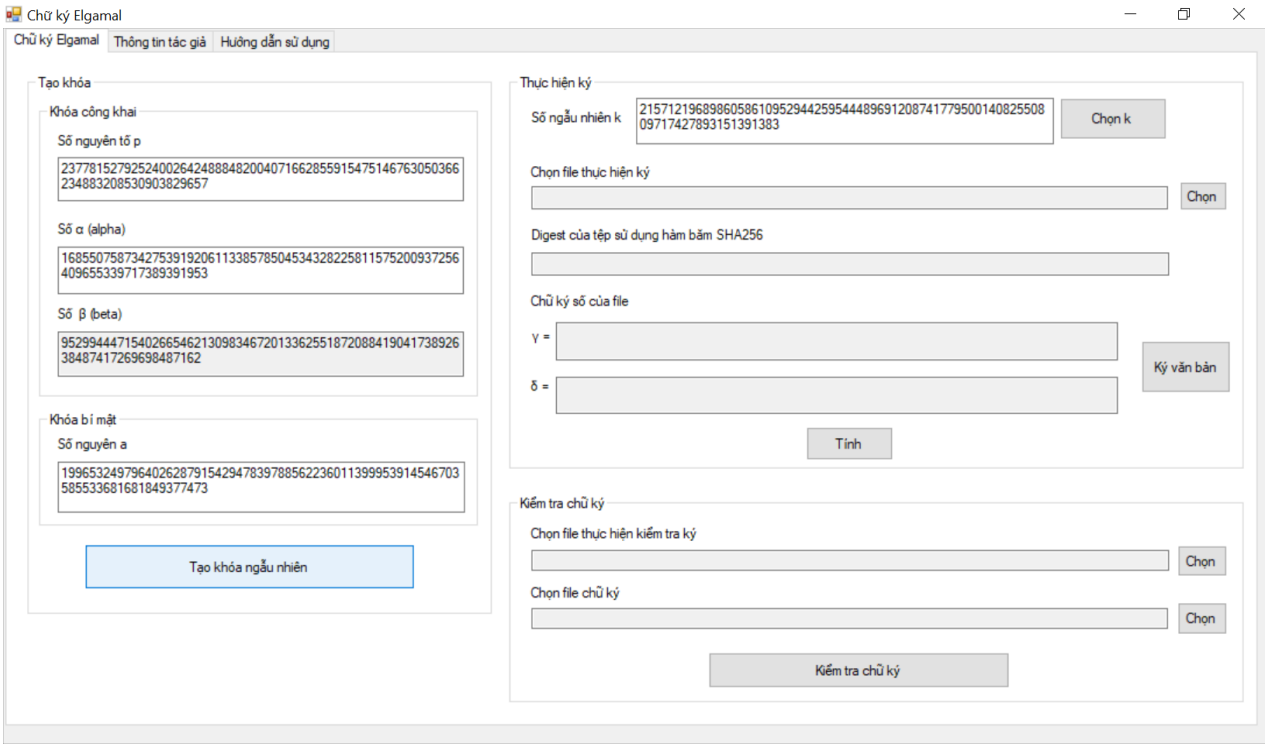
```
private void btnCheckChuKy_Click(object sender, EventArgs e)
{
    try
    {
        string digest;
        BigInteger gamma, delta;
        using (Stream stream = new FileStream(txtFileTHKiemTraKy.Text, FileMode.Open))
        {
            SHA256Managed sha256Managed = new SHA256Managed();
            byte[] hashBytes = sha256Managed.ComputeHash(stream);
            digest = Convert.ToBase64String(hashBytes);
        }
        using (StreamReader streamReader = new StreamReader(txtFileChuKy.Text))
        {
            gamma = BigInteger.Parse(streamReader.ReadLine());
            delta = BigInteger.Parse(streamReader.ReadLine());
        }
        ChuKyElgammal kiemTraChuKyElgammal = new ChuKyElgammal(gamma, delta);
        var check2 = (CacPhepToan.power(quanlykhoaElgammal.khoaCongKhai.beta, kiemTraChuKyElgammal.gamma, quanlykhoaElgammal.khoaCongKhai.p
            CacPhepToan.power(kiemTraChuKyElgammal.gamma, kiemTraChuKyElgammal.delta, quanlykhoaElgammal.khoaCongKhai.p)) % (quanlykhoaElga
        var dc = Convert.FromBase64String(digest);

        var x = new BigInteger(dc.Concat(new byte[] { 0x00 }).ToArray());
        var check1 = CacPhepToan.power(quanlykhoaElgammal.khoaCongKhai.alpha,x,quanlykhoaElgammal.khoaCongKhai.p);
        if(check1 != check2)
        {
            MessageBox.Show("Tài liệu gửi đến đã bị chỉnh sửa", "Thông báo");
        }
        else
        {
            MessageBox.Show("Tài liệu gửi đến không bị chỉnh sửa gì", "Thông báo");
        }

    }
    catch
    {
        MessageBox.Show("Lỗi trong khi đọc file", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

5.2. Giao diện sử dụng



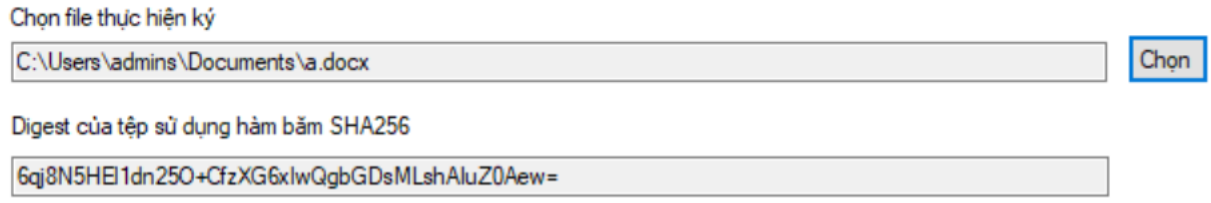
Hình 4.1 Giao diện sử dụng.

Sau khi tạo khóa và chọn k ngẫu nhiên:



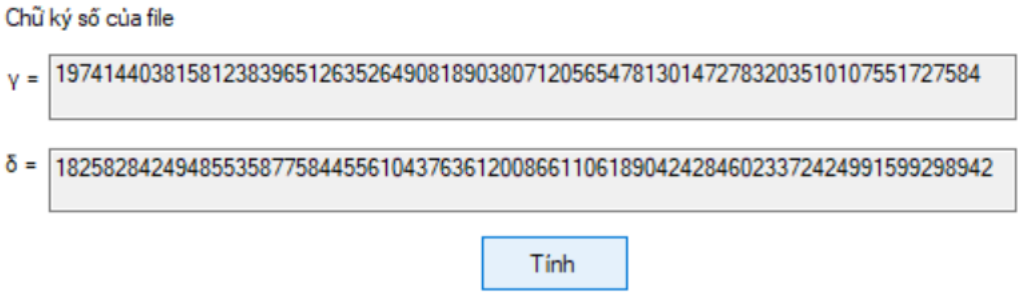
Hình 4.2 Tạo khóa và chọn k.

Chọn file thực hiện ký và băm:



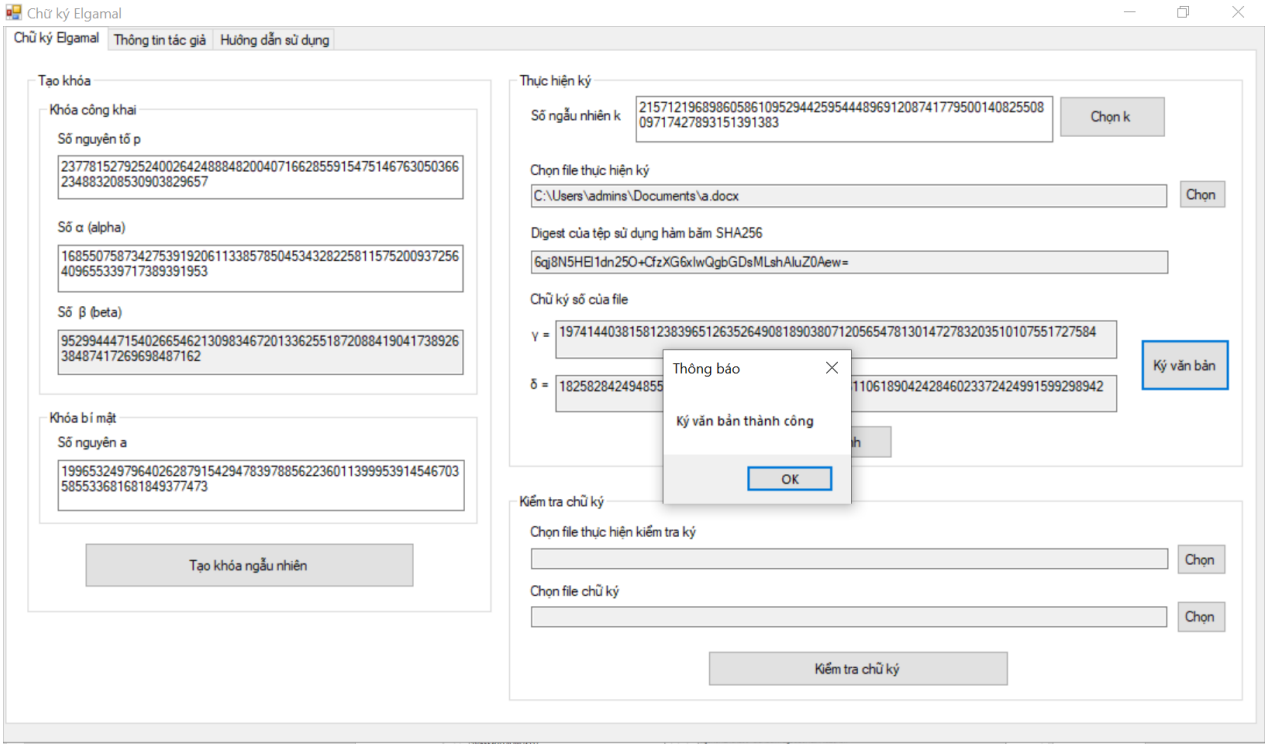
Hình 4.3 Tải văn bản cần ký.

Tạo chữ ký



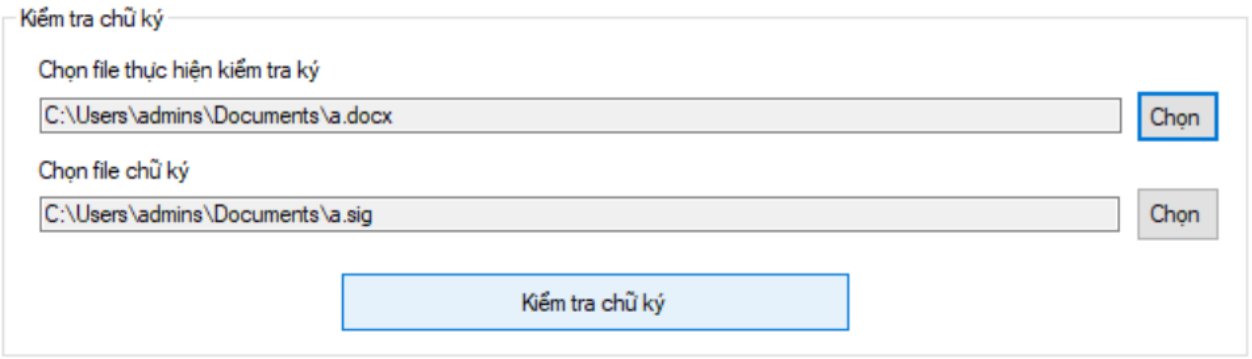
Hình 4.4 Tạo chữ ký.

Ký văn bản và đưa ra thông báo:

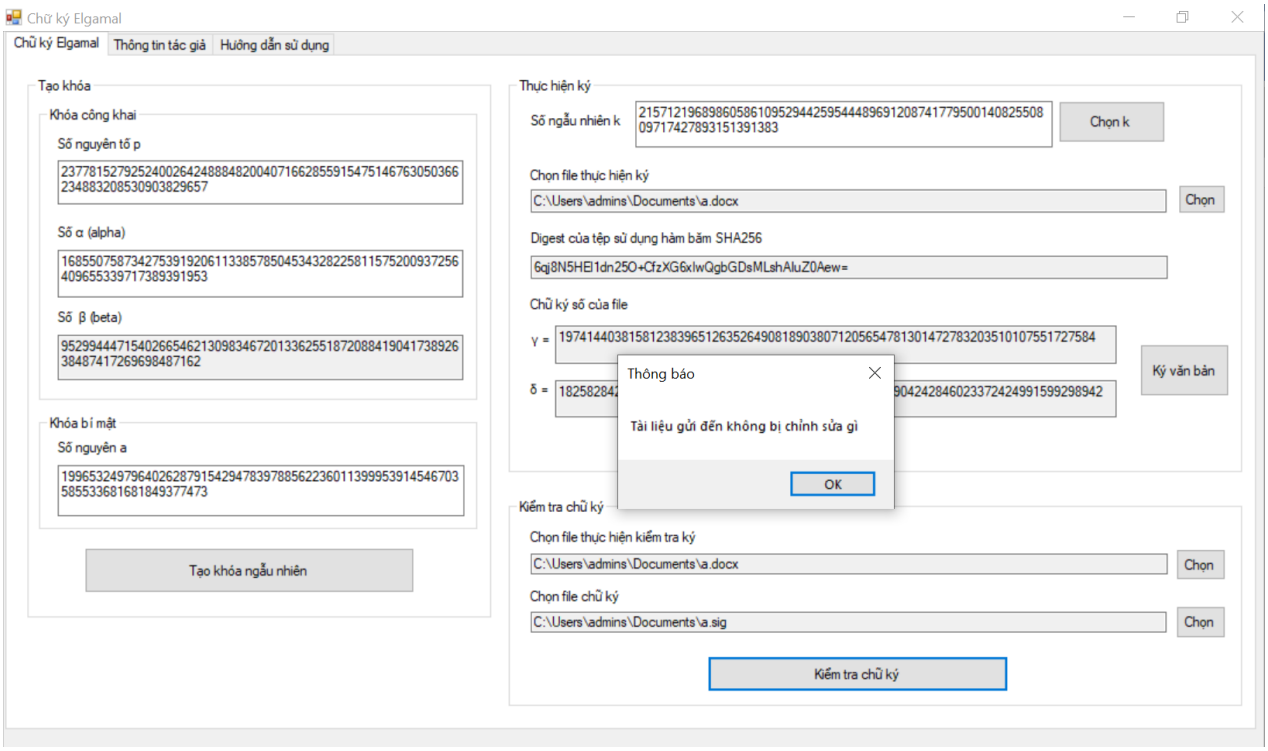


Hình 4.5 Ký văn bản.

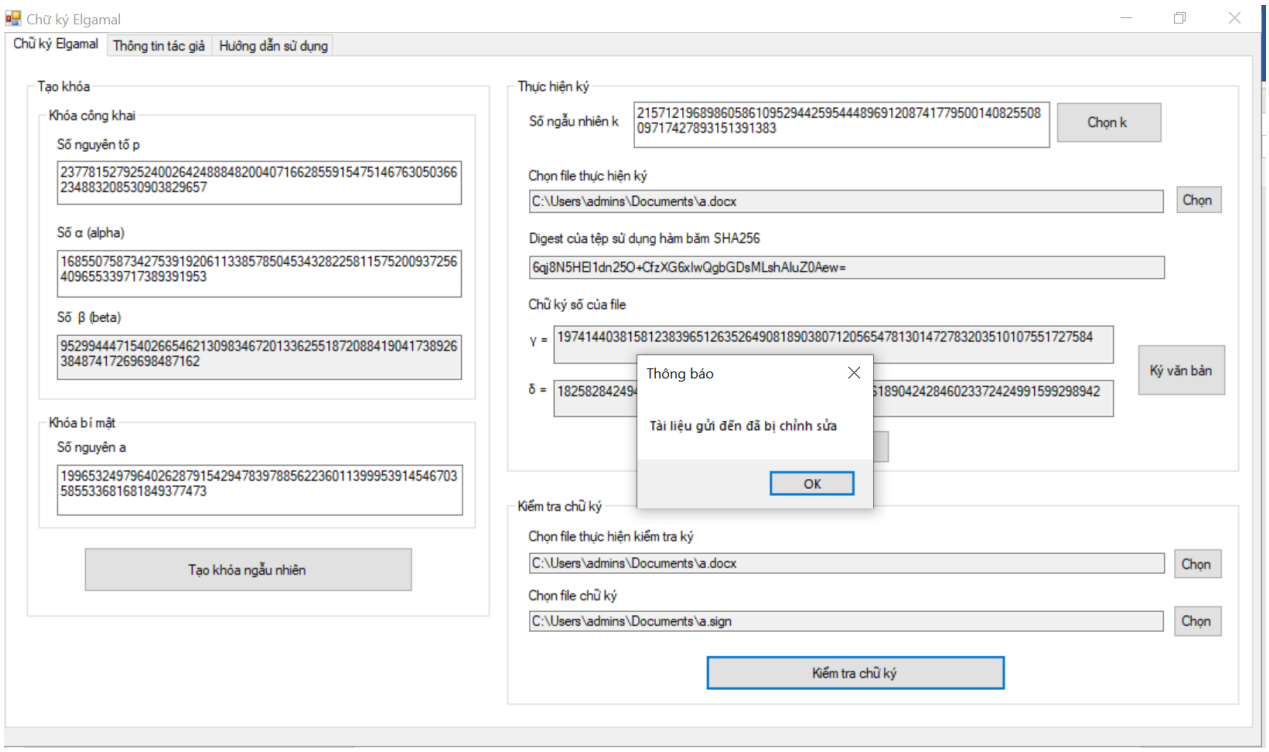
Xác nhận văn bản



Hình 4.6 Tải văn bản đã ký và chữ ký kèm theo.



Hình 4.7 Thông báo sau khi kiểm tra.



Hình 4.8 Thông báo sau khi kiểm tra.

Chương 3. PHẦN KIẾN THỨC LĨNH HỘI VÀ BÀI HỌC KINH NGHIỆM

I. NỘI DUNG ĐÃ THỰC HIỆN

Sau khi nhận được đề tài tìm hiểu về chữ ký điện tử Elgamal và viết ứng dụng minh họa, nhóm chúng em đã thực hiện được các nội dung sau:

- Đã tìm hiểu về chữ ký điện tử, chữ ký điện tử Elgamal.
- Tìm hiểu về phương pháp mã hóa bất đối xứng ứng dụng trong chữ ký điện tử.
- Tìm hiểu về hàm băm SHA gồm khái niệm, ứng dụng, thuật toán.

Ngoài ra, vì để viết ứng dụng minh họa, các thành viên trong nhóm đã tiến hành nghiên cứu về ngôn ngữ mà bản thân chọn để viết chương trình demo. Cụ thể nhóm đã nghiên cứu và trao đổi về các ngôn ngữ sau:

- C++
- C# (.Net)
- PHP
- Python

Sau khi tìm hiểu kỹ hơn về ngôn ngữ đã chọn, nhóm gặp phải 1 số khó khăn như đối với ngôn ngữ C++, các công cụ dùng để code ngôn ngữ này như CodeBlock hay DevC++ đều không thể thiết kế được giao diện, thành viên nhóm đã nghiên cứ tìm hiểu và cài đặt thành công MFC để thiết kế giao diện cho C++. Còn đối với ngôn ngữ Python cũng tương tự như C++, nhóm đã tìm hiểu và sử dụng thành công thư viện PyQt5 và QtDesigner để tạo giao diện cho chương trình demo.

Cũng qua lần thực hiện đề tài tìm hiểu về chữ ký điện tử Elgamal và viết chương trình ứng dụng, nhóm đã đạt được một số kỹ năng rất có ích cho học tập cũng như công việc sau này:

Kỹ năng làm việc nhóm: nhóm đã thảo luận về cách thực hiện chương trình, cách làm báo cáo bài tập lớn, …

Kỹ năng tư duy phản biện: các thành viên trong nhóm đều đưa ra quan điểm, ý kiến cá nhân để nhóm có hướng giải quyết vấn đề một cách tối ưu nhất có thể. Tuy nhiên, các thành viên rất mềm mỏng trong việc đưa ra ý kiến nên các cuộc tranh luận của nhóm đều đi đến kết quả tốt nhất và được tất cả các thành viên trong nhóm đồng ý.

Kỹ năng tra cứu thông tin: Vì đề tài về chữ ký điện tử Elgamal là một đề tài mới mẻ đối với nhóm nên nhóm đã phải sử dụng các công cụ hỗ trợ như google, các tài liệu cả tiếng anh cả tiếng việt để tra cứu thông tin. Thông tin tra cứu được của từng thành viên rất nhất quán, hỗ trợ cho nhau trong quá trình làm báo cáo bài tập lớn.

II. HƯỚNG PHÁT TRIỂN.

Trong thực tế, với những lợi thế về mặt bảo mật, an toàn, dễ triển khai trên hệ thống mạng không an toàn, chữ ký số được ứng dụng rất nhiều, cụ thể là trong việc mã hóa, chống giả mạo, xác thực, chống chối bỏ nguồn gốc, bảo mật phần mềm, bảo mật website, ...Với kết quả đã nghiên cứu về đề tài chữ ký số Elgamal, nhóm nhận thấy có thể đi sâu vào ứng dụng cho rất nhiều công nghệ trong mảng an toàn bảo mật thông tin. Hiện tại nhóm đã tìm hiểu về chữ ký số Elgamal và xây dựng được phần mềm ứng dụng bằng nhiều ngôn ngữ khác nhau: ngôn ngữ lập trình hướng đối tượng và mã nguồn mở đã giúp nhóm có được nền tảng ban đầu trong ngành an toàn và bảo mật thông tin.

Nhóm nhân thấy từ chữ ký Elgamal sử dụng phương pháp mã hóa bất đối xứng có thể được phát triển mạnh trong một số lĩnh vực như có thể sử dụng thay thế chữ ký tay trong tất cả các trường hợp giao dịch thương mại điện tử trong môi trường số :

Sử dụng chữ ký số điện tử trong các giao dịch thư điện tử, ký vào các email để các đối tác, khách hàng của bạn biết có phải bạn là người gửi thư không.

Sử dụng dụng chữ ký số điện tử để đầu tư chứng khoán trực tuyến, mua bán hàng trực tuyến, có thể dùng để thanh toán online, chuyển tiền trực tuyến mà không sợ bị mất cắp tiền như với đối với các tài khoản VISA, Master.

Các đối tác có thể ký hợp đồng kinh tế hoàn toàn trực tuyến không cần gặp mặt trực tiếp với nhau, chỉ cần ký vào file hợp đồng và gửi qua e-mail.

Dùng để kê khai, nộp thuế trực tuyến, khai báo hải quan và thông quan trực tuyến mà không phải mất thời gian in các tờ khai, trình ký đóng dấu đỏ của công ty rồi đến cơ quan thuế xếp hàng và ngồi đợi để nộp tờ khai này.

Bên cạnh đó, trong tương lai, các cơ quan chính phủ, nhà nước sẽ làm việc với nhân dân hoàn toàn trực tuyến và một cửa. Việc sử dụng chữ ký số sẽ giúp các cá nhân, tổ chức, doanh nghiệp dễ dàng sử dụng với các ứng dụng chính phủ điện tử, các cơ quan nhà nước khi cần làm thủ tục hành chính hay xin một xác nhận của cơ quan nhà nước để khai vào mẫu và ký số để gửi.

Một số ứng dụng chữ ký số điện tử điển hình:

Ứng dụng trong Chính phủ điện tử.

- Ứng dụng của Bộ Tài chính
- Ứng dụng của Bộ Công thương
- Ứng dụng của Bộ KHCN, …

Ứng dụng trong Thương mại điện tử.

- Mua bán, đặt hàng trực tuyến
- Thanh toán trực tuyến, …

Ứng dụng trong giao dịch trực tuyến.

- Giao dịch qua email

Hội nghị truyền hình và làm việc từ xa với Mega e-Meeting...

Với những ứng dụng hiện tại của chữ ký số, có thể thấy trong tương lai chữ ký số ngày càng trở nên quan trọng trong đời sống của con người, nếu phát triển chữ ký số trong việc bảo mật thông tin, đưa việc bảo mật thông tin của khách hàng, doanh nghiệp tư nhân, doanh nghiệp nhà nước và các cơ quan nhà nước lên

một tầng cao mới thì đó sẽ là bước nhảy vọt trong lĩnh vực an toàn bảo mật nói chung và trong thế giới công nghệ thông tin nói riêng. Đây chính là đích đến mà nhóm chúng em hướng đến trong tương lai.

Lời cảm ơn

Để hoàn thành bài tập lớn này chúng em xin gửi lời cảm ơn đến **Ths. Trần Phương Nhung** đã hướng dẫn chúng em thời gian qua giúp chúng em hoàn thành đề tài một cách thành công nhất.

Cuối cùng nhóm em mong nhận được sự góp ý của giảng viên, cũng như các ý kiến đóng góp của các bạn để chỉnh sửa, bổ sung cho đề tài của em ngày càng hoàn thiện và có ứng dụng sâu hơn nữa.

Chúng em xin chân thành cảm ơn!