TRƯỜNG ĐẠI HỌC HÀNG HẢI VIỆT NAM
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

AN TOÀN BẢO MẬT THÔNG TIN

MÔ PHỎNG
HỆ MÃ VÀ HỆ CHỮ KÝ ĐIỆN TỬ RSA

Giáo viên hướng dẫn: Đặng Hoàng Anh

Nhóm 1:

1. Vũ Đình Trung (Nhóm trưởng)
2. Chu Đức Thiện
3. Mạc Văn Nghĩa
4. Nguyễn Trung Kiên

Lớp : Cnt49 – Đh1.

Hải Phòng - 2012

MỤC LỤC

Chương 1: Cơ sở lý thuyết

1. Hệ mã RSA

Hệ mã RSA là hệ mã hóa công khai được đặt tên dựa theo các chữ cái đầu của 3 tác giả của hệ mã là Rivest, Shamir và Adleman.

Để cài đặt RSA ban đầu mỗi người dùng sinh khóa công khai và khóa bí mật của mình bằng cách:

- Chọn hai số nguyên tố lớn ngẫu nhiên khác nhau p và q
- Tính N = p*q
- Chọn một số e nhỏ hơn N và (e, ϕ(N)) = 1, e được gọi là số mũ lập mã
- Tìm phần tử ngược của e trên vành module ϕ(n) , d là số mũ giải mã
- Khóa công khai là K_P = (e, N)
- Khóa bí mật là $K_S = K^{-1}_P$ = (d, p, q)
- Mã hóa một thông điệp M: $C = M^e \pmod N$ (0 <= M < N)
- Giải mã: $M = C^d \pmod N$

Độ an toàn của hệ mã RSA phụ thuộc vào độ khó của việc phân tích N ra thừa số nguyên tố để tính ϕ(n).

Ưu điểm:

- Trong các hệ mật mã RSA, một bản tin có thể được mã hóa trong thời gian tuyến tính
- Các khóa cho hệ mã hóa RSA có thể được tạo ra mà không phải tính toán quá nhiều
- Tính bảo mật cao

Nhược điểm:

- Tốc độ mã hóa chậm
- Dung lượng trên đường truyền lớn

- Tạo lỗ hổng trong phân phối khóa công khai: tấn công đứng giữa (*man - in - the - middle attack*)

2. Hệ chữ ký RSA

Dựa vào ưu điểm của hệ mã RSA, nếu thiết lập được sơ đồ chữ ký dựa trên bài toán phân tích ra thừa số nguyên tố thì độ an toàn của chữ ký sẽ rất cao. Việc thiết lập sơ đồ xác thực chữ ký RSA rất đơn giản, ta chỉ cần đảo ngược hàm mã hóa và giải mã. Sau đây là sơ đồ chữ ký RSA :

- Cho n = p*q, trong đó p, q là các số nguyên tố. Đặt P = A = Z_n và định nghĩa K = {(n, p, q, a, b) : n = p*q, p và q là các số nguyên tố, a*b ≡ 1(mod ϕ(n))}.
- Các giá trị n và b là khóa công khai ; còn p, q là khóa bí mật.
- Với k = (n, p, q, a, b), ta xác định :

  + $sig_k(x) = x^a \bmod n$

  + $ver_k(x, y) = TRUE \Leftrightarrow x \equiv y^b \pmod n$ với $x, y \in Z_n$

Thông thường, chữ ký được kết hợp với hàm mã hóa công khai. Giả sử A muốn gửi một bức điện được mã hóa và đã được ký đến cho B. Với bản rõ x cho trước, A sẽ tính toán chữ ký của mình $y = sig_A(x)$ và sau đó mã hóa cả x, y sử dụng khóa công khai e_B của B, kết quả nhận được là $z = e_B(x, y)$. Bản mã z sẽ được gửi tới B, khi B nhận được z, đầu tiên anh ta giải mã với hàm giải mã d_B của mình để nhận được (x, y). Sau đó anh ta dùng hàm xác minh công khai của A để kiểm tra xem $ver_A(x, y) = TRUE$ hay không.

Chương 2: Khảo sát thực tế và xác lập dự án

1. Nhu cầu thực tiễn

Trong quá trình học tập môn An toàn bảo mật thông tin, nhiều khi chúng ta phải tính toán với những số mũ rất lớn không thể tính bằng tay hay dùng ứng dụng máy tính của Windows được.

Các bài tập liên quan đến RSA thường xuyên phải tính toán với số mũ khá lớn, do đó chúng ta rất cần một chương trình máy tính để tiện cho việc học tập tính toán một cách nhanh chóng.

2. Giải pháp

Từ nhu cầu thực tiễn đòi hỏi chúng ta phải xây dựng một ứng dụng mô phỏng hệ mã và hệ chữ ký điện tử RSA.

3. Thành lập nhóm phát triển

Nhóm phát triển ứng dụng mô phỏng hệ mã và hệ chữ ký điện tử RSA gồm 4 thành viên với bảng phân công công việc như sau:

Họ tên	Vai trò	Công việc
Vũ Đình Trung	Trưởng nhóm	- Khảo sát - Phân tích thiết kế - Viết báo cáo - Hỗ trợ kỹ thuật - Kiểm tra và hoàn thiện chương trình
Chu Đức Thiện	Thành viên	- Cài đặt các nút sinh - Cài đặt các nút mã hóa và giải mã
Nguyễn Trung Kiên	Thành viên	- Cài đặt các nút tính và kiểm tra - Cài đặt các nút ký và xác thực chữ ký
Mạc Văn Nghĩa	Thành viên	- Cài đặt các nút xem, mở và lưu

Chương 3: Phân tích và thiết kế

1. Đặc tả yêu cầu

1.1. Mục đích

Xây dựng một chương trình mô phỏng hệ mã và hệ chữ ký điện tử RSA đáp ứng nhu cầu học thuật.

1.2. Mô tả tổng quan

- Chương trình dùng để mô phỏng hệ mã và hệ chữ ký điện tử RSA.
- Có giao diện người dùng trực quan, dễ sử dụng.
- Hỗ trợ tính toán trên số nguyên lớn.
- Hỗ trợ cả tự động tính toán và cho phép người dùng tự nhập và có kiểm tra tính hợp lệ của các yếu tố đầu vào.
- Cho phép mở và lưu các yếu tố đầu vào và đầu ra.

1.3. Yêu cầu chức năng

- Cho phép tự nhập hoặc sinh ngẫu nhiên các số nguyên tố lớn p, q với số bit được nhập vào tối đa là 1024 bit cũng như tự động kiểm tra xem p, q có phải là số nguyên tố hay không.
- Tự động tính ϕ(n) = (p-1)*(q-1) mỗi khi thay đổi p hoặc q.
- Cho phép tự nhập hoặc tính toán n = p*q cũng như kiểm tra xem n có bằng p*q hay không.
- Cho phép tự nhập hoặc tính toán e với e là số nguyên tố cùng nhau với ϕ(n) và < n cũng như kiểm tra xem e có đúng là số nguyên tố cùng nhau với ϕ(n) hay không

- Cho phép tự nhập hoặc tính toán d với d là phần tử ngược của e trên vành $Z_{\phi(n)}$ và < n cũng như kiểm tra xem d có đúng là phần tử ngược của e trên vành $Z_{\phi(n)}$ hay không.
- Cho phép xem khóa công khai (e, n) và khóa bí mật (d, p, q).
- Cho phép mở file chứa khóa công khai (e, n) và khóa bí mật (d, p, q) có kiểm tra tính hợp lệ của dữ liệu.
- Cho phép lưu khóa công khai và khóa bí mật.
- Cho phép mở file chứa nội dung bản rõ, bản mã.
- Cho phép lưu file chứa nội dung bản rõ, bản mã, bản giải mã.
- Cho phép so sánh nội dung bản giải mã và bản rõ.
- Cho phép mã hóa bản rõ, giải mã bản mã, ký lên bản rõ và xác thực chữ ký trên bản mã.

1.4. Yêu cầu phi chức năng

1.4.1. Yêu cầu về độ tin cậy và hiệu suất

Phải đáp ứng được đầy đủ các chức năng, đặc biệt là chức năng mã hóa và giải mã, ký và xác thực chữ ký phải nhanh và chính xác.

1.4.2. Yêu cầu về môi trường

Phần mềm hoạt động trên môi trường từ Microsoft Windows XP SP2 và Net framework 2.0 trở lên.

1.4.3. Yêu cầu về giao diện

Giao diện được trình bày khoa học, hợp lý và đảm bảo mỹ thuật hài hòa với mục đích của phần mềm, tuân thủ các chuẩn về truy cập thông tin.

1.4.4. Thành phần bên ngoài

Có thể sử dụng thư viện xử lý số lớn mã nguồn mở bên ngoài.

2. Thiết kế giao diện và tương tác người dùng

Thiết kế toàn bộ các chức năng trên cùng một form. Bố cục form chia ra làm 3 phần chính theo chiều ngang:

2.1. Phần 1 - bên trái (60%):

Hiển thị giao diện của các yếu tố đầu vào và chia tiếp làm 3 phần theo chiều dọc, phần trên là giao diện nhập các số nguyên tố lớn p và q, phần giữa là nhập n và hiển thị ϕ(n), phần dưới là giao diện nhập e và d.

- Giao diện nhập **p, q**: Trên cùng bên trái là một NumericUpdown để nhập số bit, trên cùng bên phải là nút Sinh để sinh ngẫu nhiên một số nguyên tố, bên dưới là một RichTextBox để hiển thị dữ liệu.
- Giao diện nhập **n**: Trên cùng bên trái là nút Kiểm tra để kiểm tra xem n có bằng p*q hay không, trên cùng bên phải là nút Tính để tính n = p*q, bên dưới là một RichTextBox để hiển thị dữ liệu.
- Giao diện **ϕ(n)**: Một RichTextBox để hiển thị dữ liệu.
- Giao diện nhập **e**: Trên cùng bên trái là nút Kiểm tra để kiểm tra xem e có nguyên tố cùng nhau với ϕ(n) hay không, trên cùng bên phải là nút Sinh để sinh ngẫu nhiên một số e nguyên tố cùng nhau với ϕ(n), bên dưới là một RichTextBox để hiển thị dữ liệu.
- Giao diện nhập **d**: Trên cùng bên trái là nút Kiểm tra để kiểm tra xem d có là phần tử ngược của e hay không, trên cùng bên phải là nút Tính để tính d là phần tử nghịch đảo của e, bên dưới là một RichTextBox để hiển thị dữ liệu.

2.2. Phần 2 - giữa (10%):

Hiển thị các nút lệnh chính bao gồm: Mã hóa, Giải mã, Ký, Xác thực, Xem khóa công khai, Xem khóa bí mật, Mở khóa công khai, Mở khóa bí mật, Lưu khóa công khai, Lưu khóa bí mật.

2.3. Phần 3 - bên phải (30%):

Hiển thị giao diện Bản rõ, Bản mã, Bản giải mã.

- Giao diện nhập **Bản rõ**: Trên cùng bên trái là nút Mở để mở file bản rõ, trên cùng bên phải là nút Lưu để lưu bản rõ, bên dưới là một RichTextBox để hiển thị dữ liệu.

- Giao diện nhập **Bản mã**: Trên cùng bên trái là nút Mở để mở file bản mã, trên cùng bên phải là nút Lưu để lưu bản mã, bên dưới là một RichTextBox để hiển thị dữ liệu.
- Giao diện **Bản giải mã**: Trên cùng bên trái là nút So sánh để kiểm tra xem bản giải mã có giống bản rõ hay không, trên cùng bên phải là nút Lưu để lưu bản giải mã, bên dưới là một RichTextBox để hiển thị dữ liệu.

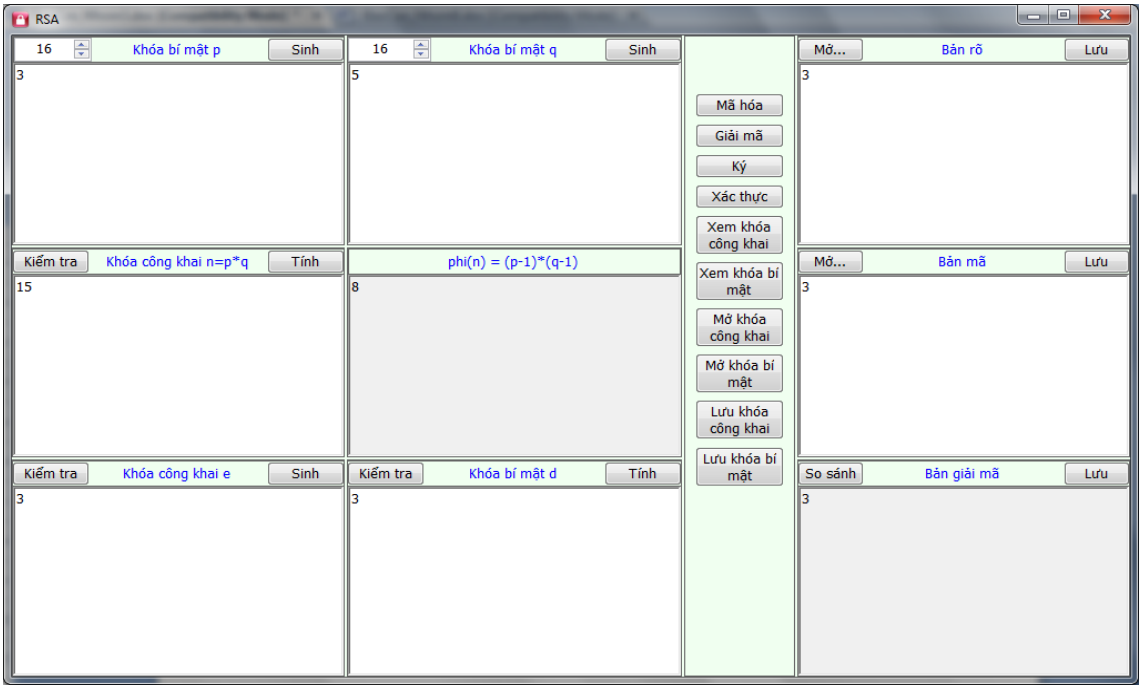
Chương 4: Cài đặt

1. Các công cụ sử dụng

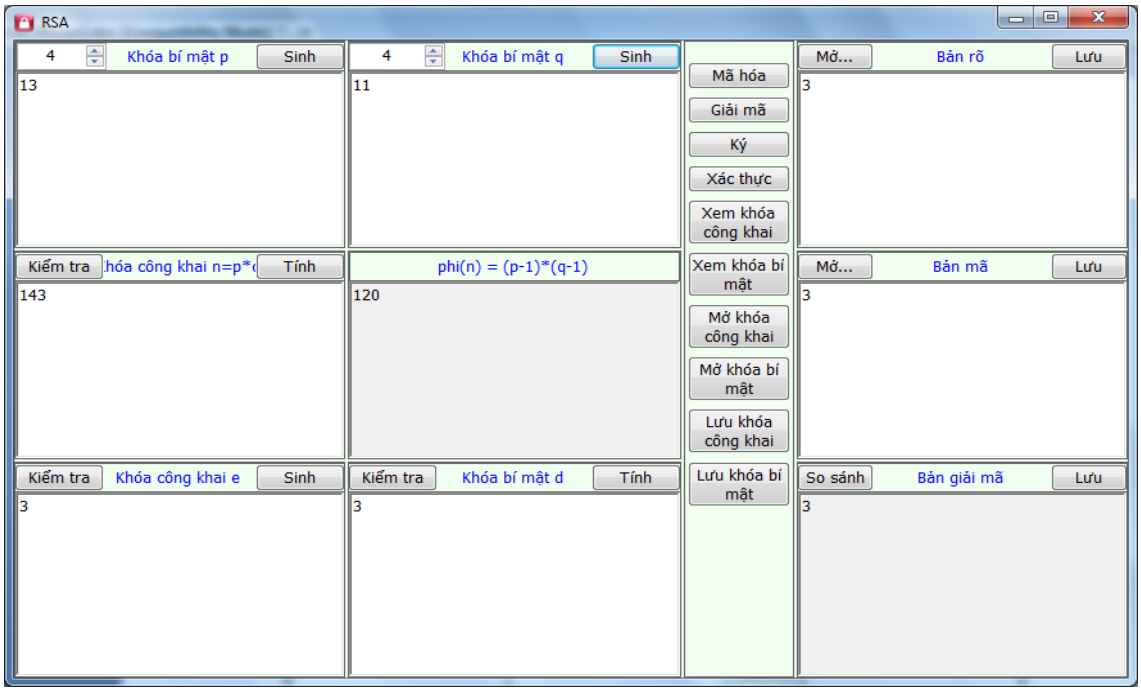
- Microsoft Visual Studio 2008 và 2010, ngôn ngữ C#, netframe work 2.0
- Thư viện BigInteger (codeproject)

2. Các giao diện

2.1. Giao diện chính

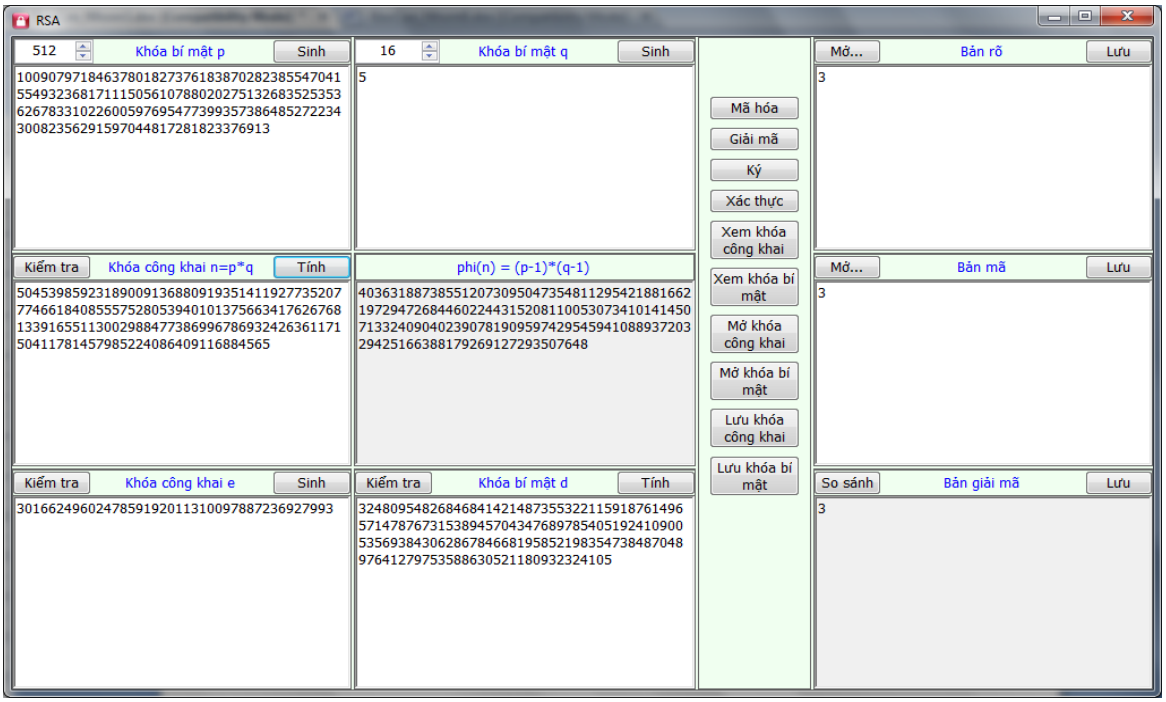


2.2. Giao diện sinh số nguyên tố



Góc trên bên trái, nhập số bit của số nguyên tố p muốn sinh, sau đó nhấn nút Sinh. Kết quả sẽ hiển thị ở hộp văn bản bên dưới. Đồng thời n và phi cũng sẽ tự động được tính.

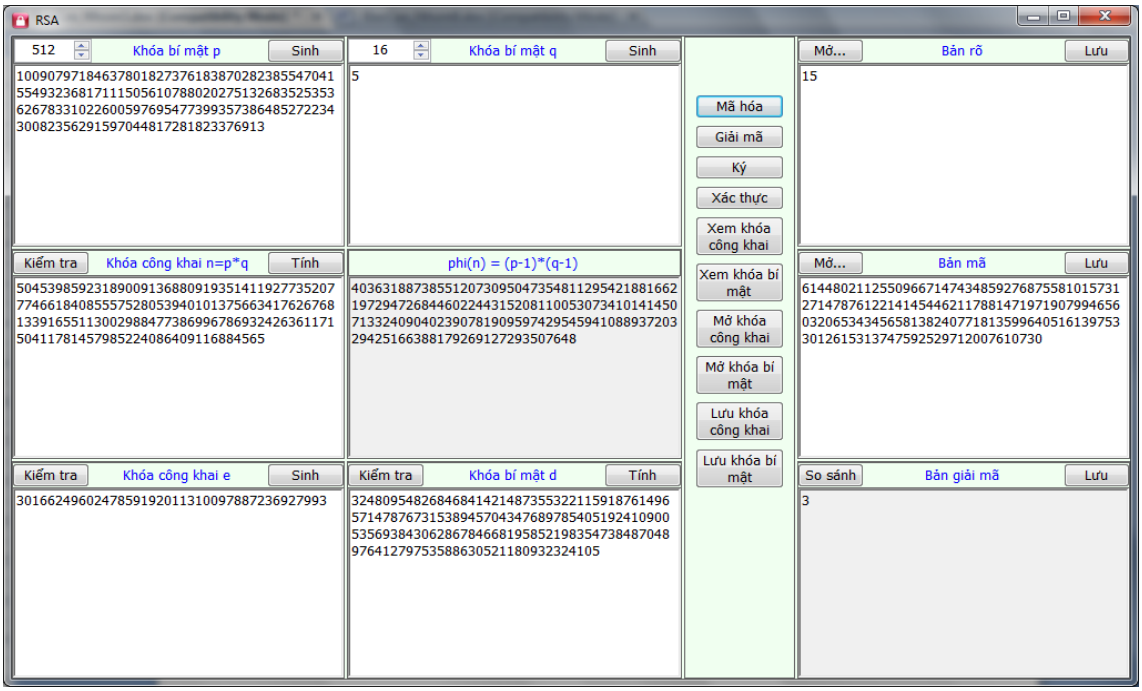
2.3. Giao diện tính n, sinh e và tính d



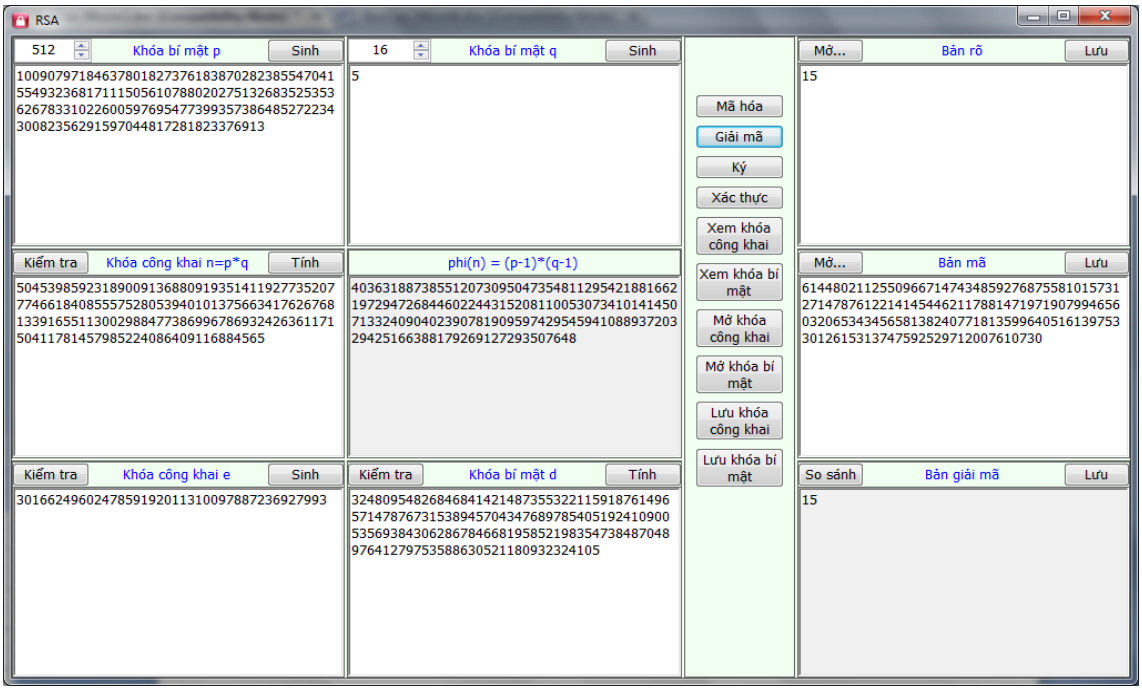
- Số n có thể được tính tự động khi p hoặc q thay đổi cũng có thể nhập bằng tay.

- Nhấn nút Kiểm tra để kiểm tra xem n có bằng p*q hay không.
- Nhấn nút Tính để tính lại n=p*q.
- Sau khi có n, nhập tiếp khóa công khai e.
- Nhấn nút Kiểm tra để kiểm tra xem e có là số nguyên tố cùng nhau với phi hay không.
- Nhấn nút sinh để tự động sinh một số e ngẫu nhiên là số nguyên tố cùng nhau với phi và e < phi.
- Nhấn nút Kiểm tra để kiểm tra xem khóa bí mật d có là phần tử nghịch đảo của e hay không
- Nhấn nút Tính để tính d.

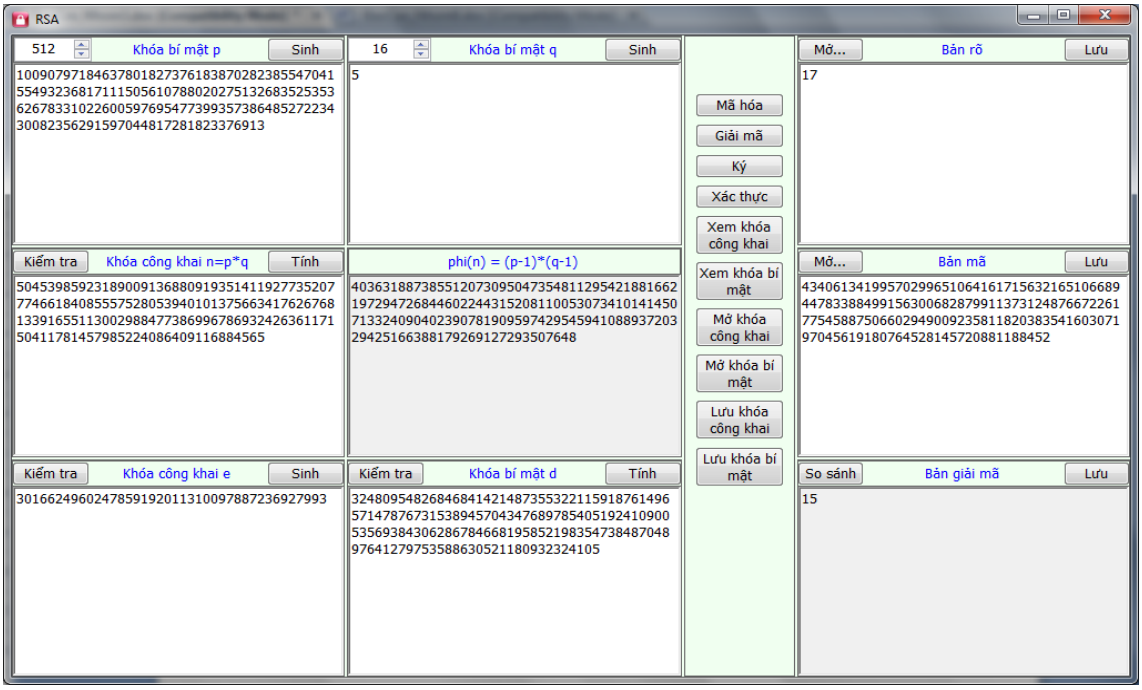
2.4. Giao diện sau khi mã hóa



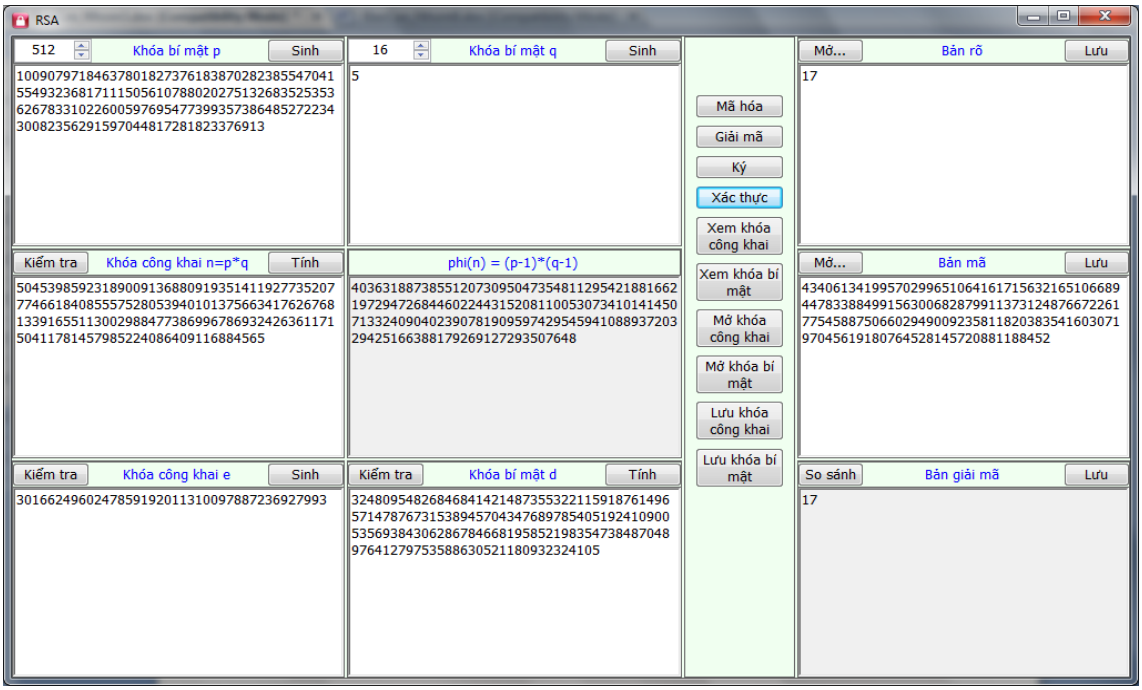
2.5. Giao diện sau khi giải mã



2.6. Giao diện sau khi ký

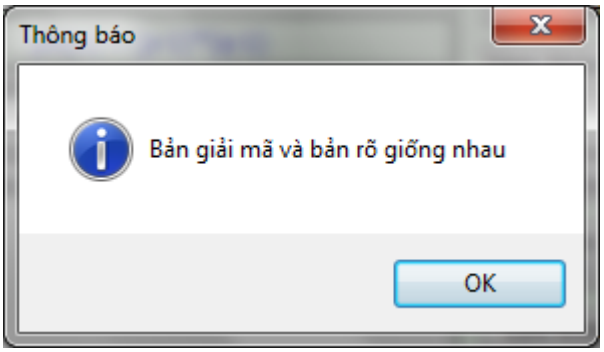


2.7. Giao diện sau khi xác thực chữ ký

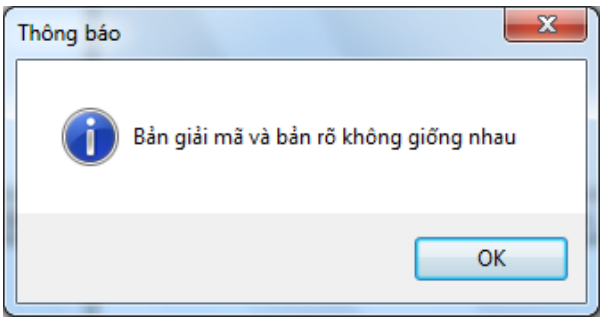


2.8. Giao diện so sánh bản giải mã và bản rõ

Nhấn vào nút so sánh để so sánh bản giải mã và bản rõ, nếu giống nhau sẽ hiện ra cửa sổ thông báo sau:



Ngược lại sẽ hiện cửa sổ thông báo sau:



3. Các mã lệnh và thuật toán

3.1. Hàm khởi tạo số nguyên lớn từ một chuỗi

```
BigInteger(string s, int radix);
```

Với s là chuỗi số nguyên lớn và radix là hệ cơ số (chẳng hạn 10 => hệ thập phân). Ví dụ:

```
BigInteger k = new BigInteger("123456", 10); //k=123456
```

3.2. Hàm sinh ngẫu nhiên một số nguyên tố lớn

```
static BigInteger genPseudoPrime(int bits, int confidence, Random rand);
```

Sinh ngẫu nhiên một số nguyên tố lớn có số bit là bits, độ tin cậy của thuật toán xác suất kiểm tra số nguyên tố Rabin Miller là confidence và độ ngẫu nhiên là rand.

```
// Ví dụ: sinh ngẫu nhiên số nguyên tố dài 512 bit, độ tin cậy là 50.
BigInteger p = BigInteger.genPseudoPrime(512, 50, new Random());
```

3.3. Hàm tính ước số chung lớn nhất

```
public BigInteger GCD(BigInteger bi)
// Tính ước số chung lớn nhất của hai số nguyên k và phi
k.GCD(phi);
```

3.4. Hàm tính phần tử ngược

```
public BigInteger modInverse(BigInteger modulus)
// Tìm d là phần tử ngược của e trên vành số nguyên phi
d = e.modInverse(phi);
```

3.5. Hàm tính a^m mod n

```
public BigInteger modPow(BigInteger exp, BigInteger n)
// Tìm m = m^e mod n
Ví dụ: m = m.modPow(e, n);
```

Tài liệu tham khảo

1. Giáo trình An toàn bảo mật thông tin - ThS. Nguyễn Hữu Tuân
2. BigInteger Class của Chew Keong TAN - http://www.codeproject.com/Articles/2728/C-BigInteger-Class
3. Wikipedia
4. http://www.scribd.com/doc/16652505/bo-mt-trong-h-thng-truyn-tin-s-dng-RSA-DES
5. Giáo trình An toàn bảo mật thông tin trường Đại học Kinh tế kỹ thuật Công nghiệp - Khoa CNTT - Bộ môn kỹ thuật máy tính: http://www.scribd.com/acuvodoi/d/87419018/11-V-2-RSA