Chữ ký số RSA G05 Csattd 19 3

Quản Lý An Toàn Thông Tin (Học viện Công nghệ Bưu chính Viễn thông)

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN 1

BÁO CÁO TIỂU LUẬN

Môn: CƠ SỞ AN TOÀN THÔNG TIN

Đề tài: Tìm hiểu về giải thuật tạo chữ ký số sử dụng RSA

Giảng viên: Hoàng Xuân Dậu

Nhóm 5 – Lớp CSATT03 (2021-2022)

1. Trần Quốc Hoàn
2. Phan Thị Kim Hoàn
3. Phạm Minh Hiếu
4. Đỗ Phạm Hòa

Hà Nội 9/2021

Tìm hiểu về giải thuật tạo chữ ký số sử dụng RSA

I - Giới Thiệu

Hiện nay khoa học công nghệ ngày càng hiện đại thì nhu cầu tiết kiệm thời gian, công sức trong các công việc là điều cấp thiết. ***Chữ ký số*** *ra đời giúp các tổ chức, cá nhân thỏa mãn được yêu cầu trên*

Chữ ký số **(digital signature)** là tập con của chữ ký điện tử.

- Chữ ký điện tử (**electronic signature)** là thông tin đi kèm theo dữ liệu (bao gồm văn bản, hình ảnh, video…) với công dụng là xác định chủ dữ liệu.

- Hiểu đơn giản chữ ký số như 1 con dấu online chứa thông tin của chính chủ, rất khó để giả mạo, bắt chước.

- RSA là một hệ mã hóa bất đối xứng được phát triển bởi Ron Rivest, Adi Shamir và Leonard Adleman (tên của nó cũng chính là tên viết tắt của 3 tác giả này) và được sử dụng rộng rãi trong công tác mã hoá và công nghệ chữ ký điện tử.

- Chữ ký số được thiết kế dựa trên công nghệ mã khóa công khai (RSA):

    - Tức là người sử dụng sẽ có 1 cặp khóa (gọi là keypair). Trong cặp khóa này có chứa khóa công khai (gọi là public key) và khóa bí mật (private key).

    - Điều này giúp các chủ nhân của chữ ký gia tăng được độ bảo mật, hạn chế những thành phần xấu lợi dụng chiếm đoạt chữ ký cho những mục đích không chính đáng.

- So với chữ ký truyền thống, private key đóng vai trò của chính người gửi, public key của người gửi đóng vai trò là bản sao của chữ ký mà có thể được công khai.

- Khóa bí mật thuộc hệ thống cặp khóa trong chữ ký online. Đây là một loại khóa sử dụng mật mã (password) không đối xứng. Tiếp đó sẽ được các chuyên gia kỹ thuật dùng để tạo ra chữ ký cho người dùng.

- Khóa công khai hay còn gọi là public key có tác dụng để kiểm tra chữ ký online đã được tạo bởi khóa bí mật. Khóa công khai cũng tương tự như loại khóa còn lại trong cặp thuộc hệ thống mật mã khóa không cân xứng.

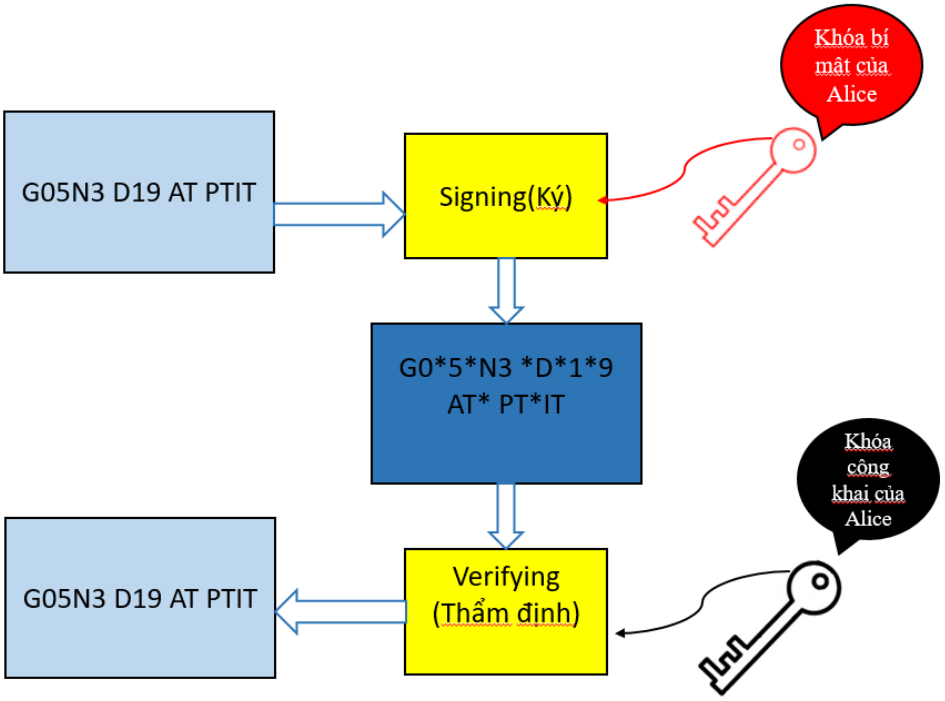
- Toàn bộ quá trình gồm 3 thuật toán:
  - Thuật toán tạo khóa
  - Thuật toán tạo chữ ký số
  - Thuật toán kiểm tra chữ ký số
- Giải thuật tạo chữ ký số bằng RSA còn để tạo chữ ký số cho văn bản sử dụng hàm băm
  - Hàm băm (Hash Function) là hàm toán học chuyển đổi thông điệp (message) có độ dài bất kỳ (hữu hạn) thành một dãy bits có độ dài cố định (tùy thuộc vào thuật toán băm). Dãy bít này được gọi là thông điệp rút gọn.(message digest) hay giá trị băm (hash value), đại diện cho thông điệp ban đầu.
- Trong một số trường hợp đoạn thông tin cần gửi bị phá mã hóa hoặc rơi vào các giá trị không an toàn, RSA trên thực tế thường bao gồm một hình thức chuyển đổi ngẫu nhiên ví dụ như :
  - Sử dụng chuẩn PKCS(Public Key Cryptography Standards)
  - Tiêu chuẩn PKCS còn được bổ sung các tính năng khác để đảm bảo an toàn cho chữ ký RSA (Probabilistic Signature Scheme for RSA - RSA-PSS).

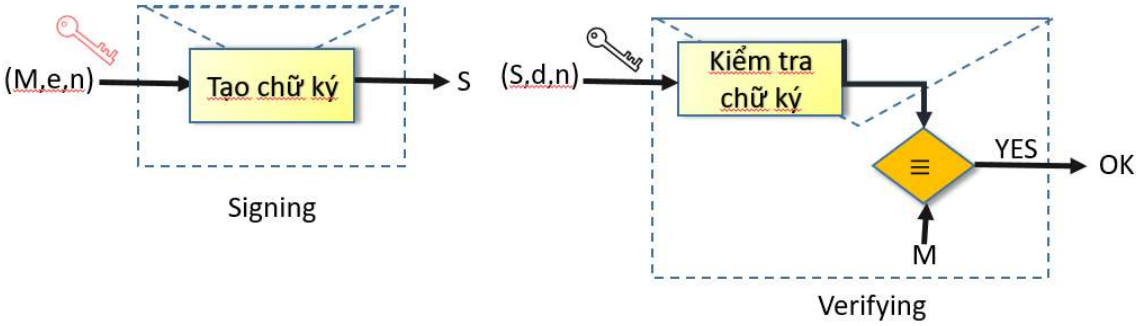
II - Nội Dung

1. Kiến trúc

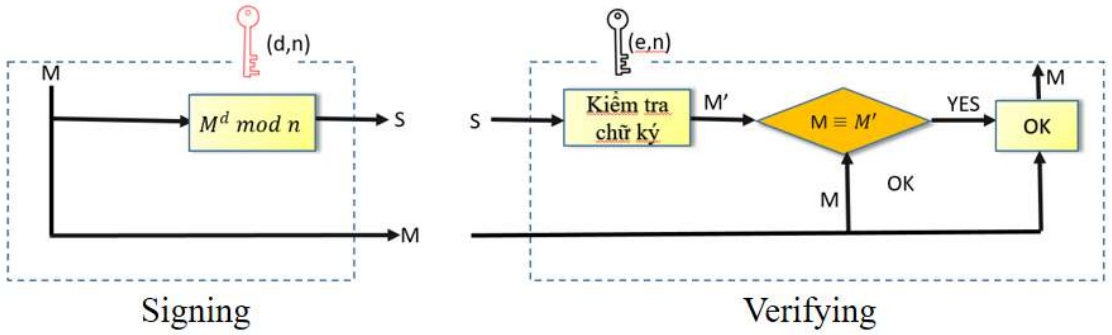
1.1. Cơ chế, ý tưởng

***Ví dụ**: Alice có hai khóa, một khóa công khai và một khóa riêng. Alice đưa khóa công khai của mình cho Bob, nhưng giữ lại khóa riêng cho mình.Khi muốn chuyển tài liệu cho Bob, Alice có thể xác nhận (ký) các tài liệu này dùng chính khóa riêng của mình và gửi chúng đến Bob .Bob sau đó sẽ dùng khóa công khai của Alice, để có thể kiểm tra tài liệu mà cô ấy nhận được, thực sự được gửi bởi Alice*





1.2. Kiến trúc



Kiến trúc tạo và thẩm định chữ ký

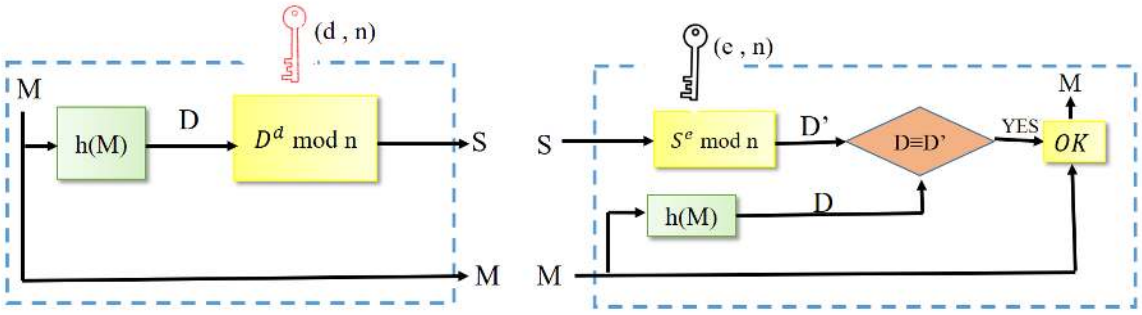
- **Phần Signing(Ký)**

1. Tính toán chuỗi đại diện (message digest/ hash value) của thông điệp sử dụng một giải thuật băm SHA-1

2. Chuỗi đại diện được ký sử dụng khóa riêng (Private key) của người gửi và giải thuật tạo chữ ký RSA. Kết quả thu được là chữ ký số (S) của thông điệp

3. Thông điệp ban đầu (M) được ghép với chữ ký số (S) tạo thành thông điệp đã được ký (S)

4. Thông điệp đã được ký (S + M) được gửi cho người nhận



Kiến trúc tạo và thẩm định chữ ký sử dụng hàm băm

- **Phần Verifying(Thẩm định)**

1. Tách chữ ký số RSA và thông điệp gốc khỏi thông điệp đã ký để xử lý riêng;

2. Tính toán chuỗi đại diện D của thông điệp gốc sử dụng giải thuật băm (là giải thuật sử dụng trong quá trình ký là SHA-1)

3. Sử dụng khóa công khai (Public key) của người gửi để giải mã chữ ký số RSA→ chuỗi đại diện thông điệp D'

4. So sánh D và D':

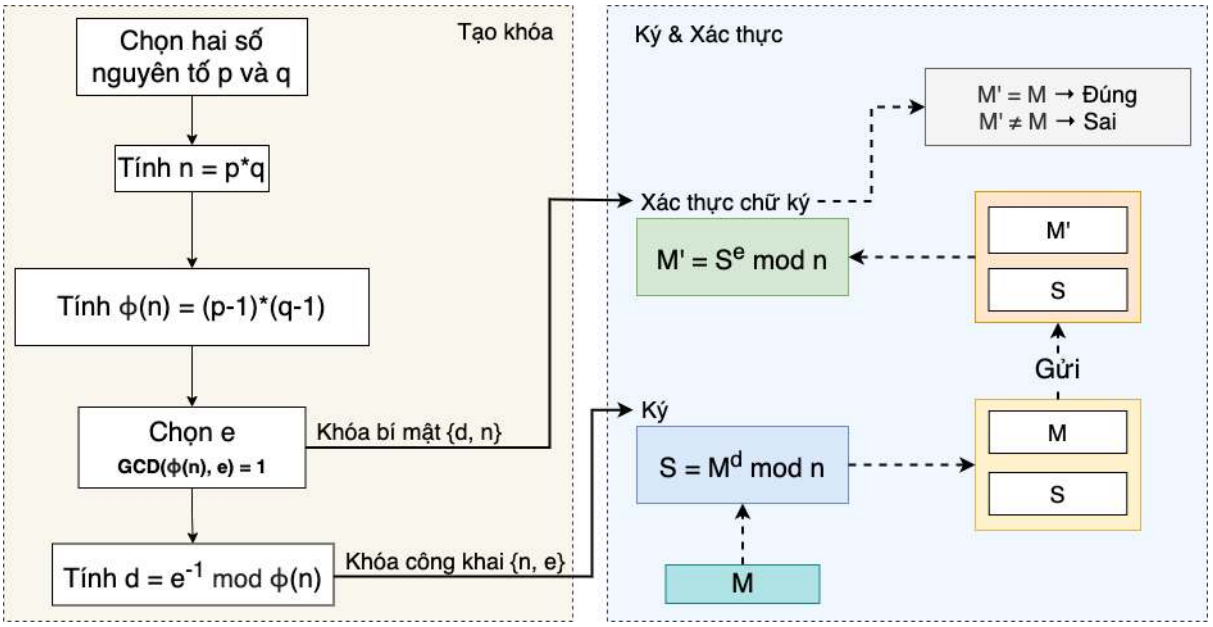
   - Nếu D = D' → chữ ký kiểm tra thành công. Thông điệp đảm bảo tính toàn vẹn và thực sự xuất phát từ người gửi (do khóa công khai được chứng thực).

   - Nếu D ≠ D' → chữ ký không hợp lệ. Thông điệp có thể đã bị sửa đổi hoặc không thực sự xuất phát từ người gửi.

2. Giải thuật

2.1. Giải thuật RSA được dùng trong việc tạo khóa, tạo chữ ký và kiểm tra chữ ký

2.1.1. Sơ đồ giải thuật



2.1.2. Tạo khóa

LÝ THUYẾT	VÍ DỤ
Bước 1: Sinh ngẫu nhiên hai số nguyên tố lớn p và q	***Bước 1***: Chọn số 23 và 41 (2 số này là số nguyên tố)
Bước 2: Tính n = p*q và φ (n) = (p -1) (q -1)	***Bước 2:*** Tính n = p*q và φ (n) = (p -1) (q -1) n=943, φ (n)=880
Bước 3**:** Chọn ngẫu nhiên số e 1 < e < φ (n) sao cho (e, φ (n)) = 1	***Bước 3***: Chọn e = 7 Vì UCLN (7, 880) = 1;

Bước 4: Tính d = e^{-1} mod φ (n) bằng cách sử dụng thuật toán Euclide mở rộng để tính số nguyên d, 1< d < φ (n) sao cho e*d= 1 mod (φ (n))	***Bước 4***: 7d = 1+880x → d=503 và x=4
Bước 5: + ***n*** và ***e*** làm khóa công khai + **d** làm khóa bí mật	***Bước 5:*** + n= 943 và e=7 làm khóa công khai + d=503 làm khóa bí mật.

2.1.3. Tạo chữ ký số

LÝ THUYẾT	VÍ DỤ
Bước 1: Số hóa thông điệp P thành số nguyên M sao cho 1 < M < n-1	***Bước 1***: Thông tin cần gửi M= 35
Bước 2: Tính chữ ký S = M^d mod n.	***Bước 2***: S= 35^{503} mod 943
Bước 3: Gửi S, M cho Bob.	***Bước 4***: S=790

2.1.4. Kiểm tra chữ ký

LÝ THUYẾT	VÍ DỤ

Bước 1: Xác thực đúng khóa công khai của Alice là (n,e) và S,M	***Bước 1***: Bob nhận được khóa công khai n=943 và e =7 và S=790
Bước 2: Kiểm tra: - Tính M'= S^e mod n.	***Bước 2***: Kiểm tra: M'= 790^7 mod 943 → M=35.
Bước 3: Xác nhận chữ ký của A	***Bước 3:*** M=M'=35 → trả về đúng

2.2. Sử dụng hàm băm ánh xạ dữ liệu thành các giá trị có kích thước cố định.

Trong thực tế để tạo chữ ký số bằng RSA thường sử dụng hàm băm của dữ liệu thay vì sử dụng trực tiếp dữ liệu của nó.

Ví dụ: *Alice muốn gửi cho Bob một văn bản có chữ ký của mình.*

Để làm việc này, Alice tạo ra một giá trị băm (hash value) của văn bản cần ký và tính giá trị S= mod n của nó. Giá trị cuối cùng chính là chữ ký điện tử của văn bản đang xét.

Khi Bob nhận được văn bản cùng với chữ ký điện tử, anh ta tính giá trị M'= mod n của chữ ký đồng thời với việc tính giá trị băm của văn bản.

Nếu 2 giá trị này như nhau thì Bob biết rằng người tạo ra chữ ký biết khóa bí mật của Alice và văn bản đã không bị thay đổi sau khi ký.

Việc này mang lại rất nhiều lợi ích như:

- Các hàm hash là hàm 1 chiều, vì vậy dù có được hash cũng không thể biết được bản tin gốc như thế nào.
- Độ dài hash là cố định và thường rất nhỏ, vì vậy chữ số sẽ không chiếm quá nhiều dung lượng.

- Giá trị hash còn có thể dùng để kiểm tra lại bản tin nhận được có nguyên vẹn hay không?

2.3. Cài đặt giải thuật trong ngôn ngữ Python 3

- Python hỗ trợ tính toán với số rất lớn phù hợp với giải thuật RSA (cộng, trừ, nhân, chia, lũy thừa, lũy thừa với modulo, chia lấy phần dư)

- Dễ dàng tạo hàm GCD và Modular Inverse, có thư viện hashlib có sẵn SHA1

- Ngoài ra có cộng đồng lập trình viên viết thư viện hỗ trợ lớn dễ dàng tìm hiểu và tham khảo.

- Cụ thể cài đặt ký và kiểm tra chữ ký RSA trên Python 3:

- Tạo khóa:

```
# ramdom hai số nguyên tố p và q với kích thước bits
# bits mặc định là 1024 bit, cỡ 10^308
self.__p = getPrime(bits)
self.__q = getPrime(bits)
# tính n = p * q
self.n = self.__p * self.__q
# tính phi = (p-1) * (q-1)
self.__phi = (self.__p - 1) * (self.__q - 1)
# giới hạn số bit của e lại để chạy nhanh hơn
e_size = bits // 8 if bits > 64 else ((bits + 8) % 16)
# tìm e phù hợp với phi
while True:
    # random e là số nguyên tố
    self.e = getPrime(e_size)
    # GCD(e, phi) = 1 -> thỏa mãn điều kiện
    if self.__gcd_extended(self.e, self.__phi)[0] == 1:
        break
# tính d = e^(-1) mod(phi)
self.__d = self.__modulo_inverse(self.e, self.__phi)
```

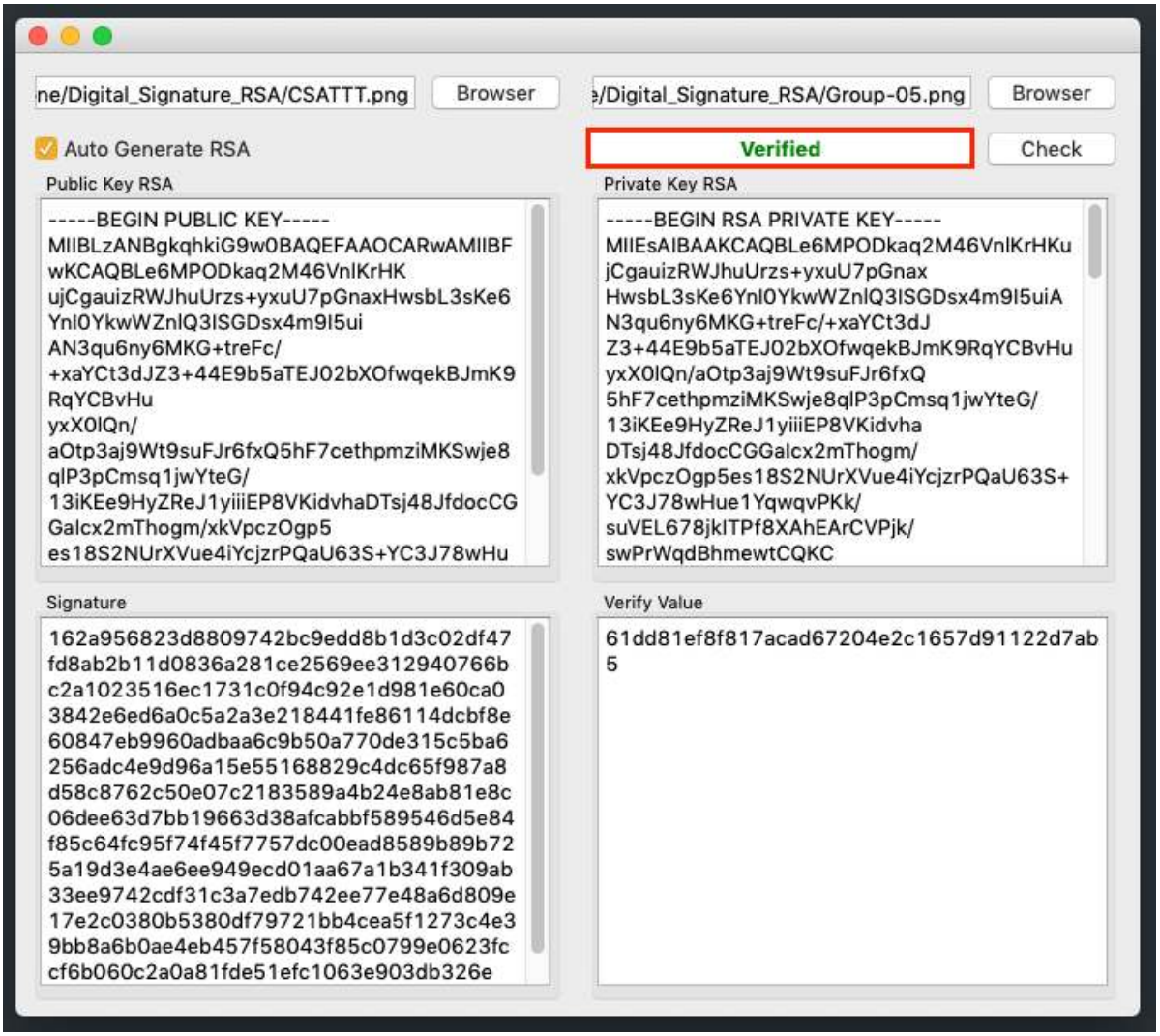
- Hàm GCD và Modular Inverse:

```
# Euclid mở rộng
def __gcd_extended(self, a, b):
    """
    Nếu d = gcd(a, b)
    thì tồn tại cặp số nguyên x, y
    sao cho ax + by = d
    """
    if a == 0:
        return b, 0, 1
    gcd, y, x = RSA.__gcd_extended(self, b % a, a)
    return gcd, x - (b // a) * y, y

# Ngịch đảo modulo
def __modulo_inverse(self, phi, e):
    gcd, x, y = RSA.__gcd_extended(self, phi, e)
    if gcd != 1:
        raise Exception("No modular inverse")
    return x % e
```

- Ký và kiểm tra chữ ký:

```
# Ký
def sign(self, m):
    m = int(self.hash_file(m), 16)
    return pow(m, self.__d, self.n)

# Kiểm tra chữ ký
def verify(self, sign):
    return pow(sign, self.e, self.n)
```

- Demo kiểm tra tính toàn vẹn của dữ liệu



3. Các điểm yếu

3.1. Tốc độ thực hiện của thuật toán RSA

Tốc độ thực hiện của hệ RSA là một trong những điểm yếu so với các hệ mật mã khóa đối xứng.

Theo ước tính, thực hiện tạo khóa RSA chậm hơn 100 lần so với hệ mã khóa đối xứng DES (Khi thực hiện bằng phần mềm). Và chậm hơn 1000 lần so với DES (Khi thực hiện bằng phần cứng).

→ Phương thức này cũng tạo ra những vấn đề an ninh mới. Một ví dụ là cần phải tạo ra khóa đối xứng thật sự ngẫu nhiên. Nếu không, kẻ tấn công sẽ bỏ qua RSA và tập trung vào việc đoán khóa đối xứng.

3.2. Chi phí

- Để thực hiện thuật toán RSA phần lớn tốn chi phí thực hiện các phép tính cơ bản như : Tạo khóa, mã hóa, giải mã.

- Quá trình ký và xác thực chữ ký tương đương với chi phí thực hiện các phép tính lũy thừa modulo n.

- Để đảm bảo cho khóa bí mật được an toàn thì thường chọn mũ công khai e nhỏ hơn nhiều so với số mũ bí mật d

3.3. Hiệu suất của thuật toán RSA

Do tốc độ của RSA không nhanh nên RSA không được dùng để ký khối lượng dữ liệu lớn mà thường dùng với giải thuật băm để tăng tính linh hoạt trong việc ký và xác thực chữ ký

4. Các dạng tấn công

4.1. Dạng tấn công 1: Tìm cách xác định khóa bí mật.

4.1.1. Bị lộ một trong các giá trị: p, q, (n)

Nếu trong quá trình lập khóa mà người sử dụng vô tình để lộ 1 trong 3 nhân tử trên ra ngoài thì kẻ tấn công sẽ dễ dàng tính được khóa bí mật theo công thức:

$$de \equiv 1 \pmod{\phi(n)}$$

Biết được khóa bí mật, kẻ tấn công sẽ giả mạo được chữ ký của người dùng.

⇨ Giải pháp: đảm bảo tính bí mật của p, q, φ (n) trong quá trình lập khóa.

4.1.2. Tấn công dựa theo khóa công khai n và e của người ký.

Lúc này, kẻ tấn công sẽ tìm cách phân tích giá trị n ra hai thừa số nguyên tố p và q. Từ đó, sẽ tính được f(n)=(p-1).(q-1); cuối cùng tính được khóa bí mật d.

⇨ Giải pháp phòng tránh: Nên chọn số nguyên tố p và q đủ lớn để việc phân tích n thành tích của hai thừa số nguyên tố là khó có thể thực hiện được trong thời gian thực. Trong thực tế, người ta thường sinh ra các số lớn (ít nhất 100 chữ số), sau đó kiểm tra tính nguyên tố của nó.

4.1.3. Sử dụng các tham số (p-1) hoặc (q-1) có các ước nguyên tố nhỏ

Nếu ta bất cẩn trong việc chọn các tham số p và q để cho (p-1) hoặc (q-1) có các ước nguyên tố nhỏ thì sơ đồ chữ ký sẽ trở nên mất an toàn. Bởi vì, khi (p-1) hoặc (q-1) có các ước nguyên tố nhỏ thì ta có thể dùng thuật toán (p-1) của Pollar để phân tích giá trị modulo n thành thừa số một cách dễ dàng.

⇨ Giải pháp phòng tránh: Chọn các tham số p và q sao cho (p-1) và (q-1) phải có các ước nguyên tố lớn

4.2. Tấn công dạng 2: Giả mạo chữ ký (không tính trực tiếp khóa bí mật)

Ta có: S^e = m (mod N). Với S là chữ ký hợp lệ của m, vì vậy có thể xây dựng một cặp thông điệp/ chữ ký hợp lệ mà không cần biết khóa riêng.

$$(m_1 \times m_2)^d = m_1^d \times m_2^d \,(mod\ N) = \ S_1 \times S_2 \,(mod\ N)$$

Với hai chữ ký, chúng ta có thể tạo chữ ký thứ ba mà không cần biết khóa riêng.

⇨ Giải pháp phòng tránh: Mã hóa trước, ký sau

- Đầu tiên mã hóa m bằng hàm mã hóa μ(m)
- Sau đó ký σ = $\mu(m)^d$ (mod N)

  Trong đó hàm mã hoá μ(m) được nhắc đến có hai loại:

  - Mã hóa dành cho dạng tấn công này (Ad-hoc encodings)
    - PKCS#1 v1.5, ISO 9796-1, ISO 9796-2.
    - Được thiết kế để ngăn chặn các cuộc tấn công cụ thể, nhưng có thể bộc lộ một số điểm yếu.
  - Mã hóa đảm bảo an toàn (Provably secure encodings)
    - RSA-FDH, RSA-PSS
    - Được chứng minh là an toàn theo các giả định được xác định.

5. Ứng dụng

Sử dụng trong việc đảm bảo vẹn toàn dữ liệu: Chữ ký, công văn, file, tệp tin của người gửi qua môi trường Internet của các cá nhân, cơ quan tổ chức …

Cụ thể như:

- **Trong giao tiếp (E-Mail, SMS):** Khi viết e-mail, ta không cần phải tin tưởng nhà cung cấp email về quyền riêng tư và giả mạo. Ta có thể mã hóa thư bằng khóa công khai của người nhận và ký nó ở bên cạnh. Bằng cách này, người gửi biết chắc chắn rằng không có sự giả mạo và tin nhắn đến từ đúng người gửi. Nó chỉ có thể xảy ra rằng nhà cung cấp không cung cấp thông điệp ở tất cả.

- **Đóng góp mã nguồn:** Rất nhiều mã được viết dưới dạng mã nguồn mở bởi nhiều người. Họ có thể là những người sử dụng mã nguồn đó hoặc họ có thể được trả tiền cho những đóng góp đó. Những người duy trì các dự án cần đảm bảo rằng tất cả các khoản đóng góp đều hữu ích, nhưng họ có thể thiếu thời gian để kiểm tra từng khoản đóng góp. Họ cần phải có khả năng tin tưởng một số người. Thông thường những người đầu tiên được kiểm tra kỹ lưỡng, nhưng theo thời gian, thậm chí ai cũng thể trở thành người đóng góp. Mọi người tin tưởng lẫn nhau. Nhưng họ cần phải chắc chắn rằng đó là cùng một người. Họ cần phải chắc chắn rằng đóng góp của mỗi người không thay đổi. Vì lý do này, cần ký mọi đóng góp.

- **Cập nhật phần mềm:** Hãy nghĩ về TV thông minh / Alexa / FritzBox. Tất cả các thiết bị này đều cần cập nhật. Giả sử rằng ta có thể cắm USB với tệp cập nhật trong thiết bị. Là nhà sản xuất, họ muốn đảm bảo rằng bản cập nhật không bị thay đổi. Họ muốn đảm bảo rằng thiết bị sẽ tiếp tục hoạt động. Vì vậy, họ chia sẻ khóa công khai của công ty trong thiết bị. Khi thiết bị tìm thấy bản cập nhật, nó xác minh rằng công ty là nguồn gốc bằng cách kiểm tra chữ ký của bản cập nhật.

- **Văn bằng kỹ thuật số:** Khi nộp đơn xin việc, nhà tuyển dụng mới tiềm năng của ứng viên có thể muốn xem thư tham khảo và bằng tốt nghiệp của ứng viên. Đặc biệt là khi virus corona đang hoành hành trên khắp thế giới, những tài liệu đó được giao bằng kỹ thuật số. Làm thế nào để người sử dụng lao động kiểm tra xem bằng tốt nghiệp có thực sự hợp lệ không? Chữ ký số có thể giúp ích. Ứng viên sẽ cần phải có một phiên bản có chữ ký kỹ thuật số của văn bằng của ứng viên và có thể chia sẻ khóa công khai theo cách đáng tin cậy với chủ lao động của ứng viên.

- **Tiền điện tử:** Để chứng minh rằng một người là người nắm giữ Bitcoin, hệ thống sử dụng mật mã bất đối xứng. Ngay từ đầu, ai đó được đảm bảo là chủ sở hữu hợp lệ của đồng xu. Sau đó, chủ sở hữu hợp lệ được xác định là chủ sở hữu của khóa riêng, phù hợp với khóa công khai nhất định. Xin lưu ý rằng chữ ký số chỉ chứng minh quyền sở hữu tại một thời điểm cụ thể. Họ không giải quyết được vấn đề mà chủ sở hữu có thể chỉ chi tiêu đồng xu hai lần - vấn đề chi tiêu gấp đôi.

III - Kết Luận

Bài báo cáo đã giới thiệu được kiến trúc, cài đặt giải thuật, các điểm yếu, các dạng tấn công của chữ ký số sử dụng giải thuật RSA

- Tạo và kiểm tra chữ ký số.
- Cài đặt thử nghiệm và kiểm tra chữ ký số để đảm bảo tính toàn vẹn dữ liệu.

IV - Tài Liệu Tham Khảo

1. Luận văn Thạc sĩ: Các phương pháp tấn công chữ ký số: RSA, ELGAMAL, DSS (tác giả Lê Công Tuấn Anh)
2. https://www.cs.cornell.edu/courses/cs5430/2015sp/notes/rsa_sign_vs_dec.php
3. https://geeklaunch.net/blog/what-does-my-rsa-public-key-actually-mean/
4. http://www2.lawrence.edu/fast/GREGGJ/CMSC510/Ch31/RSA.html
5. https://slideplayer.com/slide/5974446/
6. https://en.wikipedia.org/wiki/RSA_(cryptosystem)
7. https://cp-algorithms.com/algebra/module-inverse.html
8. https://levelup.gitconnected.com/5-applications-of-digital-signatures-4e785d22d439
9. http://www.crypto-uni.lu/jscoron/cours/mscrypto/cc3c.pdf