**BÀI TẬP NHÓM 4**

**Thành Viên :**

**Hệ mã hóa RSA**

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. Trong hệ mã hóa này, public key có thể chia sẻ công khai cho tất cả mọi người. Hoạt động của RSA dựa trên 4 bước chính: sinh khóa, chia sẻ key, mã hóa và giải mã.

**Sinh khóa**

Mấu chốt cơ bản của việc sinh khóa trong RSA là tìm được bộ 3 số tự nhiên e, d và n sao cho:

và một điểm không thể bỏ qua là cần bảo mật cho d sao cho dù biết e, n hay thậm chí cả m cũng không thể tìm ra d được.

Cụ thể, khóa của RSA được sinh như sau:

* Chọn 2 số nguyên tố p và q
* Tính n = pq. Sau này, n sẽ được dùng làm modulus trong cả public key và private key.
* Tính một số giả nguyên tố bằng phi hàm carmichael như sau: λ(n) = BCNN(λ(p), λ(q)) = BCNN(p − 1, q − 1). Giá trị này sẽ được giữ bí mật.
* Chọn một số tự nhiên e trong khoảng (1, λ(n)) sao cho ƯCLN(e, λ(n)) = 1, tức là e và λ(n) nguyên tố cùng nhau.
* Tính toán số d sao cho d ≡ 1/e (mod λ(n)) hay viết dễ hiểu hơn thì de ≡ 1 (mod λ(n)). Số d được gọi là số nghịch đảo modulo của e (theo modulo mod λ(n)).

Public key sẽ là bộ số (n, e), và private key sẽ là bộ số (n, d). Chúng ta cần giữ private key thật cẩn thận cũng như các số nguyên tố p và q vì từ đó có thể tính toán các khóa rất dễ dàng.

Trong thực hành, chúng ta thường chọn e tương đối nhỏ để việc mã hóa và giải mã nhanh chóng hơn. Giá trị thường được sử dụng là e = 65537. Ngoài ra, chúng ta có thể tính số giả nguyên tố bằng [phi hàm Euler](http://math-it.org/Mathematik/Zahlentheorie/Euler.html) φ(n) = (p − 1)(q − 1) và dùng nó như λ(n). Vì φ(n) là bội của λ(n) nên các số d thỏa mãn de ≡ 1 (mod φ(n)) cũng sẽ thỏa mãn d ≡ 1/e (mod λ(n)). Tuy nhiên, một số tác dụng phụ của việc này là d thường sẽ trở nên lớn hơn mức cần thiết.

**Mã hóa và giải mã**

Trong phần này, chúng ta sẽ tìm hiểu cách mã hóa với public key (n, e) và giải mã với private key (n, d).

Nếu chúng ta có bản rõ M, chúng ta cần chuyển nó thành một số tự nhiên m trong khoảng (0, n) sao cho m, n nguyên tố cùng nhau. Việc này rất dễ dàng thực hiện bằng cách thêm một các kỹ thuật padding. Tiếp theo, chúng ta sẽ má hóa m, thành c như sau:

Sau đó giá trị c sẽ được chuyển cho người nhận.

Ở phía người nhận, họ sẽ giải mã từ c để lấy được m như sau:

Từ m có thể lấy lại được bản tin bằng cách đảo ngược padding

Chúng ta lấy một ví dụ đơn giản như sau:

p = 5, q = 7

=> n = pq = 35

=> φ(n) = 24

Chúng ta chọn e = 5 vì ƯCLN(5, 24) = 1, cuối cùng chọn d = 29 vì ed - 1 = 29x5 - 1 chia hết cho 24.

Giả sử m = 32 (dấu cách), chúng ta sẽ mã hóa m và thu được

c = 32 ^ 5 % 35 = 2

Giải mã c để thu được m

m = 2 ^ 29 % 35 = 32

Đây chính là m ban đầu. Bạn có thể thứ m với các giá trị khác nhau để thấy thuật toán hoàn toàn chính xác. Tất nhiên rồi, nó đã được chứng minh mà. Việc chứng minh RSA hoạt động chính xác là một vấn đề toán học khá phức tạp mà bài viết này sẽ không đi vào chi tiết của nó.

Mức độ bảo mật của RSA phụ thuộc rất lớn vào khả năng phân tích thừa số nguyên tố của các số lớn. Bởi vì chúng ta cung cấp public một cách rộng rãi, nếu việc phân tích thừa số nguyên tố đơn giản, thì việc bị lộ private là không thể tránh khỏi.

Vì vậy, khi sinh khóa, chúng ta cần chọn các số nguyên tố p và q một cách ngẫu nhiên. Bản thân hai số nguyên tố này cũng rất lớn, và để việc phân tích thừa số nguyên tố khó khăn hơn, hai số nguyên tố này sẽ không có cùng độ dài. Trong tương lai gần, có lẽ vẫn chưa có một phương pháp hiệu quả nào cho phép thực hiện điều này với các máy tính cá nhân.

Tuy nhiên, với sự phát triển của công nghệ, các siêu máy tính xuất hiện ngày càng nhiều. Cùng với chúng ta máy tính lượng tử cho phép tính toán với tốc độ cao hơn rất nhiều có thể sẽ phá vỡ sự bảo mật của RSA.

Ngày từ năm 1993, [thuật toán Shor](http://tph.tuwien.ac.at/~oemer/doc/quprog/node18.html) đã được phát triển và chỉ ra rằng [máy tính lượng tử](http://computer.howstuffworks.com/quantum-computer.htm) có thể giải bài toán phân tích ra thừa số trong thời gian đa thức. Rất may là những điều này mới chỉ là lý thuyết vì đến thời điểm hiện tại và trong vài năm tới, máy tính lượng tử vẫn chưa hoàn thiện.

Một câu hỏi khá thú vị là có thể đảo vai trò của public key và private key hay không? Tức là chúng ta dùng private key để mã hóa và dùng public key để giải mã hay không? Câu trả lời là có. Tuy nhiên, không ai làm như vậy cả vì rất đơn giản: từ public key rất khó để suy ra private key nhưng từ private để suy ra public thì rất đơn giản. Nếu cho người khác private key (để họ mã hóa), những kẻ tấn công man in the middle sẽ dễ dàng tính toán được public và thay đổi hoàn toàn cuộc hội thoại của chúng ta. Như thế thì nguy hiển quá.

Tuy nhiên, có một trường hợp chúng ta đã đảo vai trò của private key và public key như vậy. Đó chính là khi chúng ta sử dụng chữ ký số.

**Chữ ký số sử dụng hệ mã hóa RSA**

Việc ký tên và xác thực chữ ký số sử dụng hệ mã hóa RSA tương tự như quá trình mã hóa mà giải mã ở trên. Tuy nhiên vai trò của public key và private thì có thay đổi đôi chút.

Để tạo chữ ký, người gửi sẽ dùng private key và người nhận sẽ dùng public key để xác thực chữ ký đó.

Tuy nhiên, vì bản tin rất dài nên việc mã hóa toàn bộ bản tin sẽ rất mất thời gian. Vì vậy, trong thực hành, chữ ký số thường sử dụng phương pháp mã hóa giá trị hash của bản tin. 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?

Chữ ký số đem lại nhiều giá trị hơn chữ ký tay rất nhiều. Có lẽ cũng vì vậy, việc xử lý chữ ký số phức tạp hơn hẳn chữ ký tay truyền thống.

**Xác định nguồn gốc**

Hệ mã hóa bất đối xứng cho phép tạo chữ ký với private key mà chỉ người chủ mới biết. Khi nhận gói tin, người nhận xác thực chữ ký bằng cách dùng public key giải mã, sau đó tính giá trị hash của bản tin gốc và so sánh với hash trong gói tin nhận được. Hai chuỗi này phải trùng khớp với nhau. Tất nhiêu hệ mã hóa RSA vẫn có những thách thức về an ninh nhất định nhưng dù sao thì nó vẫn khá an toàn.

**Dữ liệu được giữ một cách toàn vẹn**

Tin nhắn gửi từ chủ private key rất khó có thể bị giả mạo. Bởi vì nếu thay đổi tin nhắn thì giá trị hash cũng phải thay đổi theo. Những kẻ nghe lén trong mạng đương nhiên là có thể tìm cách đọc tin nhắn gốc và cả hash của nó. Nhưng hắn ta không thể thay đổi tin nhắn được vì hắn không có private key để sửa đổi chữ ký số cho phù hợp.

**Chữ ký số không thể phủ nhận**

Trong giao dịch, một gói tin kèm chữ ký số rất dễ dàng tìm ra được nguồn gốc của chữ ký đó. Bởi vì private key là bí mật và chỉ người chủ của nó mới có thể biết, họ không thể chối cãi rằng chữ ký này không phải do họ phát hành. Cũng tương tự trường hợp trên, hệ mã hóa RSA hay bất kỳ hệ mã hóa nào khác cũng đều có những vấn đề về an ninh nên việc này không thể đảm bảo tuyệt đối chính xác được.

**Kết luận**

Công nghệ ngày càng phát triển, các giao dịch điện tử, đặc biệt là thương mại điện tử đang được thực hiện liên tục hàng ngày, hàng giờ. Chữ ký số là một công nghệ cho phép xác định chủ nhân của dữ liệu nào đó và kiểm tra dữ liệu có nguyên vẹn hay không.

Ở nước ta, chữ ký số áp được áp dụng trong một số hoạt động ngành ngân hàng và kế toán. Nhưng trong thời gian tới, rất có thể chữ ký số sẽ phổ biến hơn, khi chúng ta đang dần dần hướng đến một thế giới không còn giấy bút.