

HỆ MÃ HÓA MERKLE - HELLMAN (KNAPSACK)

Tóm tắt

Báo cáo này trình bày về một hệ mã hoá được gọi là hệ mã khoá công khai được ra đời nhằm khắc phục những nhược điểm của hệ mã khoá đối xứng mà cụ thể là thuật toán mã hoá do Merkle và Hellman đề xuất dựa trên bài toán ĐÓNG THÙNG (hay còn gọi là bài toán BA LÔ)

Abstract

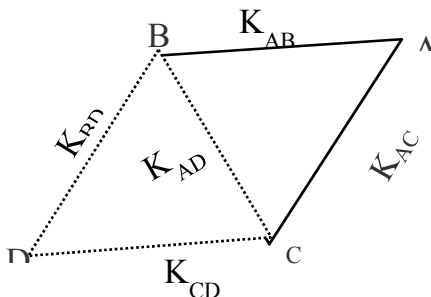
This report presents an encryption system called public key cryptosystems are created to overcome the disadvantages of Symmetric Key Cryptosystems that is specific encryption algorithms proposed by Merkle and Hellman based on the PACKING (also known as the problem of the backpack)

1. Giới thiệu

Như đã nêu, các hệ thống mật mã đã giới thiệu cho đến giờ đều được gọi là các hệ mật mã khóa đối xứng (Symmetric Key Cryptosystems) do vai trò hai bên gửi và nhận tin đều như nhau vì đều sở hữu chung một khoá bí mật. Cũng có nhiều cách gọi khác đối với các hệ mật mã này, sử dụng tùy vào các ngữ cảnh phù hợp:

- Hệ mã với khóa sở hữu riêng (Private Key Cryptosystems)
- Hệ mã với khóa bí mật (Secret Key Cryptosystems)
- Hệ mã truyền thống (Conventional Cryptosystems)

Chúng ta sẽ sử dụng ký hiệu viết tắt cho hệ mật mã đối xứng là SKC.



Tuy nhiên các hệ mã đối xứng có những nhược điểm cơ bản như sau:

- Vấn đề quản lý khoá (tạo, lưu mật, trao chuyển ...) là rất phức tạp khi sử dụng trong môi trường trao đổi tin giữa rất nhiều người dùng. Với số lượng NSD là n thì số lượng khoá cần tạo lập là $n(n-1)/2$. Mỗi người dùng phải tạo và lưu $n-1$ khoá bí mật để làm việc với $n-1$ người khác trên mạng. Như vậy rất khó khăn và không an toàn khi n tăng lớn.
- Thứ hai là, trên cơ sở mã đối xứng, ta không thể thiết lập được khái niệm chữ ký điện tử (mà thể hiện được các chức năng của chữ ký tay trong thực tế) và cũng do đó không có dịch vụ non-repudiation¹ (không thể phủ nhận được) cho các giao dịch thương mại trên mạng.

Vấn đề là ở chỗ trong hệ SKC, thông tin mật được chia sẻ chung bởi cả hai bên Alice và Bob, do đó Alice có thể làm được bất kỳ cái gì mà Bob làm và ngược lại. Giải pháp duy nhất cho vấn đề này là phải có thêm một thành phần thứ ba trong bất cứ giao dịch nào giữa Alice và Bob, tức là một

người có thẩm quyền (trusted authority) mà cả Alice và Bob đều tin tưởng là trung thực. Người này sẽ làm chứng và trọng tài trong trường hợp xảy ra tranh cãi giữa hai bên trung thực. Người này sẽ làm chứng và trọng tài trong trường hợp xảy ra tranh cãi giữa hai bên Alice và Bob. Tuy nhiên công việc của người trọng tài này sẽ rất nặng vì phải tham gia vào tất cả các giao dịch của các bên, và sớm muộn cũng sẽ trở thành điểm quá tải về giao thông truyền tin cũng như tốc độ xử lý -- điểm tắc nghẽn cổ chai (bottleneck).

Sớm nhận thức những vấn đề đó, Diffie & Hellman trong công trình nổi tiếng của mình (1976) đã đề xuất những tư tưởng về một loại hệ mã với nguyên tắc mới, xây dựng xoay quanh một NSD – chủ nhân hệ thống – chứ không phải là xoay quanh một cặp NSD như trong bài toán kênh truyền tin mật truyền thống.

Trong hệ thống mới này, mỗi NSD có hai khoá, một được gọi là khoá bí mật (secret key hay private key) và một được gọi là khoá công khai (public key). Khoá thứ nhất chỉ mình user biết và giữ bí mật, còn khoá thứ hai thì anh ta có thể tự do phổ biến công khai. Khoá thứ nhất thường đi liền với thuật toán giải mã, còn khoá thứ hai thường đi liền với thuật toán sinh mã, tuy nhiên điều đó không phải là bắt buộc. Ta hãy ký hiệu chúng là z (khóa riêng) và Z (khóa công khai)

Hoạt động của chúng là đối xứng:

$$X = D(z, E(Z, X)) \quad (1)$$

$$\text{Và } X = E(Z, D(z, X)) \quad (2)$$

Trong đó hệ thức (1) biểu tượng cho bài toán truyền tin mật: bất kỳ NSD nào khác như $B, C, D \dots$ muốn gửi tin cho A chỉ việc mã hoá thông tin với khoá công khai (Z_A) của A rồi gửi đi. Chỉ có A mới có thể khoá riêng để giải mã (z_A) và đọc được tin; kẻ nghe trộm Eve không thể giải mã để lấy được tin vì không có khoá z_A .

Còn hệ thức (2) sẽ được sử dụng để xây dựng các hệ chữ ký điện tử như sau này ta sẽ nghiên cứu, trong đó thao tác Ký chính là thực hiện $E(Z_A)$ còn kiểm định chữ ký là thông qua gọi $D(z_A)$.

Hệ mật mã theo nguyên tắc nói trên được gọi là hệ mã với khoá công khai (public key cryptosystems) hay còn được gọi là mã khóa phi đối xứng (asymmetric key cryptosystems). Ta sẽ viết tắt hệ thống kiểu này bằng PKC.

Nguyên tắc cấu tạo một hệ PKC sử dụng cửa bẫy (trapdoor)

Một hệ mã PKC có thể được tạo dựng trên cơ sở sử dụng một hàm một chiều (one-way). Một hàm f được gọi là một chiều nếu:

1. Đối với mọi X tính ra $Y = f(X)$ là dễ dàng.
 2. Khi biết Y rất khó để tính ngược ra X .
- Ví dụ 3.1.* Cho n số nguyên tố $p_1, p_2, \dots, p_n$ ta có thể dễ dàng tính được $N = p_1 * p_2 * \dots * p_n$, tuy nhiên khi biết N , việc tìm các thừa số nguyên tố của nó là khó khăn hơn rất nhiều, đặc biệt là khi N lớn và các thừa số nguyên tố của nó cũng lớn.

Tuy nhiên, chúng ta cần một hàm một chiều đặc biệt có trạng bị một cửa bẫy (trap door) sao cho nếu biết sử dụng nó thì việc tìm nghịch đảo của f là dễ dàng, còn nếu không (không biết bí mật cửa bẫy) thì vẫn khó như thường.

Một hàm một chiều có cửa bẫy như thế có thể dùng để tạo ra một hệ mã PKC như sau. Lấy EZ (hàm sinh mã) là hàm một chiều có cửa bẫy này. Như vậy bí mật cửa bẫy chính là khóa bí mật z , mà nếu biết nó thì có thể dễ dàng tính được cái nghịch đảo của EZ tức là biết Dz , còn nếu không biết thì rất khó (chỉ còn cách thử vét cạn, thực tế sẽ là bất khả thi vì khối lượng tính toán quá lớn).

Sau đây chúng ta sẽ khảo sát hai ví dụ về việc xây dựng hàm một chiều có cửa bẫy.

Ví dụ đầu tiên là một cố gắng nhưng thất bại, hệ Trapdoor Knapsack. Ví dụ thứ hai là một hệ đã thành công và rất nổi tiếng, đó là hệ RSA.

2. Merkle-Hellman Trapdoor Knapsack (Cửa bẫy dựa trên bài toán đóng thùng)

Vào năm 1978, hai ông Merkle và Hellman đã đề xuất một thuật toán mã hoá theo mô hình PKC dựa trên bài toán ĐÓNG THÙNG (hay còn gọi là bài toán “cái túi”, hay “ba lô”) như sau:

Cho 1 tập hợp các số dương a_i , $1 \leq i \leq n$ và một số T dương. Hãy tìm một tập hợp chỉ số $S \subset \{1, 2, \dots, n\}$

sao cho: $\sum_{i \in S} a_i = T$

Bài toán này là một bài toán khó (NP-khó), theo nghĩa là chưa tìm được thuật toán nào tốt hơn là thuật toán thử-vét cạn và như vậy thời gian xử lý sẽ là

hàm mũ (trong khi bài toán được quan niệm là dễ theo nghĩa tin học nếu có thuật toán thời gian đa thức).

Ví dụ 3.2

$$(a_1, a_2, a_3, a_4) = (2, 3, 5, 7)$$

$$T = 7.$$

Như vậy ta có 2 đáp số $S = (1, 3)$ và $S = (4)$.

Từ bài toán *Đóng thùng* này chúng ta sẽ khảo sát các khả năng vận dụng để tạo ra thuật

toán mã khối PKC. Sơ đồ đầu tiên như sau:

Chọn một vector $a = (a_1, a_2, \dots, a_n)$ - được gọi là vector mang (cargo vector) Với một khối tin $X = (X_1, X_2, X_3, \dots, X_n)$, ta thực hiện phép mã hoá như sau:

$$T = \sum_{i=1, n} a_i X_i (*)$$

Việc giải mã là: Cho mã T , vector mang a , tìm các X_i sao cho thoả mãn (*).

Sơ đồ này đã thể hiện một hàm một chiều mà dùng làm sinh mã thì tính toán dễ dàng nhưng việc giải mã, tức tính hàm ngược của nó, là rất khó. Bây giờ ta sẽ tiếp tục tìm cách đưa vào một cửa bẫy (trapdoor) để việc giải mã có thể làm được dễ dàng (nếu biết cửa bẫy bí mật).

Ví dụ 3.3

Vector mang siêu tăng: $a = (1, 2, 4, 8)$

Cho $T = 14$, ta sẽ thấy việc tìm

$X = (X_1, X_2, X_3, X_4)$ sao cho $T = \sum a_i X_i$ là dễ dàng: Đặt $T = T_0$

$$X_4 = 1 \quad T_1 = T_0 - X_4 = 6 \rightarrow (X_1 X_2 X_3 1)$$

$$X_3 = 1 \quad T_2 = T_1 - X_3 = 2 \rightarrow (X_1 X_2 1 1)$$

$$X_2 = 1 \quad T_3 = T_2 - 2 = 0 \rightarrow (X_1 1 1 1)$$

$$X_1 = 0 \rightarrow (0 1 1 1)$$

Merkle áp dụng một mẹo dựa trên sử dụng vector mang đặc biệt là vector siêu tăng (super- increasing) như sau. Một vector là siêu tăng nếu thành phần $i+1$ là lớn hơn tổng giá trị của các thành phần đứng trước nó ($1 \leq i$). Khi sử dụng một vector siêu tăng làm vector mang thì sẽ thấy việc tính ngược, tức là giải bài toán đóng thùng là dễ dàng nhờ một giải thuật thăm ăn đơn giản. Điều này được minh họa qua ví dụ bằng số sau.

Ở bước i , tổng đích là T_i (tức là phải tìm các a_j để tổng bằng T_i). Ta đem so sánh T_i với thành phần lớn nhất trong phần còn lại của vector, nếu lớn hơn thì thành phần này được chọn tức là X_i tương ứng bằng 1, còn ngược lại thì X_i tương ứng bằng 0. Sau đó tiếp tục chuyển sang bước sau với $T_{i+1} = T_i - X_i$.

Mặc dù ta đã thấy sử dụng vector siêu tăng là vector mang cho phép giải mã dễ dàng nhưng, tất nhiên, ta còn phải làm thế nào để cho chỉ có người chủ mới biết được và sử dụng nó còn kẻ thù thì không. Tóm lại, cần tạo ra một bí mật của bấy thông qua việc người chủ phải chủ động “ngụy trang” vector siêu tăng để chỉ có anh ta mới biết còn người ngoài không thể lần ra được.

Sơ đồ sau đây sẽ trình bày một cơ chế ngụy trang như vậy. Vector a' là một vector siêu tăng bí mật, sẽ được “ngụy trang”, tức là biến đổi thông qua một hàm g được chọn sẵn để tạo thành vector a không hề có tính siêu tăng

(thậm chí là có thể giảm); vector a này sẽ được sử dụng làm vector mang. Trong quá trình giải mã, người chủ (Alice) sẽ thực hiện

một biến đổi vào dữ liệu, trên cơ sở áp dụng hàm ngược g^{-1} , chuyển việc giải mã thành giải

một bài toán đóng thùng với vector siêu tăng là vector mang. Phép biến đổi g được chọn chính là phép nhân đồng dư với một giá trị khóa bí mật.

Tạo khoá:

1. Alice chọn một vector siêu tăng:

$$a' = (a_1', a_2', \dots, a_n')$$

a' được giữ bí mật tức là một thành phần của khóa bí mật

2. Sau đó chọn một số nguyên $m > \sum a_i'$, gọi là mô-đun đồng dư và một số nguyên ngẫu nhiên ω , gọi là nhân tử, sao cho nguyên tố cùng nhau với m .

Khoá công khai của Alice sẽ là vector a là tích của a' với nhân tử ω :

$$a = (a_1, a_2, \dots, a_n)$$

$$a_i = \omega \times a_i' \pmod{m}; \quad i=1, 2, 3, \dots, n$$

Còn khóa bí mật sẽ là bộ ba (a', m, ω)

Sinh mã:

Khi Bob muốn gửi một thông báo X cho Alice, anh ta tính mã theo công thức:

$$T = \sum a_i X_i$$

Giải mã:

Alice nhận được T , giải mã như sau:

1. Để bỏ lớp ngụy trang cô ta trước hết tính ω^{-1} (là giá trị nghịch đảo của ω , tức là $\omega \times \omega^{-1} = 1 \pmod{m}$, sẽ giới thiệu thuật toán tính sau), rồi tính $T' = T \times \omega^{-1} \pmod{m}$
2. Alice biết rằng $T' = a' \cdot X$ nên cô ta có thể dễ dàng giải ra được X theo siêu tăng a' .

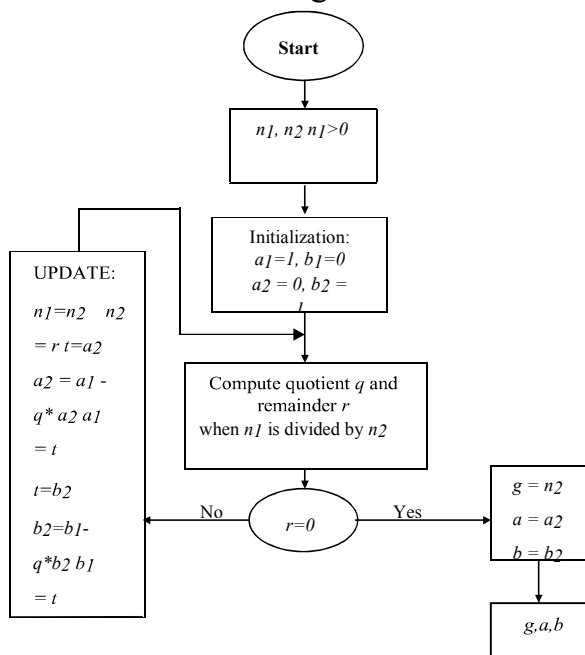
Chú thích: ở đây ta có

$$T' = T \times \omega^{-1} = \sum a_i X_i \omega^{-1} = \sum a_i' \omega X_i \omega^{-1} \\ = \sum (a_i' \omega \omega^{-1}) X_i \omega^{-1} = \sum a_i' X_i = a' \cdot X$$

Như vậy chúng ta đã xem xét xong sơ đồ cụ thể của Merkle-Hellman về một hệ PKC dựa trên bài toán đóng thùng

Tấn công vũ lực (Brute Force Attack)

Ban đầu tấn công vũ lực được xem là



cách duy nhất để phá hệ thống mật mã này.

Với những kẻ không biết trapdoor (a', m, ω) , phá giải mã đòi hỏi phải tìm kiếm vét cạn qua 2^n khả năng của X. Vì vậy với n được chọn đủ lớn tấn công vũ lực là bất khả thi về khối lượng tính toán. Tuy nhiên tấn công vũ lực không phải là cách duy nhất..

Sự đổ vỡ của giải pháp dùng Knapsack (1982-1984).

Shamir-Adleman đã chỉ ra chỗ yếu của giải pháp này bằng cách đi tìm 1 cặp (ω', m') sao cho nó có thể biến đổi ngược a về a' (tính được khóa bí mật - Private key – từ khóa công khai). Năm

1984, Brickell tuyên bố sự đổ vỡ của hệ thống Knapsack với dung lượng tính toán khoảng 1 giờ máy Cray -1, với 40 vòng lặp chính và cỡ 100 trọng số.

Thuật toán tìm giá trị nghịch đảo theo modul đồng dư

Việc xây dựng Knapsack với cửa bẫy đòi hỏi phải tính giá trị nghịch đảo của $\square$ theo modul

m. Thuật toán tìm $x = \omega^{-1} \bmod m$, sao cho $x \cdot \omega = 1 \pmod m$ được gọi là thuật toán GCD mở rộng hay Euclide mở rộng (GCD - Greatest common divisor - ước số chung lớn nhất). Sở dĩ như vậy là vì trong khi đi tìm ước số chung lớn nhất của hai số nguyên n_1 và n_2 , người ta sẽ tính luôn các giá trị a, b sao cho $\text{GCD}(n_1, n_2) = a \cdot n_1 + b \cdot n_2$.

Từ đó suy ra nếu ta đã biết $(n_1, n_2)=1$ thì thuật toán này sẽ cho ta tìm được a, b thoả mãn $a \cdot n_1 + b \cdot n_2 = 1$, tức là n_1 chính là nghịch đảo của a theo modulo n_2 (tức là m)

Sau đây là sơ đồ thuật toán và một ví dụ áp dụng bằng số:

Ví dụ 3.4. Tìm nghịch đảo của 39 theo modulo 11

Đặt $n_1=39, n_2=11$ ta có bảng tính minh họa các bước như sau:

n_1	n_2	r	q	a_1	b_1	a_2	b_2
39	11	6	3	1	0	0	1
11	6	5	1	0	1	1	-3
6	5	1	1	1	-3	-1	4
5	1			-1	4	2	-7

Để thấy $a=a_2=2$ chính là nghịch đảo của 39 theo modulo 11

Kể từ năm 1976, nhiều giải pháp cho PKC đã được nêu ra nhưng khá nhiều trong số đó đã bị phá vỡ hoặc bị chê là không thực dụng do dung lượng tính toán lớn hoặc thông tin nở ra quá lớn khi mã hoá.

Một hệ thống PKC có thể sử dụng vào 2 mục đích cơ bản: (1) Bảo mật thông tin và truyền tin (2) Chứng thực và chữ ký điện tử. Hai thuật toán đáp ứng các ứng dụng trên thành công nhất là RSA và Elgamal. Nói chung thuật toán PKC là chậm và không thích hợp cho mật mã trên dòng (online) với truyền tin tốc độ cao, vì vậy chỉ thường được sử dụng khi cần đến tính an toàn cao và chấp nhận tốc độ chậm. Ngoài ra người ta thường sử dụng kết hợp PKC và SKC (symmetric key cryptosystems) với PKC có tác dụng “khởi động môi” cho SKC: dùng PKC để thiết lập thuật toán tạo ra khoá bí mật thống nhất chung giữa hai bên truyền tin sau đó sử dụng khoá bí mật trên cho pha truyền tin chính bằng SKC sau đó.

Lời cảm ơn

Báo cáo này được sự khuyến khích và hướng dẫn của TS. Lê Nhật Duy, khoa Công nghệ Thông tin trường Đại học Công nghiệp Tp. Hồ Chí Minh.

Tài liệu tham khảo

- [1] Nguyễn Khanh Văn & Trần Khánh Đức: Giáo trình An toàn Thông tin. Đại học Bách Khoa Hà Nội, 2012
- [2] Trần Văn Minh: Bài giảng Bảo mật và An toàn Thông tin. Đại học Nha Trang, 2008.
- [3] Phan Đình Diệu: Lý thuyết mật mã & An toàn Thông tin. NXB ĐHQG Hà Nội, 2002.
- [4] TS. Thái Thanh Tùng: Giáo trình Mật mã học & AN toàn Thông tin, NHÀ XUẤT BẢN THÔNG TIN VÀ TRUYỀN THÔNG.