MỘT PHƯƠNG PHÁP XÂY DỰNG HỆ MẬT POHLIG-HELLMAN TRÊN VÀNH ĐA THỨC

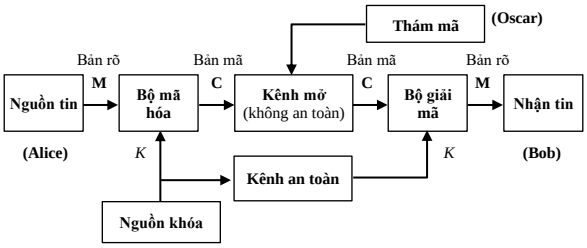
Ngô Đức Thiện
Học viện Công nghệ Bưu chính Viễn thông

Tóm tắt: Cho đến nay, cách thức mã hóa và giải mã của hệ mật khóa bí mật chủ yếu sử dụng các phép hoán vị, phép thay thế, lại ghép hai phép này, hoặc phép xử lý bit. Bài báo này đề xuất một phương pháp thực hiện hệ mật khóa bí mật nhưng dựa trên bài toán logarit rời rạc, trong đó phép mã hóa và giải mã được thực hiện bằng hàm lũy thừa các đa thức theo modulo, theo cách tương tự như hệ mật Pohlig-Hellman. Cùng với đó, bài báo cũng đề xuất thuật toán thực hiện hàm lũy thừa này.

Từ khóa: Hệ mật khóa bí mật, bài toán logarit rời rạc, hệ mật Pohlig-Hellman, vành đa thức, trường số.

I. GIỚI THIỆU

Hệ mật khóa bí mật [1], [3], [4]) (hay còn được biết đến là hệ mật khóa đối xứng) có lịch sử phát triển rất lâu đời. Phương pháp xây dựng hệ mật khóa bí mật cũng khá đơn giản, không có phép toán học nào đặc biệt mà chủ yếu dựa vào các phép thay thế, phép hoán vị, hoặc sử dụng cả hai phép này như các hệ mật DES hay AES; hoặc phương pháp xử lý bít như trong các hệ mật mã dòng (Stream cipher). Khi sử dụng lai ghép phép thay thế với phép hoán vị, thông thường các hệ mật đều hay sử dụng phép thay thế phi tuyến nhằm tăng độ an toàn. Sơ đồ chức năng của hệ mật khóa bí mật như hình 1.



Hình 1. Sơ đồ chức năng của hệ mật khóa bí mật

Hệ mật này có các ưu điểm nổi bật là tốc độ mã hóa và giải mã nhanh, hệ số mở rộng bản tin thấp. Chính vì thế các hệ mật khóa bí mật hay được dùng để mã hóa bảo mật dữ liệu hoặc trong các ứng dụng bảo mật thời gian thực.

Tuy nhiên, các nhược điểm lớn nhất của hệ mật này là việc sinh khóa, lưu trữ khóa và bảo vệ khóa khá phức tạp, nhất là khi số lượng người dùng trên mạng tăng cao. Ngoài ra, các hệ mật này còn phải sử dụng kênh an toàn để phân phối khóa dẫn đến chi phí tăng; hoặc phải sử dụng một giao thức thỏa thuận khóa an toàn. Các hệ mật này cũng khó thực hiện được các dịch vụ như xác thực, chữ ký số, thương mại điện tử…

Các hệ mật khóa công khai (hay hệ mật khóa bất đối xứng) thường được xây dựng trên các bài toán một chiều. Một trong các hàm một chiều sử dụng nhiều đó là bài toán logarit rời rạc, với các hệ mật như: trao đổi và thỏa thuận khóa Diffie-Hellman, hệ mật Omura-Massey, Pohlig-Hellman, hệ mật và chữ ký số ElGamal, hệ mật trên đường cong elliptic...

Bài toán logarit rời rạc thường được thực hiện trên trường số, các dữ liệu bản rõ và bản mã được biểu diễn bằng các con số nguyên dương trong trường số $GF(p)$ với p là số nguyên tố. Từ các nghiên cứu trong [6] cho thấy sự đẳng cấu giữa vành đa thức có 2 lớp kề cyclic với trường số, và do đó ta có thể thực hiện bài toán logarit rời rạc trên các đa thức, khi đó dữ liệu sẽ được mô tả bằng các đa thức.

Bài báo này đề xuất một phương pháp thực hiện một hệ mật mã khóa bí mật với phép mã hóa và giải mã là hãm lũy thừa các đa thức trên vành đa thức có hai lớp kề cyclic. Cùng với đó, bài báo cũng đề xuất thuật toán tính hàm lũy thừa của đa thức theo modulo. Cấu trúc bài báo chia thành 5 phần, sau phần giới thiệu là các phần: phần 2 hệ mật Pohlig-Hellman; phần 3 cấu trúc tựa đẳng cấu giữa vành đa thức có hai lớp kề cyclic và trường số; phần 4 đề xuất hệ mật Pohlig-Hellman trên vành đa thức có 2 lớp kề cyclic và cuối cùng là phần kết luận.

II. HỆ MẬT POHLIG-HELLMAN

A. Bài toán logarit rời rạc

Cho đến này chưa có thuật toán hiệu quả nào để giải bài toán logarit rời rạc tổng quát. Có nhiều thuật toán phức tạp, thường sinh ra từ những thuật toán tương tự như bài toán phân tích thừa số, chúng chạy nhanh hơn các thuật toán thô sơ, nhưng vẫn còn chậm hơn so với thời gian đa thức. Có thể kể đến một số thuật toán như: baby-step giant-step, Pollard, Pohlig-Hellman, COS, index calculus...

Tóm tắt bài toán logarit rời rạc như sau [1], [2]:

Tác giả liên hệ: Ngô Đức Thiện,
Email: thiennd@ptit.edu.vn
Đến tòa soạn: 5/2020, chỉnh sửa: 6 /2020, chấp nhận đăng: 7/2020.

Xét một vành số $\mathbb{Z}_p$, nếu p là nguyên tố thì $\mathbb{Z}_p$ là một trường ($\mathbb{Z}_p = GF(p)$). Tập tất cả các phần tử khác 0 của trường sẽ tạo nên một nhóm nhân cyclic $\mathbb{Z}_p^*$.

$$\mathbb{Z}_p^* = \mathbb{Z}_p / \{0\} = \{1,2, \dots, p-1\} \quad (1)$$

- Cho $g \in \mathbb{Z}_p^*$ là một phần tử sinh (nguyên thủy) của nhóm nhân.

- Cho $y \in \mathbb{Z}_p^*$, yêu cầu hãy tìm x (nếu tồn tại) sao cho: $g^x = y$, tức là: $x = \log_g y$.

Nhận xét: $\forall y \in \mathbb{Z}_p^*$ thì:

- Bài toán có nghiệm khi g là phần tử nguyên thủy.

- Bài toán có thể không có nghiệm khi g bất kỳ.

Ví dụ 1: Xét $p = 17$ và $g = 3$ là phần tử nguyên thủy của nhóm nhân $\mathbb{Z}_{17}^*$, ta có các giá trị 3^t và $\log_3 t$ như trong bảng 1 (Chú ý, các phép tính đều lấy theo modulo của 17).

BẢNG 1. GIÁ TRỊ HÀM MŨ VÀ LOGARIT RỜI RẠC CƠ SỐ 3 CỦA CÁC PHẦN TỬ TRONG NHÓM NHÂN $\mathbb{Z}_{17}^*$.

t	1	2	3	4	5	6	7	8
3^t	3	9	10	13	5	15	11	16
$\log_3 t$	16	14	1	12	5	15	11	10
t	9	10	11	12	13	14	15	16
3^t	14	8	7	4	12	2	6	1
$log_3 t$	2	3	7	13	4	9	6	8

Từ bảng 1 ta nhận thấy cả hàm mũ và hàm logarit rời rạc đều không phải hàm đồng biến và nó phân bố ngẫu nhiên. Với trường hợp p nhỏ thì việc tính $x = \log_g y$ dễ dàng có được từ việc tính toàn bộ các số $y = g^x$ như trong bảng 1. Nhưng khi p lớn (từ vài trăm bit trở lên) thì số lượng phép tính sẽ lớn hơn rất nhiều và khó có thể giải được.

Bài toán logarit rời rạc không phải lúc nào cũng khó, độ khó của nó phụ thuộc vào các nhóm nhân được lựa chọn. Ví dụ, các hệ mật dựa trên phép logarit rời rạc thường chọn các nhóm nhân $\mathbb{Z}_p^*$ trong đó p là số nguyên tố lớn. Tuy nhiên, nếu $p-1$ có thừa số là các số nguyên tố nhỏ, thì có thể sử dụng thuật toán Pohlig-Hellman để giải bài toán logarit rời rạc rất hiệu quả. Vì thế người ta thường lựa chọn p là số nguyên tố lớn an toàn, để thành lập nhóm nhân $\mathbb{Z}_p^*$ cho các hệ mật.

Một số nguyên tố an toàn là một số nguyên tố có dạng $p = 2q + 1$, với q là số nguyên tố lớn. Điều này đảm bảo $p - 1 = 2q$ có thừa số nguyên tố lớn và không dễ dàng có thể giải được bài toán logarit rời rạc bằng thuật toán Pohlig-Hellman.

B. Hệ mật Pohlig – Hellman

Bài toán logarit rời rạc là bài toán khó, trong khi bài toán lũy thừa rời rạc lại không khó (có thể tính bằng thuật toán nhân và bình phương). Trường hợp này, cũng giống như bài toán phân tích thừa số hay phép nhân các số nguyên, chúng đều có thể dùng để xây dựng cấu trúc cho một hệ mật mã.

Hệ mật Pohlig-Hellman là một hệ mật sử dụng bài toán logarit rời rạc, có thể tóm tắt hệ mật này như sau [1]:

- Chọn p là một số nguyên tố lớn và an toàn.

- Phép mã hóa được thực hiện theo phương trình đồng dư sau:

$$c \equiv m^e \bmod p \quad (2)$$

- Phép giải mã được thực hiện như sau:

$$m \equiv c^d \bmod p \quad (3)$$

Trong đó: m là bản rõ; c là bản mã; e là số mũ mã hóa và d là số mũ giải mã.

Số mũ mã hóa e (hay khóa) phải là số khả nghịch và do đó e phải thỏa mãn [1], [3], [4]:

$$\gcd(e, \varphi(p)) = 1 \quad (4)$$

Với $\varphi(p)$ là hàm Phi-Euler [1].

Do p là số nguyên tố nên $\varphi(p) = p - 1$, và như thế số mũ giải mã tương ứng d được tính từ phép nghịch đảo của $e \bmod \varphi(p)$ như sau:

$$de \equiv 1 \bmod (p-1) \quad (5)$$

Hệ mật Pohlig – Hellman có thể sử dụng làm hệ mật khóa bí mật thông thường vì rất dễ xác định d từ e và p. Thậm chí nếu giữ bí mật p thì nó có thể suy ra từ kích thước của khối bản mã.

III. CẤU TRÚC TỰA ĐẲNG CẤU CỦA VÀNH ĐA THỨC CÓ HAI LỚP KỀ CYCLIC VỚI TRƯỜNG SỐ

* *Định nghĩa 1*: Vành đa thức theo modulo $\mathbf{Z}_2[x]/(x^n + 1)$ được gọi là vành đa thức có hai lớp kề cyclic nếu phân tích $x^n + 1$ có dạng sau [5], [6]:

$$x^n + 1 = (x+1)\sum_{i=0}^{n-1} x^i \quad (6)$$

Trong đó: $(x + 1)$ và $\sum_{i=0}^{n-1} x^i$ là các đa thức bất khả quy.

Chú ý: *Các phép tính nhân và cộng các đa thức đều được tính theo modulo của* $x^n + 1$. *Tức là coi* $x^n + 1 = 0$ *hay* $x^n = 1 = x^0$ *(phép cộng là phép cộng mod 2). Và để thuận tiện, trong bài báo này tác giả sẽ không ghi* $\bmod(x^n + 1)$ *đối với các phép cộng, nhân và lũy thừa các đa thức.*

Trong các vành đa thức này tồn tại nhóm nhân cyclic có cấp cực đại [5], [6]:

$$G = \{[a(x)]^i, i = 1,2,3,\dots,k\} \quad (7)$$

Với:

$$k = \max ord\, a(x) = 2^{n-1} - 1 \quad (8)$$

* *Mối quan hệ giữa* $\mathbf{Z}_2[x]/(x^n + 1)$ *và* $GF(p)$

Xét một số nguyên tố p với $p = 2^n - 1$. Khi đó vành số modulo $\mathbf{Z}_p$ sẽ trở thành trường hữu hạn $GF(p)$ và trên trường này tồn tại một nhóm nhân cyclic [6]:

$\mathbf{Z}_p^* = \mathbf{Z}_p / \{0\}$ có cấp $| \phi_p^* | = p - 1 = 2^n - 2$

và $\forall a \in \mathbf{Z}_p^* \rightarrow \exists a^{-1} \in \mathbf{Z}_p^* : aa^{-1} \equiv 1 \bmod p$.

Xét $a(x) \in \mathbf{Z}_2[x]/(x^n + 1)$ với $a(x)$ có trọng số lẻ. Khi đó $\exists a^{-1}(x)$ với $W(a^{-1}(x))$ lẻ thỏa mãn:

$$a(x)a^{-1}(x) \equiv 1 \bmod (x^n + 1) \qquad (9)$$

Do vậy, có thể xây dựng phép tương ứng sau [6]:

$$a(x) = \sum_{i \in I} f_i x^i \in \mathbf{Z}_2[x]/(x^n + 1)$$
$$\text{a} \quad a = \sum_{i \in I} f_i 2^i \in \mathbf{Z}_p^*$$

và coi lũy đẳng $e_0(x) = \sum_{i=0}^{n-1} x^i = 0$.

Khi đó ta có thể coi đây là một ánh xạ 1-1 giữa các phần tử của $\mathbf{Z}_2[x]/(x^n + 1)$ với các phần tử của $GF(p)$. Như vậy, vành đa thức có hai lớp kề cyclic và trường $GF(p)$ với $p = 2^n - 1$ (p – nguyên tố) được gọi là tựa đẳng cấu (quasi-isomorphism). Ta có thể so sánh việc thực hiện các phép toán cộng và nhân trên hai cấu trúc này như bảng 2.

BẢNG 2: PHÉP CỘNG VÀ NHÂN TRÊN CẤU TRÚC VÀNH ĐA THỨC VÀ TRƯỜNG SỐ.

Phép tính	Vành đa thức $\mathbf{Z}_2[x]/(x^n + 1)$	Trường số $GF(p)$
Phép cộng	$a(x) = \sum_{i \in I \cap \mathbf{Z}_n} a_i x^i$; $b(x) = \sum_{j \in J \cap \mathbf{Z}_n} b_j x^j$ $c(x) = a(x) + b(x)$ $= \sum_{k \in K \cap \mathbf{Z}_n} c_k x^k$ $K = (I \cup J) - (I \cap J)$	$a,b \in GF(p)$ $c = a + b$ $\equiv (a + b) \bmod p$
Phép nhân	$c(x) = a(x)b(x)$ $(\equiv a(x)b(x) \bmod (x^n + 1))$	$c = ab$ $(\equiv ab \bmod p)$

Quan hệ tựa đồng cấu chỉ xảy ra đối với một số vành đa thức có hai lớp kề cyclic đặc biệt, các vành đa thức này được liệt kê dưới đây [7].

Số nguyên tố Mersenne: $p = 2^n - 1$

$n =$ 2, 3, **5**, 7, 13, 17, **19**, 31, **61**, 89, **107**, 127, 52, 607, 1279, 2203, 3217, **4253**, 9689, **9941**, 19937,… , 74207281.

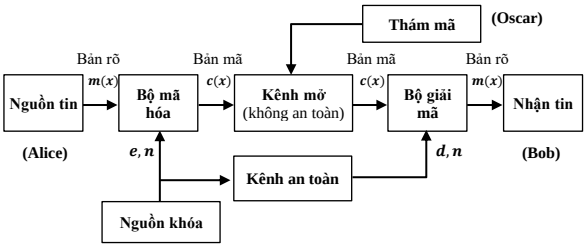
Vành đa thức có hai lớp kề cyclic [5], [6]:

$n =$ **5**, 11, **19**, 29, 37, 53, 59, **61**, 67, 83, 101, **107**, 131,… , 523, 613, 1277, 2213, 3203, 3253, **4253**, …, **9941**,…

IV. HỆ MẬT POHLIG-HELLMAN TRÊN VÀNH ĐA THỨC CÓ HAI LỚP KỀ CYCLIC

A. *Mô tả hệ mật*

Trong phần này tác giả đề xuất một phương pháp xây dựng hệ mật khóa bí mật theo cách của Pohlig-Hellman. Tuy nhiên, hàm mã hóa và giải mã đều là hàm lũy thừa các đa thức theo modulo, bản rõ $m(x)$ và bản mã $c(x)$ đều biểu diễn bằng các đa thức. Mô hình truyền tin của hệ mật như mô tả trong hình 2.



Hình 2. Mô hình truyền tin của hệ mật Pohlig-Hellman xây dựng trên vành đa thức

Mô tả hệ mật như sau:

+ *Tạo khóa*:

Bên Alice tạo khóa bí mật: e,d,n theo các bước sau:

Bước 1: chọn số n thỏa mãn:

- $\mathbf{Z}_2[x]/(x^n + 1)$ là vành đa thức có 2 lớp kề cyclic.
- $p = 2^n - 1$ là một số nguyên tố.

Bước 2: Tính $k = 2^{n-1} - 1$ và chọn số mũ mã hóa e thỏa mãn điều kiện:

$$\gcd(e,k) = 1$$

Chú ý: $\gcd(e,k)$ là ước chung lớn nhất của e,k.

Sở dĩ ta lấy ước chung lớn nhất của e với k là do k chính là cấp cực đại của một phần tử trong vành đa thức $\mathbf{Z}_2[x]/(x^n + 1)$ như trong biểu thức (8).

Bước 3: tìm số mũ giải mã d thỏa mãn:

$$de \equiv 1 \bmod k \qquad (10)$$

Có nhiều cách để giải phương trình (10) tuy nhiên cách hiệu quả nhất là sử dụng thuật toán Euclid [2], [3].

Khóa mã hóa của Alice là e,n còn khóa giải mã của Bob là bộ số d,n (hoặc ngược lại). Alice gửi khóa giải mã cho Bob qua kênh an toàn, hoặc sử dụng một thủ tục trao đổi khóa an toàn nào đó.

+ *Mã hóa*:

Bên Alice cần mã hóa một bản tin rõ là đa thức $m(x) \in \mathbf{Z}_2[x]/x^n + 1$, Alice tính:

$$c(x) = m^e(x) \qquad (11)$$

Sau đó Alice sẽ gửi $c(x)$ đến Bob qua kênh mở.

+ *Giải mã:*

Bob nhận bản mã $c(x)$, khóa giải mã d,n và tiến hành giải mã theo phương trình sau:

$$m(x) = c^d(x) = [m^e(x)]^d = m^{ed}(x) = m(x) \quad (12)$$

Ví dụ 2:

+ Tạo khóa:

Bước 1: Alice chọn $n = 5$ thỏa mãn $x^5 + 1$ là vành đa thức có hai lớp kề cyclic và $p = 2^5 - 1 = 31$ là số nguyên tố.

Bước 2: Alice tính $k = 2^{5-1} - 1 = 15$ và chọn $e = 13$ thỏa mãn $\gcd(13, k) = \gcd(13, 15) = 1$

Bước 3: Tính $d = 7$ thỏa mãn $7.13 \equiv 1 \bmod 15$

+ Mã hóa:

Giả sử Alice cần gửi bản tin rõ $m(x)$ cho Bob:

$$m(x) = 1 + x^3 + x^4 \leftrightarrow (0,3,4)$$

Chú ý: $(0,3,4)$ là biểu diễn dạng *số mũ* của đa thức $1 + x^3 + x^4 = x^0 + x^3 + x^4$.

Alice tính:

$$c(x) = m^e(x) = (1 + x^3 + x^4)^{13} = x + x^2 + x^3 \leftrightarrow (1,2,3)$$

Sau đó Alice gửi $c(x)$ qua kênh mở cho Bob.

+ Giải mã:

Bob nhận $n = 5, d = 7$ và $c(x)$ và giải mã:

$$m(x) = c^d(x) = (x + x^2 + x^3)^7 = (1 + x^3 + x^4) \leftrightarrow (0,3,4)$$

B. Thuật toán tính lũy thừa của đa thức theo modulo $x^n + 1$

Thông thường các hệ mật sử dụng bài toán logarit rời rạc đều phải thực hiện lũy thừa các số theo modulo trên trường số và người ta thường sử dụng thuật toán nhân và bình phương [1], [3], [4].

Với hệ mật đề xuất như trong bài báo cũng phải thực hiện phép lũy thừa nhưng là lũy thừa đa thức theo modulo của $x^n + 1$. Dựa vào một tính chất đặc biệt của đa thức sau đây, bài báo đưa ra thuật toán tính lũy thừa cho đa thức.

Xét đa thức $a(x) \in \mathbf{Z}_2[x]/x^n + 1$:

$$a(x) = a_0 x^0 + a_1 x^1 + a_2 x^2 + ... + a_{n-1} x^{n-1}$$

với $a_i \in [0,1]$.

Biểu diễn dạng số mũ (chỉ cho các $a_i = 1$):

$$a(x) \leftrightarrow \hat{a} = (a_0 0, a_1 1, a_2 2, ..., a_{n-1}(n-1))$$

+ Nếu một số k có dạng $k = 2^u$ khi đó:

$$[a(x)]^k = [a(x)]^{2^u} = \sum_{i=0}^{n-1} a_i x^{i.k \bmod n} \quad (13)$$

Dạng mũ:

$$(\hat{a})^k = (a_0 0k \bmod n,\ a_1 1k \bmod n,\ a_2 2k \bmod n, ..., a_{n-1}(n-1)k \bmod n) \quad (14)$$

Chứng minh:

$$[a(x)]^k = [a(x)]^{2^u} = ((([a(x)]^2)^2)^{2...2}) \quad (u \text{ lần})$$

Mà :

$$[a(x)]^2 = \sum_{i=0}^{n-1} a_i^2 x^{2i \bmod n} + 2\sum_{\substack{i,j=0; \\ i \neq j}}^{n-1} a_i a_j x^{(i+j) \bmod n}$$

Ta thấy với $i \neq j$:

$$2a_i a_j x^{(i+j) \bmod n} = a_i a_j x^{(i+j) \bmod n} + a_i a_j x^{(i+j) \bmod n} = 0$$

do phép cộng đa thức là cộng modulo 2.

Vì thế: $2\sum_{\substack{i,j=0; \\ i \neq j}}^{n-1} a_i a_j x^{(i+j) \bmod n} = 0$

Vậy ta có: $[a(x)]^2 = \sum_{i=0}^{n-1} a_i^2 x^{2i \bmod n}$

Tương tự như thế ta tính được:

$$[a(x)]^4 = ([a(x)]^2)^2 = \sum_{i=0}^{n-1} (a_i^2 x^{2i \bmod n})^2 = \sum_{i=0}^{n-1} a_i^4 x^{4i \bmod n}$$

Tổng quát:

$$[a(x)]^{2^u} = \sum_{i=0}^{n-1} a_i^{2^u} x^{2^u i \bmod n} = \sum_{i=0}^{n-1} a_i x^{2^u i \bmod n}$$

Chú ý: do $a_i \in [0,1]$ nên $a_i^{2^u} = a_i$

Điều phải chứng minh

Ví dụ 3: xét $n = 5; a(x) = 1 + x^3 + x^4 \leftrightarrow \hat{a} = (0,3,4)$

- Nếu $k = 2$ thì:

$$[a(x)]^2 = 1 + x^{3.2 \bmod 5} + x^{4.2 \bmod 5} = 1 + x + x^3$$

- Nếu $k = 2^3 = 8$ thì:

$$(\hat{a})^8 = (0*8 \bmod 5,\ 3*8 \bmod 5,\ 4*8 \bmod 5) = (0,4,2) = (0,2,4)$$

Tức là để tính lũy thừa $[a(x)]^k$ *ta chỉ việc nhân các số mũ của từng đơn thức* x *trong* $a(x)$ *với* k *rồi lấy modulo theo* n *như biểu thức (13), (14).*

Dựa vào tính chất này của đa thức ta có thể tính lũy thừa bất kỳ cho đa thức $a(x)$ như sau:

Cho k nguyên dương và có phân tích như sau:

$$k = \sum_t 2^{u_t} = \sum_t k_t$$

Ví dụ: $k = 19 = 2^0 + 2^1 + 2^4 = 1 + 2 + 16$

$\hat{u} = (0,1,4)$; $\bar{k} = [k_t] = [1,2,16]$

Khi đó phép lũy thừa $[a(x)]^k \bmod (x^n + 1)$ có thể tính như sau:

$$[a(x)]^k = \prod_t [a(x)]^{k_t} = \prod_t [a(x)]^{2^{u_t}}$$

Thuật toán tính lũy thừa của đa thức theo modulo $x^n + 1$ như sau:

Thuật toán: Tính lũy thừa các đa thức theo modulo $x^n + 1$
Vào: n, $\hat{a} = (a_1, a_2, ..., a_r)_{1\times r}$, $\bar{k} = [k_1, k_2, ..., k_t]_{1\times t}$ **Ra:** $\hat{b} = (\hat{a})^k \bmod (x^n + 1)$
[**1**] $\hat{b} \leftarrow (0)$, if $k = 0$ then return $\hat{b}$ [**2**] For i from 1 to t do: [2.1] for j from 1 to r do: $A_j \leftarrow a_j . k_i \bmod n$ [2.2]: $\hat{b} \leftarrow \hat{b}.\hat{A}$ [**3**] **Return** $(\hat{b})$

Chú thích

+ Số n đảm bảo $\mathbf{Z}_2[x]/x^n + 1$ là vành đa thức có 2 lớp kề cyclic và $p = 2^n - 1$ là số nguyên tố.
+ Đa thức $a(x) \in \mathbf{Z}_2[x]/x^n + 1$; dạng số mũ $a(x) \leftrightarrow \hat{a} = (a_1, a_2, ..., a_r)_{1\times r}$ độ dài $\hat{a}$ là $r \le n$.
+ Số nguyên k, $(0 \le k \le 2^{n-1} - 1)$; k được biểu diễn thành một vector bao gồm t số thập phân $\bar{k} = [k_1, k_2, ..., k_t]_{1\times t}$; trong đó $k_i = 2^{u_t}$:

$$k = \sum_t k_t \leftrightarrow \bar{k} = [k_t]_{1\times t}$$

+ Mục [1] $\hat{b} = (0) \leftrightarrow b(x) = 1 = 2^0$;
+ Mục [2.1] tập các số A_j là biểu diễn dạng mũ của đa thức $A(x)$; $A(x) \leftrightarrow \hat{A} = (A_1, A_2, ..., A_r)$. Trong một số ngôn ngữ lập trình (như Matlab) có thể dễ dàng tính được ngay cho toàn bộ các phần tử trong $\hat{A}$ mà không cần phải dùng vòng lặp. Tức là ta có thể tính trực tiếp $(A_j) \leftarrow (a_j . k_i \bmod n): j = 1,2,...,r$.
+ Mục [2.2] là phép nhân đa thức theo modulo, đây là phép nhân bình thường trên vành đa thức được lấy theo modulo của $x^n + 1$.
+ Kết quả dạng mũ: $\hat{b} = (\hat{a})^k \bmod (x^n + 1)$

Ví dụ 4:

xét $n = 5$; $a(x) = 1 + x^2 + x^4 \leftrightarrow \hat{a} = (0,2,4)_{1\times 3}$ và

$k = 13 = 1 + 4 + 8 = 2^0 + 2^2 + 2^3$, biểu diễn k như sau: $\bar{k} = [1,4,8]_{1\times 3}$. Ta có: $r = 3; t = 3$

Khi đó $\hat{b} = \hat{a}^{13}$ được tính như sau:

[1] $\hat{b} \leftarrow (0)$

[2] For i from 1 to 3 do:

- $i = 1$: (với $k_1 = 1$)

$+ \hat{A} = (A_1, A_2, A_3)$

$= (0*1 \bmod 5,\ 2*1 \bmod 5,\ 4*1 \bmod 5)$

$= (0,2,4)$

$+ \hat{b} \leftarrow (0)*(0,2,4) = (0,2,4)$

- $i = 2$: (với $k_2 = 4$)

$+ \hat{A} = (A_1, A_2, A_3)$

$= (0*4 \bmod 5, 2*4 \bmod 5, 4*4 \bmod 5)$

$= (0,3,1)$

$+ \hat{b} \leftarrow (0,2,4)*(0,3,1) = (0,1,4)$

- $i = 3$: (với $k_3 = 8$)

$+ \hat{A} = (A_1, A_2, A_3)$

$= (0*8 \bmod 5,\ 2*8 \bmod 5,\ 4*8 \bmod 5)$

$= (0,1,2)$

$+ \hat{b} \leftarrow (0,1,4)*(0,1,2) = (1,3,4)$

[3] Return $\hat{b} = (1,3,4)$

Vậy kết quả có được là:

$$(1 + x^2 + x^4)^{13} \bmod (x^5 + 1) = x + x^3 + x^4$$

Tiến hành mô phỏng thuật toán nêu trên bằng phần mềm Matlab (phiên bản R2016a), cấu hình máy tính: chip Intel Core i5 (7th gen), RAM 8GB, hệ điều hành Windows 64 bits.

Với mỗi bộ tham số mô phỏng được thực hiện 5000 lần và sau đó lấy trung bình thời gian tính toán, một số kết quả có được như trong bảng 3.

BẢNG 3: THỜI GIAN XỬ LÝ CỦA THUẬT TOÁN

TT	Tham số mô phỏng	Thời gian xử lý (ms)
1	$n = \mathbf{5}$; $k = 13$; $\hat{a} = (0,3,4)$	0,050
2	$n = \mathbf{19}$; $k = 103.567$ $\hat{a} = (0,2,5,8,10,11,13,15,17)$	0,164

3	$n = \mathbf{61}$; $k = 1.239.878$ $\hat{a} = (1,3,7,12,19,21,29,$ $32,38,45,50,55,59)$	0,236
4	$n = \mathbf{107}$; $k = 2.341.235.671$ $\hat{a} = (1,9,17,26,38,47,54,$ $62,74,82,91,98,105)$	0,436
5	$n = \mathbf{4253}$; $k = 139.749.574.567$ $\hat{a} = (1,56,98,147,209,300,478,698,$ $1002,1348,2034,3045,4002)$	4,300
6	$n = \mathbf{9941}$; $k = 13.974.957.456.787.957$ $\hat{a} = (0,100,456, 989, 1456, 2002,$ $2560, 3001, 3982, 4679, 5398,$ $6003,7623,7982,8567,9234,9657)$	19,300

Nhận xét: Với các giá trị n nhỏ thì tốc độ tính toán là nhanh. Với trường hợp $n = 4253$ tương đương với việc tính toán với các con số 4252 bit mà thời gian tính toán của một phép lũy thừa là 4,3ms có thể nói là hoàn toàn chấp nhận được. Cho đến hiện nay để đảm bảo tính an toàn, các hệ mật cũng chỉ dùng các con số từ 1000 đến 2000 bit. Với trường hợp $n = 9941$ thời gian tính toán với khả năng của máy tính laptop như cấu hình ở trên là $19,3ms$. Cho đến nay thì chưa cần thiết dùng đến giá trị n lớn như vậy, trong tương lai có thể sử dụng đến với các con số lớn hơn, khi đó tốc độ tính của máy tính cũng như các chip xử lý sẽ nhanh hơn thời điểm hiện tại và như thế sẽ rút ngắn được thời gian tính toán và hoàn toàn có thể áp dụng được hệ mật này.

V. KẾT LUẬN

Bài báo đề xuất phương pháp thực hiện một hệ mật khóa bí mật theo cách của hệ mật Pohlig-Hellman. Trong đó, bản rõ và bản mã được mô tả bằng các đa thức trên vành đa thức, thay vì được mô tả bằng các con số trên trường số. Sở dĩ có thể thực hiện được điều này là nhờ cấu trúc tựa đẳng cấu của vành đa thức có hai lớp kề cyclic với trường số.

Độ an toàn của hệ mật này tương đương với độ an toàn của các hệ mật khác xây dựng trên bài toán logarit rời rạc, cho đến nay với giá trị số nguyên tố lớn thì bài toán logarit rời rạc vẫn là bài toán khó.

Để hệ mật có tính khả thi, bài báo cũng đề xuất thuật toán thực hiện hàm lũy thừa cho các đa thức theo modulo. Các kết quả mô phỏng cho thấy tốc độ tính toán của thuật toán với tường hợp các số lớn là rất khả quan để áp dụng vào thực tế.

TÀI LIỆU THAM KHẢO

[1] Nguyễn Bình, Giáo trình Mật mã học, Học viện Công nghệ BCVT, 2013.

[2] Frederik Vercauteren, Discrete Logarithms in Cryptography, ESAT/COSIC - K.U. Leuven ECRYPT Summer School 2008.

[3] Jean-Yves Chouinard, ELG 5373, "Secure communications and data encryption," School of Information Technology and Engineering, University of Ottawa, April 2002.

[4] A. Menezes, P. van Oorschot, and S. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1996.

[5] Nguyen Trung Hieu, Ngo Duc Thien, Tran Duc Su, "On Constructing Cyclic Multiplicative Groups with Maximum Order over Polynomial Rings with Two Cyclotomic Cosets", Jounal of scientific research and military technology, Vol. 17, February - 2012, pp. 133-140, ISSN 1859-1043.

[6] Lê Danh Cường, Nguyễn Bình, "Cấu trúc tựa đẳng cấu giữa vành đa thức có 2 lớp kề cyclic và trường số", Tạp chí Khoa học và Công nghệ các trường đại học kỹ thuật, ISSN 2354-1083, số 121, 2017, tr. 54-57.

[7] Nguyễn Trung Hiếu, Ngô Đức Thiện, "Hệ mật Omura-Massey xây dựng trên vành đa thức có hai lớp kề cyclic", Tạp chí khoa học và Công nghệ các trường đại học kỹ thuật, ISSN 2354-1083, số 125, 2018, tr. 29-34.

ONE METHOD OF IMPLEMENTING POHLIG-HELLMAN CRYPTOSYSTEM OVER POLYNOMIAL RINGS

Abstract: Until now, the methods of encryption and decryption of a symmetric cryptosystem mainly used some operations, such as permutation, substitution or combine these two operations, or bit processing (in stream ciphers). This paper proposes a new method for implementing a symmetric cryptosystem but is based on discrete logarithm problem, in which encryption and decryption are performed by polynomial exponential function with modulo, in a manner like the Pohlig-Hellman cryptosystem. Along with that, this article also proposed an algorithm to implement that exponential function.

Ngô Đức Thiện, Nhận học vị Tiến sỹ năm 2010. Hiện công tác tại Học viện Công nghệ Bưu chính Viễn thông.
Lĩnh vực nghiên cứu: Lý thuyết thông tin và mã hóa, mật mã.