MỘT PHƯƠNG PHÁP HIỆU QUẢ ĐỂ SINH DÃY GIẢ NGẪU NHIÊN KIỂU LỒNG GHÉP PHI TUYẾN VỚI BẬC LỚN

Đặng Vân Trường
Viện Nghiên cứu Khoa học - Công nghệ Mật mã, Ban Cơ yếu Chính phủ

***Tổng quan*:** Trong nhiều thập kỷ qua, dãy giả ngẫu nhiên dựa trên m-dãy đã có quá trình nghiên cứu phát triển liên tục, với nhiều ứng dụng rộng khắp trong kỹ thuật, đặc biệt là kỹ thuật mật mã và kỹ thuật truyền thông. Các tác giả đã đưa ra nhiều thiết kế nhằm tăng tính an toàn của dãy, nhất là tính phi tuyến. Nhóm nghiên cứu của tác giả Lê Chí Quỳnh đã đưa ra phương pháp sinh dãy giả ngẫu nhiên kiểu lồng ghép và lồng ghép phi tuyến. Để có thể ứng dụng dãy lồng ghép phi tuyến vào mật mã, cần sử dụng các dãy có bậc rất cao để tăng tính an toàn. Trong bài báo này, tác giả sẽ đưa ra phương pháp sinh dãy lồng ghép phi tuyến với bậc rất lớn, đồng thời đánh giá hiệu quả thuật toán trên khía cạnh lý thuyết và từ các tính toán thực nghiệm.

Từ khóa: dãy giả ngẫu nhiên, dãy lồng ghép, dãy lồng ghép phi tuyến, bình phương và nhân.

I. GIỚI THIỆU CHUNG

M-dãy, hay dãy có chu kỳ cực đại được phát triển từ lý thuyết trường Galois, được tác giả S.W. Golomb hệ thống hóa và mở ra các hướng ứng dụng trong việc sinh các dãy giả ngẫu nhiên phục vụ truyền thông [1]. Dãy giả ngẫu nhiên dựa trên m-dãy đã có nhiều ứng dụng rất rộng rãi trong công nghệ viễn thông hiện nay, từ thuật toán trải phổ sử dụng trong CDMA, họ thuật toán mã hóa bảo vệ kênh GSM A5/1, A5/2 và A5/3, hoặc thuật toán xáo trộn dữ liệu phục vụ các kênh truyền thông SATA, USB và ngay cả truyền dẫn quang SDH.

Trong kỹ thuật mật mã, khi sử dụng một m-dãy riêng lẻ sẽ không đủ tính an toàn để bảo mật thông tin, nên các nhà khoa học đã đưa ra một số giải pháp xây dựng nên các hệ mã dòng dựa trên m-dãy, trong đó đưa ra các cấu trúc riêng, kết hợp nhiều m-dãy để tăng tính phi tuyến cũng như các tính chất mật mã của hệ mật. Ở Việt Nam có nhóm nghiên cứu của tác giả Lê Chí Quỳnh đã có nhiều năm theo đuổi hướng nghiên cứu riêng về cấu trúc lồng ghép để xây dựng các bộ tạo dãy lồng ghép dựa trên m-dãy[2, 3, 4] và ứng dụng vào nhiều bài toán [5, 6]. Các tác giả tập trung vào vấn đề nghiên cứu lý thuyết, hướng đến việc xây dựng các dãy có độ phức tạp cao và tính chất tốt. Tuy nhiên về mặt thực hành, một số phương pháp đã đưa ra còn có độ phức tạp tính toán lớn, khó có khả năng ứng dụng trong thực tế khi bậc của dãy tăng lên. Để có thể ứng dụng trong kỹ thuật mật mã, yêu cầu về bậc của dãy còn tiếp tục tăng lên để chống lại sức mạnh tính toán của các siêu máy tính và các thiết bị thám mã chuyên dụng.

Trong bài báo này, chúng tôi sẽ đề xuất phương pháp sinh dãy lồng ghép hiệu quả với các dãy có bậc rất lớn dựa trên đặc điểm của các tham số trong thiết kế của dãy. Sau đó, chúng tôi đánh giá hiệu quả của phương pháp trên khía cạnh độ phức tạp tính toán cũng như tài nguyên cần sử dụng, đồng thời kiểm chứng các đánh giá này bằng các tính toán thực nghiệm.

II. PHƯƠNG PHÁP SINH DÃY LỒNG GHÉP DỰA TRÊN M-DÃY

2.1 Thiết kế dãy lồng ghép dựa trên m-dãy

Giả sử ta có m-dãy $\{b_n\}$ được sinh bởi đa thức $g(d)$ có bậc n trên trường GF(p^n). Gọi $b(d)$ là biến đổi - d của $\{b_n\}$

$$b(d) = \frac{S(d)}{g(d)} \quad (1)$$

Ta lựa chọn n sao cho $n = m*l$

Chu kỳ của dãy ban đầu $N = p^n-1$

Chọn bước lồng ghép $T = \frac{N}{L}$ với $L = p^m-1$

Ta sinh ra toàn bộ chu kỳ của m-dãy ban đầu và xếp thành một ma trận theo từng hàng liên tiếp. Dãy lồng ghép sẽ có được bằng cách lấy lần lượt các dãy con theo từng cột của ma trận.

Để sinh ra dãy lồng ghép ta sử dụng tính chất sau

Tính chất 1 [2]: Mỗi dãy con theo 1 cột là một m-dãy được sinh từ một đa thức sinh duy nhất $g_1(x)$ có bậc m.

$$b(d) = \sum_{i=0}^{T-1} d^i F_i(d^T) \quad (2)$$

Như vậy để sinh ra dãy lồng ghép ta chỉ cần xác định được tập các bước dịch pha của các dãy con tương ứng với các cột là có thể sinh ra toàn bộ dãy lồng ghép.

2.2 Các phương pháp sinh dãy lồng ghép

Có ba phương pháp xây dựng dãy lồng ghép đã được giới thiệu [2]

a) Phương pháp sử dụng d-Transform

Bước 1: mở rộng $F_i(d)$ lên T lần (chèn T-1 bit 0 vào giữa hai bit liên tiếp của $F_i(d)$), trong biến đổi d, điều này tương đương với việc thay thế d bằng d^T trong (2)

Bước 2: Biểu diễn theo biến đổi d của $\{b_n\}$ theo cách để xen kẽ các đa thức $F_i(d)$, (hoặc chèn T pha khác nhau của $F_i(d)$ để tạo thành $b(d)$)

Tác giả liên hệ: Đặng Vân Trường,
Email: dv_truong@yahoo.com
Đến tòa soạn: 20/11/2021, chỉnh sửa: 24/03/2022, chấp nhận đăng: 25/4/2022.

Bước 3: Ta đặt $D = d^T$

- Tìm các bước dịch pha

$$F_i(D) = \frac{S_i(D)}{g_1(D)} \quad (3)$$

Sau đó nhóm lại biến đổi *d* của *b(d)* thành:

$$b(d) = \sum_{i=0}^{s-1} d^i F_i(D) \quad (4)$$

So sánh phần $F_i(D)$ với bảng biến đổi *d* của của dãy con, ta có thể xây dựng lên toàn bộ các bậc lồng ghép I_P.

b) Phương pháp sử dụng hàm vết

Với *m*, *n* là các số nguyên dương sao cho *m* chia hết *n*, α là phần tử sinh của trường hữu hạn GF(p^n). Khi đó, hàm vết $Tr_m^n(x)$ sẽ ánh xạ các phần tử của GF(p^n) vào GF(p^m) theo quan hệ sau [12]:

$$Tr_m^n(x) = \sum_{k=0}^{\frac{n}{m}-1} x^{p^{mk}} \quad (5)$$

Tập hợp thứ tự lồng ghép $I_P = I_P^0, I_P^1, \dots, I_P^{T-1}$ sẽ được xác định như sau:

$$I_P^j = \begin{cases} i \quad if \quad Tr_m^n(\alpha^j) = \alpha^{i.T} \quad i = 0,1,\dots,p^m - 1 \\ \infty \quad if \quad Tr_m^n(\alpha^j) = 0 \quad j = 0,1,\dots,T-1 \end{cases} \quad (6)$$

c) Phương pháp tính trước m-hàng của ma trận

Thực tế khi xây dựng chương trình trên máy tính để cài đặt hai phương pháp phân rã m-dãy nêu trên, việc thực hiện cả hai phương pháp đều không hiệu quả, đặc biệt là khi độ dài dãy tăng lên đáng kể.

Ta có thể tìm ra những phần tử đầu tiên của bậc lồng ghép bằng cách tính trực tiếp như sau:

Trước hết ta sinh ra phần đầu của chuỗi $\{b_n\}$ với trạng thái ban đầu được cho trước, nhưng thay vì tạo chuỗi đầy đủ chu kỳ (p^n-1 phần tử), ta chỉ cần tính toán cho *m.T* giá trị đầu tiên.

Tiếp đó ta sẽ sắp xếp lại các giá trị này bằng cấu trúc lồng ghép như định nghĩa của dãy lồng ghép, từ đó ta có thể nhận được các trạng thái ban đầu của dãy con $F_i(d)$. Vị trí của mỗi trạng thái này trong dãy con độ dài đầy đủ chính là giá trị tương ứng trong tập thứ tự lồng ghép I_P.

2.3 Phương pháp sinh dãy lồng ghép phi tuyến dựa trên m-dãy

Dãy lồng ghép nêu trên mang theo rất nhiều thuộc tính của dãy ban đầu, đặc biệt là tính tuyến tính. Để tăng tính phi tuyến của dãy lồng ghép, đã có một phương pháp sử dụng hai dãy lồng ghép với hai đa thức sinh khác nhau, song sử dụng cùng các tham số: bậc *n* và bước lồng ghép *T*, để từ đó sinh ra dãy lồng ghép phi tuyến như sau [3]:

- dãy thứ nhất sử dụng để xác định tập thứ tự lồng ghép I_P
- dãy thứ hai sử dụng để xác định đa thức sinh cho dãy con $f_1(x)$ và toàn bộ chu kỳ dãy con với 2^m-1 phần tử

Dãy lồng ghép phi tuyến được sinh ra bằng cách sử dụng tập thứ tự lồng ghép từ dãy thứ nhất, song dãy con toàn chu kỳ lại sử dụng của dãy thứ hai. Các tác giả đã sử dụng biến đổi *d* để chứng minh rằng dãy lồng ghép phi tuyến có khoảng tương đương tuyến tính (ELS – equipvalence linear span) tốt hơn dãy lồng ghép đơn[3].

2.4 Nhận xét về các phương pháp sinh dãy lồng ghép

Cả ba phương pháp đã nêu đều có thể sử dụng để sinh ra toàn bộ dãy lồng ghép mà không cần lập ma trận lồng ghép. Ta không cần tính toán toàn bộ dãy đầu vào với bậc *n*, chỉ cần tính toán trên dãy con bậc *m*.

Với giá trị *T* đủ nhỏ, ta có thể lưu trữ được toàn bộ tập thứ tự lồng ghép I_P trong bộ nhớ máy tính hoặc thiết bị nhúng. Để có thể ứng dụng trong kỹ thuật mật mã, các yêu cầu về bậc của m-dãy cần tăng lên, cụ thể là giá trị bậc *n* cần lớn cỡ 128 hoặc cao hơn. Khi này giá trị $T \approx p^{m-n}$ sẽ tăng lên rất nhanh theo hàm mũ. Điều đó dẫn tới việc lưu trữ toàn bộ tập I_P là không khả thi. Với phương pháp sử dụng hàm vết, ta chỉ cần tính toán một phần đầu của tập I_P, từ đó sinh ra phần đầu tiên của dãy lồng ghép với kích thước tùy ý theo yêu cầu.

Khi giá trị *T* tăng lên rất lớn, với cả ba phương pháp nêu trên ta cần phải tính toán các đa thức có bậc *T* nên cần tiêu tốn rất nhiều thời gian. Với giá trị $T>10^{12}$, các tính toán này là không khả thi trong thực tế. Trong phần tiếp theo của bài báo này, ta sẽ tìm cách tính toán một phần đầu của tập thứ tự lồng ghép I_P với chi phí chấp nhận được trong thực hành.

III. THUẬT TOÁN SINH DÃY GIẢ NGẪU NHIÊN LỒNG GHÉP VỚI BẬC LỚN

3.1 Thuật toán sinh dãy giả ngẫu nhiên lồng ghép phi tuyến với bậc lớn

Theo các phương pháp sinh dãy lồng ghép phi tuyến đã trình bày trong phần II, để sinh một dãy ta cần thực hiện hai pha: Pha tiền xử lý để tìm thứ tự lồng ghép và pha sinh dãy thực hiện ghép nối các dãy con thành dãy lồng ghép đầu ra. Trong phần này sẽ trình bày thuật toán tiền xử lý để tính được các phần tử đầu tiên của tập thứ tự lồng ghép một cách hiệu quả khi bậc của dãy là rất lớn.

Thuật toán tiền xử lý tìm thứ tự lồng ghép

Ký hiệu trạng thái trong của m-dãy tại thời điểm *t* là $S_{(t)}$, theo công thức sinh m-dãy[12] ta có

$$S_{(T)} = \frac{S_{(0)} \times d^T}{g(d)} \quad (7)$$

Trong đó $S_{(0)}$ là trạng thái khởi đầu của m-dãy, *g(d)* là đa thức sinh của m-dãy

Ta chú ý tới giá trị tham số *T* của dãy lồng ghép [2]

$$T = \frac{N}{L} \text{ với } L = p^m\text{-}1 \text{ và } N = p^n\text{-}1 \quad (8)$$

Chú ý là $n = m*l$, vậy ta có thể viết

$$N = p^{m*l}\text{-}1 \text{ hay } N = (p^m)^l - 1 \quad (9)$$

Để đơn giản, đặt $Q = p^m$, ta có thể viết

$$T = \frac{Q^l - 1}{Q - 1} = Q^{l-1} + Q^{l-2} + \dots + Q + 1 \quad (10)$$

Nói một cách khác, nếu biểu diễn giá trị *T* theo cơ số *Q* là

(l số 1)

$$T = 111 \dots 111_Q$$

Nếu biểu diễn T theo cơ số p, biểu diễn của *T* cũng chỉ có *l* số 1, chen giữa các số 1 là *m*-1 số 0

$$T = 100...00100...00100 \dots 001_p$$

Dựa vào biểu diễn cơ số p của *T*, ta sẽ tìm cách tính nhanh giá trị đa thức kết quả cho một phần của (7).

Đặt $U_T(d) = \frac{d^T}{g(d)}$ (11)

Ta sẽ đưa ra công thức và thuật toán để tính $U_T(d)$ khi biết giá trị của T, trong trường hợp giá trị T rất lớn.

Ta có

$$U_1(d) = d \quad (12)$$

$$U_Q(d) = \frac{d^Q}{g(d)} \quad (13)$$

Chú ý rằng

$$d^{Q^k} = d^{Q^{k-1} \times p} = \left(d^{Q^{k-1}}\right)^Q \quad (14)$$

Nên ta có

$$U_{Q^k}(d) = \frac{\prod_{j=1}^{Q} U_{Q^{k-1}}(d)}{g(d)} \quad (15)$$

$$U_T(d) = \prod_{k=0}^{l-1} U_{Q^k}(d) \quad (16)$$

Vì $Q = p^m$ có thể lên tới hàng triệu khi bậc của đa thức ban đầu tăng lên, ta cần có phương pháp hiệu quả để tính được các công thức (13), (16).

Để tính $U_Q(d)$ ta sẽ tính lần lượt từng bước như sau. Trước hết tính trực tiếp đa thức

$$U_p(d) = \frac{d^p}{g(d)} \quad (17)$$

Từ chú ý của (14) ta cũng có

$$U_{p^k}(d) = \left(U_{p^{k-1}}(d)\right)^p \quad (18)$$

Ta có thể tính $U_{p^k}(d)$ theo công thức sau

$$U_{p^k}(d) = \frac{\prod_{j=1}^{p} U_{p^{k-1}}(d)}{g(d)} \quad (19)$$

Từ đó tính được

$$U_Q(d) = U_{p^m}(d) \quad (20)$$

Để tính $U_{Q^k}(d)$ từ $U_{Q^{k-1}}(d)$ Ta sẽ sử dụng phương pháp tương tự

Chú ý rằng $U_{p^0 Q^{k-1}}(d) = U_{Q^{k-1}}(d)$, từ đó ta có thể tính theo phương pháp truy hồi

$$U_{p^i Q^{k-1}}(d) = \frac{\prod_{j=1}^{p} U_{p^{i-1} Q^{k-1}}(d)}{g(d)} \quad (21)$$

$$U_{Q^k}(d) = U_{p^m Q^{k-1}}(d) \quad (22)$$

Cuối cùng, sử dụng các giá trị tính bởi (22) ta tính được đa thức (16).

Phương pháp bình phương và nhân

Có một cách hiệu quả hơn để tính các công thức (19) và (21). Khi ta có biểu diễn của *p* dưới dạng nhị phân thành tập các bit $\{p_i\}$ với i=0..r, sau đó sử dụng phương pháp bình phương và nhân để tính đa thức kết quả của công thức (19) theo thuật toán trình bày dưới dạng giả mã như sau:

Thuật toán 1: Tính $U_Q(d)$

Đầu vào:

Biểu diễn nhị phân của *p:* $\{p_j\}$, j=0..r

$U_{p^{k-1}}(d)$

Đầu ra: $U_{p^{k-1}}(d)$

Thuật toán:

$U^* = 1$

$U_{tmp} = U_{p^{k-1}}(d)$

For j = 0 to r do

If $p_j = 1$ *then*

$U^* = U^* \times U_{tmp}$

$U_{tmp} = \frac{(U_{tmp})^2}{g(d)}$

Next j

Sau khi kết thúc vòng lặp ta có $U^* = U_{p^k}(d)$

Áp dụng thuật toán *m* lần với *k* nhận các giá trị từ 1 tới *m* (chú ý là $U_{p^0}(d) = U_1(d) = d$ theo *(12))* ta sẽ tính được giá trị của (20). Kết quả đầu ra của thuật toán lần trước sẽ sử dụng làm đầu vào cho thuật toán lần sau,

Bằng phương pháp tương tự ta tính được đa thức kết quả của công thức (22) như sau

Thuật toán 2: Tính $U_{Q^k}(d)$

Đầu vào:

Biểu diễn nhị phân của *p:* $\{p_j\}$, j=0..r

$U_{p^{i-1}Q^{k-1}}(d)$

Đầu ra: $U_{p^i Q^{k-1}}(d)$

Thuật toán:

$V^* = 1$

$V_{tmp} = U_{p^{i-1}Q^{k-1}}(d)$

For j = 0 to r do

If $p_j = 1$ *then*

$V^* = V^* \times V_{tmp}$

$V_{tmp} = \frac{(V_{tmp})^2}{g(d)}$

Next j

Sau khi kết thúc vòng lặp ta có $V^* = U_{p^i Q^{k-1}}(d)$

Áp dụng thuật toán *m*-1 lần với *i* nhận các giá trị từ 1 tới *m*-1, kết quả của lần thuật toán lần trước là đầu vào cho thuật toán lần sau, ta sẽ tính được giá trị của (22)

Thuật toán sinh dãy lồng ghép phi tuyến dựa trên m-dãy

Trong bước tiền xử lý, ta sử dụng *Thuật toán 1* để tính giá trị $U_Q(d)$, sau đó sử dụng *Thuật toán 2* để tính các giá trị $U_{Q^k}(d)$. Tiếp đó tính $U_T(d)$ theo (16) và lưu trữ $U_T(d)$ để sử dụng sau này.

Khi có giá trị trạng thái khởi đầu của dãy lồng ghép $S_{(0)}$, sử dụng (7) ta tính được $S_{(T)}$, trong đó giá trị $U_T(d)$ đã được tính toán và lưu trữ từ bước tiền xử lý. Tiếp tục sử dụng (7) song thay $S_{(0)}$ bằng $S_{(T)}$ ta sẽ có $S_{(2T)}$… từ đó tính được *m* bộ trạng thái của dãy ban đầu ứng với *m* phần tử bắt đầu các cột trong ma trận, ký hiệu là $\{S_{(k.T)}\}$. Lấy *m* phần tử đầu tiên trong bộ trạng thái này chính là giá trị *m* bit đầu tiên trong cột thứ nhất trong ma trận. Theo *Tính chất 1*, từ giá trị *m* bit này ta có thể tính được toàn bộ các giá trị của cột thứ nhất. Nếu đã có toàn bộ chu kỳ của dãy con, từ *m* bit này ta cũng có thể tìm được giá trị thứ tự lồng ghép của cột đầu tiên.

Với thuật toán tiền xử lý nêu trên, ta tìm ra được giá trị *m* phần tử đầu tiên của cột thứ nhất trong ma trận lồng ghép

(là *m* bit số 0 trong *m* bộ trạng thái). Từ các giá trị này ta có thể xây dựng phần đầu tiên của dãy lồng ghép bằng dãy con đầu tiên[2].

Để tiếp tục xây dựng các phần tiếp theo của dãy lồng ghép, ta tìm thứ tự lồng ghép tiếp theo bằng cách lấy các bit tiếp sau trong bộ *m* trạng thái $\{S_{(k.T)}\}$. Nếu đã sử dụng hết *n* bit trong các bộ *m* trạng thái, ta lại tiếp tục sử dụng từng trạng thái của$\{S_{(k.T)}\}$để xác định các trạng thái tiếp theo $S_{(k.T+1)}$ theo (7), từ đó có được giá trị khởi đầu của cột *n*+1. Như vậy ta có thể xây dựng dãy lồng ghép có độ dài bất kỳ mà không cần tính trước toàn bộ bảng thứ tự lồng ghép.

Trong trường hợp cần sinh dãy lồng ghép phi tuyến, ta áp dụng tất cả các bước nêu trên với các tham số của dãy lồng ghép thứ nhất, điểm khác biệt duy nhất là khi sinh ra các dãy con (các cột của ma trận lồng ghép) ta sẽ sử dụng đa thức sinh dãy con của dãy lồng ghép thứ hai.

3.2 Đánh giá độ phức tạp của thuật toán sinh dãy giả ngẫu nhiên lồng ghép với bậc lớn

Để tính được $U_Q(d)$ ta cần (*m*-1).(*p*-1) phép nhân đa thức trên trường GF(p^n) để tính công thức (20). Trong trường hợp tính bằng phương pháp bình phương và nhân, số phép nhân cần tính là (*m*-1).$log_2 p$ (Ta coi thời gian tính phép bình phương và phép nhân đa thức trên trường GF(p^n) là tương đương nhau)

Để tính được $U_Q{}^k(d)$ từ giá trị $U_Q{}^{k-1}(d)$ theo (22) ta cần tính các phép nhân đa thức trên trường GF(p^n). Xét trường hợp sử dụng phương pháp bình phương và nhân thay cho (19) và (21) thì số phép nhân đa thức là:

$$V_{mulq} = (m\text{-}1).log_2 p \qquad (23)$$

Để tính được $U_T(d)$ theo (16) ta cần sử dụng thêm (*l*-1) phép nhân đa thức. Tổng số phép nhân đa thức trên trường GF(p^n) là:

$$V_q = (l\text{-}1).(m\text{-}1).log_2 p + (l\text{-}1). \qquad (24)$$

Chú ý là khi n $\to \infty$, m và l đều có cỡ tương đương n. Ta có thể coi $log_2 p$ là hằng số, xét phép nhân đa thức trên trường GF(p^n) có độ phức tạp tính toán là n thì độ phức tạp tính toán của v_q là cỡ $O(n^3)$.

So sánh với phương pháp bình phương và nhân áp dụng trực tiếp cho giá trị số mũ *T* bằng một thuật toán tương tự như *Thuật toán 1*. Ta sẽ biểu diễn giá trị của *T* thành $log_2 T$ bit $\{t_i\}$ để áp dụng bình phương và nhân.

Số phép bình phương đa thức cần tính là

$$V_{mul2} = log_2 T = (n\text{-}m)*log_2 p = m.(l\text{-}1).log_2 p \quad (25)$$

Số phép nhân cần tính sẽ có giá trị trung bình là *v*/2 (do phân bố bit 0/1 trong biểu diễn nhị phân của *T* là đều nhau).

Vậy tổng số phép nhân đa thức cần tính là

$$V_2 = {}^3/_2 \cdot v_{mul2}$$

$$V_2 = {}^3/_2 \; m.(l\text{-}1).log_2 p \qquad (26)$$

Phương pháp tính theo biểu diễn cơ số *p* có được lợi thế hơn phương pháp biểu diễn nhị phân bởi vì trong biểu diễn cơ số *p* của *T* có rất nhiều phần tử bằng 0 theo (10), trong khi biểu diễn nhị phân của *T* không có được lợi thế này.

Để so sánh cụ thể số bước tính toán giữa hai trường hợp tính toán với cơ số p và tính toán theo phương pháp bình phương và nhân (trên biểu diễn cơ số 2 của *T*) ta xét bảng 1 và bảng 2 như sau

Trong 2 bảng này, các giá trị *p*, *n*, *m* và *T* là tham số của dãy lồng ghép. V_{mulq} và V_{mul2} được tính theo (23) và (25). Các giá trị $\boldsymbol{n_{bit1\text{-}p}}$ và $\boldsymbol{n_{bit1\text{-}2}}$ là số vị trí trong biểu diễn của tham số T theo cơ số p và cơ số 2 tương ứng. Giá trị $\boldsymbol{V_q}$ được tính theo (24), còn để tính chính xác giá trị thực hành của $\boldsymbol{V_2}$ ta sẽ đếm số bit 1 thực sự của T theo biểu diễn nhị phân thay vì dùng (26).

BẢNG 1: SỐ BƯỚC TÍNH TOÁN TIỀN XỬ LÝ CHO DÃY LỒNG GHÉP VỚI CƠ SỐ P

Nº	p	n	m	T	V_{mulq}	$n_{bit1\text{-}p}$	V_q
1	2	24	8	65 793	17	3	**20**
2	3	24	8	43 053 283	32	3	**35**
3	7	24	8	33 232 936 334 403	48	3	**51**
4	7	28	14	678 223 072 850	42	2	**44**
5	13	12	3	10 609 328 380	36	4	**40**
6	13	19	8	10 604 499 374	32	2	**34**
7	29	12	3	14 507 740 823 580	45	4	**49**
8	31	12	3	26 440 509 694 144	45	4	**49**

BẢNG 2: SỐ BƯỚC TÍNH TOÁN TIỀN XỬ LÝ CHO DÃY LỒNG GHÉP BẰNG PHƯƠNG PHÁP BÌNH PHƯƠNG VÀ NHÂN VỚI THAM SỐ T BIỂU DIỄN TRÊN CƠ SỐ 2

Nº	p	n	m	T	V_{mul2}	$n_{bit1\text{-}2}$	V_2
1	2	24	8	65 793	17	3	**20**
2	3	24	8	43 053 283	26	12	**39**
3	7	24	8	33 232 936 334 403	45	24	**67**
4	7	28	14	678 223 072 850	40	23	**60**
5	13	12	3	10 609 328 380	34	17	**51**
6	13	19	8	10 604 499 374	41	17	**61**
7	29	12	3	14 507 740 823 580	44	19	**66**
8	31	12	3	26 440 509 694 144	45	12	**67**

Để tính được giá trị $\boldsymbol{n_{bit1\text{-}2}}$ trong bảng 2, ta tính chính xác giá trị tham số T theo (8), sau đó chuyển đổi giá trị T sang biểu diễn nhị phân và đếm số bit 1 để có giá trị $\boldsymbol{n_{bit1\text{-}2}}$. Do đó trong một số trường hợp giá trị $\boldsymbol{n_{bit1\text{-}2}}$ không chính xác là $log_2 T/2$ theo (26). Ví dụ ở dòng 4, ta có *T* = 678 223 072 850_{10}, biểu diễn nhị phân là

$T = 10011101111010010011111010110011100101001 0_2$

Như vậy $\boldsymbol{n_{bit1\text{-}2}}$ nhận giá trị 23 thay vì $log_2 T/2 = 20$

So sánh 2 bảng trên, tập trung vào giá trị Vq và V2 ta thấy phương pháp tính toán với cơ số *p* có hiệu quả tốt hơn phương pháp tính toán bình phương và nhân trực tiếp với số bước ít hơn khoảng 25%, khi cơ số *p* có giá trị lớn. Với cơ số *p* = 2, hai phương pháp có số bước giống nhau do cùng là tính toán trên cơ số 2. Trong một số trường hợp, phương pháp tính toán trên cơ số p còn cho hiệu quả tốt hơn rất nhiều.

IV. KẾT LUẬN

Trong bài báo này tác giả đã đề xuất một thuật toán mới, đây có thể xem là một phương pháp mới để sinh dãy lồng ghép phi tuyến với bậc lớn với hiệu quả cao, có khả năng ứng dụng trong thực hành. Việc đánh giá độ phức tạp tính toán cho thấy thuật toán này có độ phức tạp tính toán ở mức hàm mũ, chính xác là cỡ $O(n^3)$ so với bậc n của đa thức sinh m-dãy. Bằng cách khai thác một đặc điểm của tham số của dãy lồng ghép, thuật toán này có lợi thế hơn khoảng 25% so với thuật toán bình phương và nhân thông thường, hiệu quả sẽ tăng lên khi giá trị cơ số p lớn. Thuật toán này giúp cho việc sinh dãy lồng ghép phi tuyến trở nên hiệu quả trong thực hành khi bậc của dãy lớn lên, có thể thỏa mãn các yêu cầu về dãy có bậc rất lớn phục vụ cho kỹ thuật mật mã.

TÀI LIỆU THAM KHẢO

[1] S.W. Golomb, Shift Register Sequences, San Francisco, Holden-Day, 1967.

[2] L.M. Hieu and L.C. Quynh, *Design and Analysis of Sequences with Interleaved Structure by d-Transform*, IETE Journal of Research, vol. 51, no. l, pp.61-67, Jan-Feb. 2005.

[3] Hieu Le Minh, Truong Dang Van, Binh Nguyen Thanh and Quynh Le Chi, *Design and Analysis of Ternary m-sequences with Interleaved Structure by d-Transform*, Journal of Information Engineering and Applications, vol.5, no.8, pp.93-101, 2015.

[4] Truong Dang Van, Binh Nguyen Thanh, Hieu Le Minh and Quynh Le Chi, *Construction of Nonlinear q-ary m-sequences with Interleaved Structure by d-Transform*, IEEE ICCE 2018, pp.389-392, 2018

[5] Quynh Le Chi, Cuong Nguyen Le, Thang Pham Xuan, *A hardware oriented method to generate and evaluate nonlinear interleaved sequences with desired properties*, Journal of Information Engineering and Applications, Vol.6, No.7, 2016

[6] Nguyen Van Son, et. al, *FPGA Implemen-tation of Optimal PN-Sequences by Time-Multiplexing Technique*, International Conference on Engineering Research and Applications (ICERA), 2019

[7] Nguyen Van Son, Le Chi Quynh, Tran Vu Kien, Dao Huy Du, Dang Khanh Hoa, "*FPGA Implemen-tation of Optimal PN-Sequences by Time-Multiplexing Technique*", International Conference on Engineering Research and Applications (ICERA), 2019.

[8] L.C.Quynh, S.Prasad, *A class of binary cipher sequences with best possible correlation funtion*, IEEE Proceeding Part F .Dec 1985. vol 132.pp.560-570.

[9] G. Cattaneo, G. De Maio, and U. F. Petrillo. "*Security issues and attacks on the GSM standard: a review*". Journal of Universal Computer Science, vol. 19, no. 16, pp. 2437–2452, 2013.

[10] G. Gong, "*New design for signal sets with low cross correlation, balance property and large linear span - GF(p) case*", IEEE Trans. Inform. Theory, vol 48, no. 11, pp.2847-2867, Nov. 2002.

[11] G. Gong, *Theory and application of q-ary interleaving sequences*, IEEE Trans. Inform. Theory, vol41, pp. 400-411, March 1995

[12] Jing He, *Interleaved Sequences Over Finite Fields*, Carleton University, Ottawa 2013 .

[13] X.D.Lin and K.H.Chang: "*Optimal PN sequences design for quasisynchronous CDMA communication systems*", IEEE Trans. Comm.vol 45. pp 221-226. Feb 1997 .

[14] R.J. McEliece, Finite fields for computer scientists and engineers, Springer, 1987.

[15] J.S. No, P-ary unified sequences: *P-ary extended d form sequences with the ideal autocorrelation property*, IEEE Trans. Inform. Theory, vol 48, no. 9, pp. 2540-2546, Sept 2002

[16] Alan Shamir, "*Stream Cipher: Dead or Alive?*", Asia Crypt, 2014.

[17] X.H.Tang, F.Z.Fan: "*A class of PN sequences over GF (P) with low correlation zone*", IEEE Trans. Inform. Theory, vol.41, no.4, pp. 1644-1649, May 2001.

AN EFFECTIVE METHOD TO GENERATE NONLINEAR INTERLEAVED SEQUENCES WITH LARGE ORDER

Abstracts: Over the past several decades, m-sequence-based pseudo-random sequences have experienced continuous development, with wide applications in engineering, communication and especially in cryptography. The research group of author Quynh.LC has introduced the method for generation of interleaved and nonlinearly interleaved sequences. In order to apply nonlinearly interleaved sequences in cryptography, it is necessary to use sequences of very high order. In this paper, the author will present a method to generate nonlinearly interleaved sequences with very large order and evaluate the efficiency of the implementation in terms of both theory and experimental calculations.

Keywords: pseudo-random sequences, interleaved sequences, nonlinearly interleaved sequences, large order, square and multiply

Đặng Vân Trường, Nghiên cứu sinh tại Học viện Công nghệ Bưu chính Viễn thông. Hiện công tác tại Viện Khoa học Công nghệ Mật mã - Ban Cơ yếu Chính phủ. Lĩnh vực nghiên cứu: Phát triển phần mềm bảo mật dữ liệu và bảo mật mạng