Đề xuất tầng tuyến tính và đánh giá khả năng cài đặt trong xây dựng mã khối 256 bit có cấu trúc FLC

Trần Sỹ Nam, Nguyễn Văn Long, Nguyễn Bùi Cương

***Tóm tắt*— Trong bài báo này, nhóm tác giả đề xuất tầng tuyến tính an toàn và hiệu quả trong cài đặt để hướng tới việc xây dựng mã khối kích thước 256-bit có tên là FLC. Cụ thể, nhóm tác giả đã sinh ra ma trận MDS kích thước 8×8 có dạng lũy thừa của ma trận đồng hành trên trường hữu hạn. Khi xem xét sử dụng ma trận này trong xây dựng mã khối 256-bit có cấu trúc FLC, nhóm tác giả thực hiện đánh giá khả năng cài đặt của quá trình mã hóa/giải mã trên nhiều nền tảng khác nhau. Các kết quả thực nghiệm chỉ ra rằng cài đặt của mã khối với đề xuất này hoàn toàn phù hợp để tích hợp trong các ứng dụng thực tế.**

***Abstract*— In this paper, we propose a secured and efficient linear diffusion layer that suitable for block cipher with 256-bit block size named FLC. Specifically, we have generated an 8×8 MDS matrix that has the power of a companion matrix over a finite field. We also perform a performance evaluation of the encryption/decryption process on various platforms. Our experimental results show that block cipher with our proposed linear diffusion layer is perfectly suitable for real life applications.**

Từ khóa— tầng tuyến tính; hiệu quả cài đặt; cấu trúc FLC; mã khối.

Keywords— linear diffusion layer; efficient implementation; FLC structure; block cipher.

I. GIỚI THIỆU

Hầu hết các mã khối hiện đại thường được thiết kế dựa trên mã pháp lặp, trong đó thực hiện lặp lại các biến đổi đơn giản, hiệu quả trong cài đặt để đảm bảo tính khuếch tán và xáo trộn theo nguyên lý của C. Shanon [1]. Thông thường tính xáo trộn được đảm bảo bởi những ánh xạ phi tuyến mà đại diện là các S-hộp, còn khuếch tán là tầng tuyến tính. Các S-hộp thường được sử dụng với kích thước nhỏ để giảm chi phí cài đặt, do đó để tăng độ an toàn cho mã khối, các tầng tuyến tính được sử dụng song hành để khuếch tán tính phi tuyến mà những S-hộp này mang lại. Dưới góc nhìn hiệu quả sử dụng, tầng tuyến tính thường là biến đổi chậm nhất và đòi hỏi nhiều chi phí cài đặt trong mã khối. Vì vậy, việc nghiên cứu lựa chọn tầng tuyến tính không những liên quan trực tiếp đến độ an toàn của mã khối kháng lại một số thám mã quan trọng như thám mã lượng sai và thám mã tuyến tính, mà còn ảnh hưởng đến khả năng thực thi của chính mã khối đó. Do đó, một thuật toán mật mã trước khi được đưa vào sử dụng, ngoài việc kiểm chứng về mặt an toàn, cần phải được đánh giá cẩn thận về mặt hiệu năng và tính mềm dẻo trong cài đặt trên các nền tảng khác nhau.

Đối với xây dựng thuật toán mật mã đối xứng hậu lượng tử, một hướng giải quyết đơn giản nhất là tăng kích thước khối dữ liệu được xử lý ở trạng thái trong của thuật toán. Tuy nhiên, việc tăng kích thước khối kéo theo nhiều vấn đề trong thực thi và triển khai các thuật toán mật mã. Những khó khăn đó xuất phát từ việc làm tăng chi phí cài đặt, khó khăn triển khai trên nền tảng phần cứng bởi khả năng đáp ứng tài nguyên của môi trường phần cứng là có hạn. Hiện nay, trên thế giới không có nhiều thuật toán với kích thước khối 256 bit. Điển hình với lớp thuật toán này có mã khối Rijndael-256 và Kalyna của Ukraina [2]. Tuy nhiên đối với Rijndael khi ban hành chuẩn AES thì chỉ có kích thước khối 128-bit. Đối với Kalyna, đây là một thuật toán định hướng cho các từ có 64 bit, được thiết kế dựa trên nền tảng của Rjindael và có khả năng cài đặt hiệu quả trong

Bài báo được nhận ngày 14/6/2022. Bài báo được nhận xét bởi phản biện thứ nhất ngày 21/6/2022 và được chấp nhận đăng ngày 28/6/2022. Bài báo được nhận xét bởi phản biện thứ hai ngày 22/6/2022 và được chấp nhận đăng ngày 24/6/2022.

phần mềm với các bảng tra được tính trước. Khi cài đặt ở dạng bitslice đối với môi trường tài nguyên hạn chế và trên nền tảng ngôn ngữ phần cứng thì Kalyna tỏ ra thiếu sự cân bằng giữa quá trình mã hóa/giải mã. Cụ thể, ma trận tuyến tính được lựa chọn với các hệ số nhỏ, dễ dàng cài đặt đối với quá trình mã hóa. Tuy nhiên, ma trận ở quá trình giải mã lại khó thực thi hơn rất nhiều. Chi tiết có thể tham khảo trong tài liệu [3].

Đối với bài toán xây dựng tầng tuyến tính có kích thước khối lớn như 256 bit, hiện tại có nhiều phương pháp khác nhau để đảm bảo vấn đề này. Các dạng ma trận MDS khuếch tán cao thường được lựa chọn, chúng được thể hiện ở các dạng khác nhau, như: ma trận MDS dịch vòng [4], Hadamard [5], hay ma trận MDS là lũy thừa của ma trận đồng hành [6, 7],... Một thực tế tồn tại là khi tăng kích thước ma trận, việc lựa chọn các hệ số của nó sao cho dễ dàng cài đặt vẫn là bài toán được nhiều nhà mật mã quan tâm. Ví dụ như trong AES [1, 8], hệ số trong các ma trận ở biến đổi MixColumns và InvMixColumns đem đến sự mất cân bằng trong cài đặt dạng bitslice ở quá trình mã hóa và giải mã của thuật toán này. Vấn đề tương tự cũng xuất hiện trong thuật toán Kalyna [3]. Các ma trận MDS dạng Hadamard cuộn [4], Hadamard [5] được đề xuất để giải quyết vấn đề trên, nhưng yếu điểm cố hữu là chúng tồn tại quá nhiều điểm bất động, đây có thể là điểm yếu có thể khai thác của những thám mã trong tương lai. Một cách khác giải quyết vấn đề mất cân bằng thực thi giữa quá trình mã hóa và giải mã, là sử dụng các ma trận MDS là lũy thừa của ma trận đồng hành. Điều kiện đặt ra là các ma trận đồng hành với đa thức liên kết có dạng đối xứng. Một ví dụ điển hình cho cách giải quyết theo hướng này có thể tìm thấy trong tầng tuyến tính của thuật toán Kuznyechik trong chuẩn GOST R 34.12-2015 của Liên bang Nga [9].

Đóng góp của nhóm tác giả: Với những phân tích như trên để hướng tới việc xây dựng một thuật toán mã khối hoàn chỉnh có kích thước khối 256-bit, nhóm tác giả tiến hành kiểm nghiệm khả năng thực thi (tính ứng dụng) của thuật toán với tham số ma trận MDS sinh được. Kết quả của quá trình này có ý nghĩa quan trọng để khẳng định tính hiệu quả khi triển khai ứng dụng của mã khối. Trên cơ sở như vậy, những đóng góp chính của bài báo như sau:

- Đề xuất lựa chọn dạng ma trận MDS kích thước 8×8 là lũy thừa của ma trận đồng hành: Lựa chọn các hệ số của ma trận đồng hành trong $\mathbb{F}_{2^8}$ đảm bảo hiệu quả trong cài đặt; Đưa ra một biểu diễn đơn giản cho phép dễ dàng và tiện lợi cho cài đặt tầng tuyến tính trong phần cứng và phần mềm ở dạng bitslice.

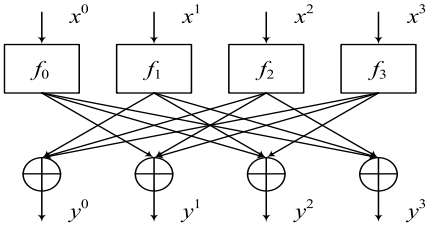
- Thực nghiệm cài đặt phần cứng và phần mềm đối với quá trình mã hóa/giải mã cho mã khối cấu trúc FLC với ma trận MDS được đề xuất trên môi trường cụ thể. Kết quả cài đặt chỉ ra rằng, đề xuất của nhóm tác giả không chỉ đảm bảo độ an toàn cho mã khối nói trên mà còn có những thông số cài đặt phù hợp để tích hợp và sử dụng trong các ứng dụng thực tế.

Phần tiếp theo của bài báo được tổ chức như sau: Trong phần II, các kiến thức cơ sở bao gồm tổng quan về mã khối có cấu trúc FLC với kích thước khối 256 bit; và những khái niệm, ký hiệu liên quan sẽ được giới thiệu. Phần III là trình bày về đề xuất ma trận cụ thể. Trong phần IV nhóm tác giả đưa ra giải pháp cài đặt hiệu quả, phân tích cài đặt phần cứng, phần mềm và cài đặt thực nghiệm. Cuối cùng là kết luận và danh mục tài liệu tham khảo.

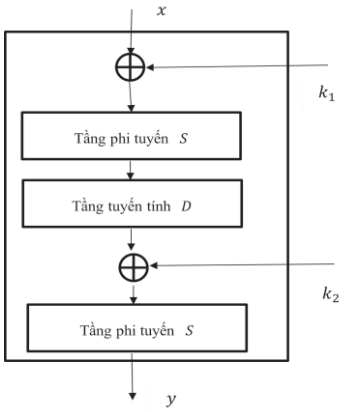
II. KIẾN THỨC CƠ SỞ

A. Lược tả mã khối 256 bit với cấu trúc FLC và thể hiện SDS

Nhóm tác giả xem xét một lược đồ mã hóa có tên là FLC, cho phép xây dựng mã khối lặp. Một vòng của FLC là một hoán vị tác động lên các từ có kích thước $4w$-bit (Hình 1). Trong đó, các hàm $f_i, i = 0,1,2,3$ là các hoán vị con giống nhau với đầu vào là một từ w bit. Một vòng mã biến đổi yêu cầu áp dụng song song 4 hoán vị $f_i, i \in \{0,1,2,3\}$...



Hình 1. Mô tả 1 vòng của lược đồ lặp FLC



Hình 2. Hàm cơ sở f_w và FLC-SDS

Trong mô hình mã pháp trên (Hình 2), một thể hiện của mã pháp FLC có dạng SDS, với 02 tầng biến đối phi tuyến S đóng vai trò xáo trộn và xen giữa là tầng biến đổi tuyến tính D đóng vai trò khuếch tán. Các hoán vị con f_0, f_1, f_2, f_3 đều có dạng hàm cơ sở f_w với các xâu có độ dài w-bit. Trong đó, hàm cơ sở f_w bao gồm tầng cộng XOR khóa vòng, tầng phi tuyến S để đảm bảo tính xáo trộn của thuật toán và tầng biến đổi tuyến tính D cho trạng thái w-bit.

Với kích thước khối 256 bit, các tham số được lựa chọn là $w = 64$, tầng phi tuyến S sử dụng các S-hộp 8-bit có dạng nghịch đảo x^{-1} và biến đổi D sử dụng ma trận MDS có kích thước 8×8 trên trường $GF(2^8)$. Mã khối với kích thước khối 256 bit hoạt động trong 6, 7 và 8 vòng tương ứng với khóa 256 bit, 384 bit và 512 bit. Trong phần này, nhóm tác giả không tập trung trình bày và phân tích lược đồ khóa để có thể tập trung sâu hơn về phân tích vài trò của tầng tuyến tính đề xuất và sự ảnh hưởng của nó đến khả năng cài đặt của thuật toán. Độ an toàn tổng thể của thuật toán cũng vậy, hai vấn đề này sẽ được nhóm tác giả trình bày ở các công bố tiếp theo.

Để thuận tiện phân tích cài đặt cho từng đề xuất, nhóm tác giả sử dụng kí hiệu FLC-SDS-256/R để chỉ việc mô tả mã pháp FLS-SDS với kích cỡ khối 256-bit với các số vòng R. Tiếp theo, nhóm tác giả tập trung xây dựng và thực thi cài đặt ma trận MDS 8×8 trên trường $GF(2^8)$. Việc lựa chọn tham số cho biến đổi này không chỉ ảnh hưởng đến độ an toàn của thuật toán mã khối, mà còn ảnh hưởng đến khả năng cài đặt hiệu quả của nó trên các nền tảng khác nhau để xây dựng một mã khối FLC-SDS.

B. Ma trận MDS và một số khái niệm cơ bản

Gọi $\mathbb{F}_2$ là trường hữu hạn gồm 2 phần tử 0 và 1, và $\mathbb{F}_{2^n}$ là trường hữu hạn gồm 2^n phần tử với đa thức sinh bất khả quy $f(x) = x^n \oplus \sum_{i=1}^{n-1} a_i x^i \oplus 1, a_i \in \mathbb{F}_2$. Một phần tử của trường $\mathbb{F}_{2^n}$ là một đa thức có bậc $n-1$ cùng với các hệ số của nó thuộc $\mathbb{F}_2$. Một ma trận M được ký hiệu là $(a_{i,j})$, ở đây $a_{i,j}$ là phần tử ở hàng i, cột j của ma trận đó. Ký hiệu "$\oplus$" hoặc "+" là phép cộng XOR và "×" hoặc "$\cdot$" là phép nhân hai phần tử trong trường $\mathbb{F}_{2^n}$. Tiếp theo, chúng ta sẽ xem xét khái niệm ma trận MDS (maximum distance separable matrix).

Định nghĩa 1 [7, 8]: *Cho p, q là 2 số nguyên, một ánh xạ từ $\mathbb{F}_{2^p}$ vào $\mathbb{F}_{2^q}$ được xác định bởi ma trận M kích thước $p \times q$, $x \mapsto M \times x$. Ta nói rằng M là một ma trận MDS nếu tập của tất cả các cặp $(x, M \times x)$ là mã MDS, có nghĩa là mã tuyến tính có số chiều bằng p, chiều dài là p+q và khoảng cách nhỏ nhất bằng q*+1.

Tiếp theo, ma trận A có dạng:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 \\ z_0 & z_1 & \dots & \dots & \dots & z_{d-1} \end{pmatrix} \quad (1)$$

gọi là ma trận đồng hành (*companion matrix*) của đa thức $z_0 + z_1 x + z_2 x^2 + \cdots + z_{d-1} x^{d-1} + x^d$. Đối với những ma trận dạng này ta dễ dàng tìm được ma trận nghịch đảo của nó. Nó có dạng:

$$A^{-1} = \begin{pmatrix} z_1 z_0^{-1} & z_2 z_0^{-1} & z_3 z_0^{-1} & \dots & z_{d-1} z_0^{-1} & z_0^{-1} \\ 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \end{pmatrix} \quad (2)$$

Tồn tại những ma trận dạng đồng hành như trong (1) mà lũy thừa bậc d hoặc lớn hơn d của nó là một ma trận MDS [7, 8]. Phép nhân với ma trận dạng A có khả năng tính toán truy hồi, đây là một lợi thế khi chọn các ma trận MDS dạng này khi xây dựng tầng tuyến tính.

Các tham số đặc trưng cho một biến đổi tuyến tính sử dụng trong các hệ mã là số nhánh của nó và số điểm bất động. Số nhánh của biến đổi tuyến tính L ký hiệu là $B(L)$, chính là số lượng nhỏ nhất các S-hộp tích cực trong 2 vòng mã liên tiếp. Với khái niệm điểm bất động của tầng tuyến tính có

thể tham khảo trong [10]. Trong thiết kế, cần xây dựng các ma trận là MDS để đảm bảo số nhánh cực đại và không có điểm bất động để đảm bảo dữ liệu chạy qua nó được thực sự thay đổi. Đây có thể nói là những tiêu chí về mặt an toàn cần phải được đảm bảo. Ngoài ra, tiêu chí hiệu quả cài đặt trên nhiều nền tảng và môi trường khác nhau cũng cần được xem xét đối với ma trận MDS. Trong phần tiếp theo, nhóm tác giả sẽ đề xuất các ma trận MDS là lũy thừa của ma trận đồng hành cụ thể, sau đó phân tích khả năng cài đặt của chúng.

III. Xây dựng ma trận MDS cho tầng tuyến tính của FLC-SDS-256

Mã khối với kích thước khối 256 bit sử dụng tầng biến đổi tuyến tính dựa trên ma trận MDS kích thước 8×8. Trạng thái của mã khối được biểu diễn dưới dạng bảng chữ nhật hoặc ma trận kích thước 8×4 (8 hàng, 4 cột), trong đó các phần tử đều nằm trong trường cơ sở được lựa chọn. Phép biến đổi tuyến tính D và nghịch đảo của nó được thực thi thông qua phép nhân trên trường cơ sở của ma trận tuyến tính kích thước 8×8 với ma trận trạng thái kích thước 8×4.

Như đã phân tích ở những mục trước, ma trận MDS được xây dựng dựa trên ma trận đồng hành có nhiều lợi thế trong cài đặt. Ma trận MDS kích thước 8×8 trong mục này chúng tôi cũng đề xuất ở dạng tương tự. Áp dụng các phương pháp đại số đã được nghiên cứu trước đó [6, 8], bằng thực nghiệm nhóm tác giả tìm kiếm và đề xuất ma trận đồng hành:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 4 & 219 & 12 & 20 & 12 & 219 & 4 \end{pmatrix} \quad (3)$$

và nghịch đảo tương ứng của nó:

$$A^{-1} = \begin{pmatrix} 4 & 219 & 12 & 20 & 12 & 219 & 4 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad (4)$$

Khi sử dụng ma trận (3) trong biến đổi D và ma trận nghịch đảo (4) của nó sẽ cho số nhánh bằng 9. Nhóm tác giả cũng tính số điểm bất động của tầng tuyến tính sử dụng ma trận này, kết quả cho thấy hạng của ma trận $M - I$ bằng 8, do vậy sẽ không có điểm bất động nào (ngoại trừ 1 điểm là véc tơ 0 tầm thường).

Để chắc chắn rằng ma trận nhận được từ lũy thừa bậc 8 của ma trận trong biểu thức (3) là MDS, nhóm tác giả sử dụng thuật toán kiểm tra được đề xuất trong [7]. Kết quả kiểm tra bằng thực nghiệm cho thấy đây là một ma trận MDS. Phần tiếp theo, nhóm tác giả sẽ tập trung phân tích khả năng cài đặt của ma trận đề xuất này.

IV. Thực nghiệm cài đặt FLC-SDS-256 với tầng tuyến tính được đề xuất

A. Cài đặt phần mềm

Phân tích giải pháp cài đặt phần mềm: Nhận thấy rằng với mã khối FLC-SDS-256, hai biến đổi S và D trong hàm vòng hoạt động tương tự như các biến đổi SubByte và MixColumn trong hàm vòng của AES [1, 8]. Do vậy, cài đặt sử dụng kỹ thuật tra bảng như của AES là hoàn toàn có thể áp dụng lên mã khối mà nhóm tác giả đang xem xét, trong đó biến đổi S còn lại được cài đặt tương tự như *SubBytes* ở vòng cuối của AES (phiên bản cài đặt của Gladman [11]). Chính vì thế trong phần này, nhóm tác giả chỉ quan tâm đến kỹ thuật cài đặt không sử dụng bảng tra, hay còn gọi là cài đặt dạng *bitslice*, để sử dụng trong các môi trường với tài nguyên hạn chế. Trong phần thực nghiệm, nhóm tác giả vẫn áp dụng cả hai kỹ thuật cài đặt dạng *bitslice* và dạng bảng tra đối với tham số xây dựng được và so sánh với mã khối Kalyna. Thuật ngữ cài đặt dạng *bitslice* được hiểu là phương pháp cài đặt không dùng các bảng tính trước, thay vào đó là thực hiện các phép toán logic trực tiếp trên dữ liệu byte, bit. Phương pháp cài đặt này thường được áp dụng để tích hợp nguyên thủy mật mã trên môi trường với tài nguyên hạn chế. Thuật toán mã khối AES với biến đổi *MixColumns* khi cài đặt trên môi trường thanh ghi 8 bit có dạng cài đặt *bitslice* [1, 8]. Tham số đánh giá trong phép cài đặt này là số phép toán XOR các từ 8 bit và phép Xtime. Phép Xtime là phép nhân với phần tử α hoặc α^{-1} trên trường hữu hạn. Với trường cơ sở $\mathbb{F}_{2^8}$ với đa thức

sinh nguyên thủy $f(x) = x^8 \oplus x^5 \oplus x^3 \oplus x \oplus 1$ được lựa chọn, phép nhân phần tử x với α hoặc α^{-1} được thực hiện như sau:

Giả code nhân với α trên trường F_{2^8}.

```
1. if (x ≫ 1)&0x1 != 0) return ((x ≪ 1) ⊕ 0x2B)&0xFF
2. else return (x ≪ 1)&0xFF
```

Giả code nhân với α^{-1} *trên trường* F_{2^8}.

```
1. if (x&0x1 != 0) return ((x ≫ 1) ⊕ 0x95)&0xFF
2. else return (x ≫ 1)&0xFF
```

Trong FLC-SDS-256 đang xem xét, phép biến đổi D là phép nhân với ma trận tuyến tính và ma trận trạng thái $Y = A^8 \times X$. Bằng phân tích:

$$219 = 223 \oplus 4 = 149 \otimes 149 \oplus 4 = 2^{-1} \otimes 2^{-1} \oplus 2^2$$
$$12 = 8 \oplus 4 = 2^3 \oplus 2^2$$
$$20 = 16 \oplus 4 = 2^4 \oplus 2^2$$

và nhận được đầu ra của D như biểu thức (5).

Để tính được (5) cần $12 \times 8 = 96$ phép XOR hai từ 8 bit và $7 \times 8 = 56$ phép Xtime. Để tính được toàn bộ ma trận trạng thái qua phép biến đổi tuyến tính trong mã pháp FLC-SDS-256 cần 4 lần thực hiện phép D. Do vậy cần $96 \times 4 = 384$ phép XOR và $56 \times 4 = 224$ phép Xtimes. Độ phức tạp cài đặt này cũng chính là độ phức tạp cho cài đặt phép toán trong quá trình giải mã của mã pháp nói trên, vì ma trận tuyến tính của quá trình mã hóa và giải mã có các phần tử giống nhau. Nói cách khác, tốc độ thực thi cho cả quá trình mã hóa và giải mã là giống nhau. Đây là tính chất mà AES, Kalyna không có được bởi dạng ma trận sử dụng trong biến đổi tuyến tính của nó là dạng dịch vòng và không có tính cuộn (tính tự nghịch đảo). Cụ thể trong phép biến đổi tuyến tính ở phép giải mã của AES hay Kalyna, các hệ số của ma trận MDS không thuận tiện cho việc thực hiện nhân nhanh trên trường hữu hạn, do vậy nó tốn tài nguyên cài đặt hơn. Chi tiết về dạng cụ thể ma trận trong biến đổi InvMixColumns của AES hoặc của Kalyna có thể tham khảo [1, 3, 8].

$$\begin{cases} y_0 = x_0 \oplus 2^2\left(x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 \oplus 2(x_3 \oplus x_5) \oplus 2^2 x_4\right) \oplus 2^{-2}(x_2 \oplus x_6) \\ y_1 = x_1 \oplus 2^2\left(x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 \oplus y_0 \oplus 2(x_4 \oplus x_6) \oplus 2^2 x_5\right) \oplus 2^{-2}(x_3 \oplus x_7) \\ y_2 = x_2 \oplus 2^2\left(x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 \oplus y_0 \oplus y_1 \oplus 2(x_5 \oplus x_7) \oplus 2^2 x_6\right) \oplus 2^{-2}(x_4 \oplus y_0) \\ y_3 = x_3 \oplus 2^2\left(x_4 \oplus x_5 \oplus x_6 \oplus x_7 \oplus y_0 \oplus y_1 \oplus y_2 \oplus 2(x_6 \oplus y_0) \oplus 2^2 x_7\right) \oplus 2^{-2}(x_5 \oplus y_1) \\ y_4 = x_4 \oplus 2^2\left(x_5 \oplus x_6 \oplus x_7 \oplus y_0 \oplus y_1 \oplus y_2 \oplus y_3 \oplus 2(x_7 \oplus y_1) \oplus 2^2 y_0\right) \oplus 2^{-2}(x_6 \oplus y_2) \\ y_5 = x_5 \oplus 2^2\left(x_6 \oplus x_7 \oplus y_0 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 \oplus 2(y_0 \oplus y_2) \oplus 2^2 y_1\right) \oplus 2^{-2}(x_7 \oplus y_3) \\ y_6 = x_6 \oplus 2^2\left(x_7 \oplus y_0 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus 2(y_1 \oplus y_3) \oplus 2^2 y_2\right) \oplus 2^{-2}(y_0 \oplus y_4) \\ y_7 = x_7 \oplus 2^2\left(y_0 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_6 \oplus 2(y_2 \oplus y_4) \oplus 2^2 y_3\right) \oplus 2^{-2}(y_1 \oplus y_5) \end{cases} \quad (5)$$

Một số kết quả thực nghiệm: Nhóm tác giả thực hiện cài đặt mã khối FLC-SDS-256/R và so sánh một số kết quả với thuật toán cùng kích thước khối trên thế giới (nếu có). Các kết quả này đều được chạy trên máy có năng lực tính toán là CPU Intel® Xeon E3-1225 v5 với RAM 8G trên hệ điều hình Windows 10, trình biên dịch Visual Studio 2017 chế độ Release, các mã khối không hề dùng một lệnh Assembler và sử dụng chế độ ECB với các kết quả như sau. Kết quả cài đặt phần mềm sử dụng các bảng tính trước được thể hiện ở Bảng 1. Kết quả cài đặt phần mềm dạng bitslice được thể hiện ở Bảng 2.

Đối với dạng cài đặt bitslice, vì không có thông số cài đặt của Kalyna nên nhóm tác giả thực hiện cài đặt lại. Phương pháp cài đặt là đưa về các phép biến đổi cơ sở là phép XOR và phép Xtime khi đánh giá tầng tuyến tính như cách tiếp cận đã sử dụng ở trên. Kết quả thực nghiệm ở trên thấy rằng mã pháp FLC-SDS-256 có thông số cài đặt tốt hơn cả về tài nguyên lẫn tốc độ thực thi có với Kalyna. Hơn nữa, tham số cài đặt cho quá trình giải mã của Kalyna là kém hơn rất nhiều so với quá trình mã hóa bởi ma trận tuyến tính của nó cho quá trình giải mã không có lợi thế cài đặt. Cụ thể, phép nhân với các hệ số trong ma trận này không có lợi thế khi biểu diễn về dạng

phép toán cơ sở (phép XOR và Xtime). Với tốc độ như vậy của FLC-SDS-256, theo đánh giá của nhóm tác giả là vẫn phù hợp để sử dụng trong một số môi trường có tài nguyên hạn chế.

Như vậy, một số kết quả cài đặt phần mềm chỉ ra rằng, với tham số tầng tuyến tính đề xuất FLC-SDS-256/R có cài đặt phần mềm có thể so sánh được với một số chuẩn mật mã hiện đại trên thế giới.

B. Cài đặt phần cứng

Phân tích giải pháp cài đặt phần cứng: Giải pháp cài đặt phần cứng sử dụng ngôn ngữ verilog được nhóm tác giả áp dụng. Phương pháp cài đặt được thực thi theo biểu thức (5) (như dạng cài đặt bitslice ở mục 4) và kỹ thuật iterative loop structures (non-pipeline). Kỹ thuật *iterative loop* thường được sử dụng để cài đặt các thuật toán có nhiều phép tính toán bởi khả năng tối ưu và tiết kiệm tài nguyên hơn các kỹ thuật cài đặt khác (*loop unrolling, pipeline*). Thuật toán nhóm tác giả đang nghiên cứu có kích thước khối lớn (256 bit) đòi hỏi nhiều phép tính toán cần được xử lý hơn so với các mã khối kích thước 128 bit. Thuật toán được cài đặt trên chip FPGA Kintex-7 (xc7k160tfbg676-3) sử dụng công cụ Vivado 2015. Hơn nữa, nhóm tác giả cũng cài đặt mã khối AES với khóa 256 trên cùng môi trường. Cần lưu ý rằng, việc đưa thông số cài đặt của AES vào chỉ mang tính chất tham khảo bởi thuật toán đang xem xét và AES là không cùng lớp. Nhóm tác giả cũng tìm kiếm các kết quả cài đặt thuật toán kích thước khối 256 bit (ví dụ Kalyna) trên thế giới nhưng hiện chưa thấy kết quả cài đặt FPGA nào được công bố. Kết quả cài đặt được đánh giá riêng cho từng quá trình mã hóa, giải mã theo kỹ thuật non-pipeline (bảng 3, bảng 4). Cuối cùng, nhóm tác giả thực hiện so sánh cài đặt thuật toán FLC-SPS-256 trên một số Kit phần cứng thông dụng để khẳng định tính ứng dụng của thuật toán (bảng 5).

Kết quả cho thấy FLC-SDS-256/R có tốc độ thực thi khá cao, tuy nhiên số tài nguyên (LUT) là lớn hơn so với các mã khối có kích cỡ khối 128-bit thông dụng. Điều này là hiển nhiên khi kích thước khối trong FLC-SDS-256/R là gấp đôi. Với số liệu thực nghiệm như vậy, theo quan điểm của nhóm tác giả thì FLC-SDS-256/R vẫn có thể triển khai trên các nền tảng phần cứng trong các ứng dụng thực tế.

BẢNG 1. CÀI ĐẶT PHẦN MỀM SỬ DỤNG CÁC BẢNG TÍNH TRƯỚC

STT	Thuật toán	Kích thước bảng tra (mã hóa + giải mã) KBytes	Tốc độ mã (Mb/s)	Tốc độ giải mã (Mb/s)	Nguồn cài đặt
1	FLC-SDS-256/6	16	1599	1612	Nhóm tác giả
	FLC-SDS-256/7	16	1386	1390	
	FLC-SDS-256/8	16	1225	1226	
2	Kalyna-256/256	16	1616	n/a	Oliynykov [12]
	Kalyna-256/512	16	1299	n/a	

BẢNG 2. CÀI ĐẶT PHẦN MỀM SỬ DỤNG BITSLICE

STT	Thuật toán	Số phép XOR-Xtime của tầng tuyến tính				Tốc độ mã (Mb/s)	Tốc độ giải mã (Mb/s)
		Cho 1 vòng (mã hóa)	Cho full vòng (mã hóa)	Cho 1 vòng (giải mã)	Cho full vòng (giải mã)	Mã	Giải mã
1	FLC-SDS-256/6	384/224	2304/1344	384/224	2304/1344	186	189
	FLC-SDS-256/7	384/224	2668/1568	384/224	2668/1568	153	155
	FLC-SDS-256/8	384/224	3072/1792	384/224	3072/1792	142	142
2	Kalyna-256/256 (14 vòng)	352/192	4928/2668	896/768	12544/10752	144	26
	Kalyna-256/512 (18 vòng)	352/192	6336/3456	896/768	16128/13824	113	20

BẢNG 3. CÀI ĐẶT QUÁ TRÌNH MÃ HÓA THEO KỸ THUẬT NON-PIPELINE

Cài đặt	Thiết bị	Số LUT	Số FF	Số BRAM	Tần số (MHz)	Số Clock	Tốc độ (Mbits/s)
Cài đặt của nhóm tác giả							
FLC-SDS-256/6	Kintex-7 xc7k160tfbg676-3	3964	1565	39	235	18	3342
FLC-SDS-256/7	Kintex-7 xc7k160tfbg676-3	4036	1565	39	235	21	2865
FLC-SDS-256/8	Kintex-7 xc7k160tfbg676-3	4044	1565	39	235	24	2506
AES-256	Kintex-7 xc7k160tfbg676-3	1226	2591	12	300	30	1280

BẢNG 4. CÀI ĐẶT QUÁ TRÌNH GIẢI MÃ THEO KỸ THUẬT NON-PIPELINE

Cài đặt	Thiết bị	Số LUT	Số FF	Số BRAM	Tần số (MHz)	Số Clock	Tốc độ (Mbits/s)
Cài đặt của nhóm tác giả							
FLC-SDS-256/6	Kintex-7 xc7k160tfbg676-3	4346	1567	39	215	18	3058
FLC-SDS-256/7	Kintex-7 xc7k160tfbg676-3	4471	1567	39	215	21	2620
FLC-SDS-256/8	Kintex-7 xc7k160tfbg676-3	4597	1567	39	215	24	2293

BẢNG 5. SO SÁNH CÀI ĐẶT FLC-SDS-256 TRÊN MỘT SỐ KIT THÔNG DỤNG

Cài đặt	Board	Số LUT	Số FF	Số BRAM
FLC-SDS-256/R	ZedBoard Zynq board (xc7z020clg484-1)	8%	2%	28%
	Artix-7 AC701 board (xc7a200tfbg676-2)	3%	1%	11%
	Kintex-7 KC705 board (xc7k325tffg900-2)	2%	1%	9%
	Virtex-7 VC709 board (xc7vx690tffg1761-2)	1%	1%	3%
	ZYNQ-7 ZC706 Board (xc7z045ffg900-2)	2%	1%	7%
	ZedBoard Zynq board (xc7z020clg484-1)	8%	2%	28%
	Artix-7 AC701 board (xc7a200tfbg676-2)	3%	1%	11%

V. KẾT LUẬN

Trong bài báo này, nhóm tác giả đã đánh giá khả năng ứng dụng thực tế của mã khối kích thước lớn (256 bit), đó là mã pháp FLC-SDS-256/R có cấu trúc FLC và có hàm vòng dạng SDS. Kết quả thực nghiệm chỉ ra rằng, việc tăng kích thước khối dữ liệu nhằm đảm bảo kháng một số tấn công tiềm năng sẽ kéo theo nhiều vấn đề về khả năng cài đặt của thuật toán. Để giải quyết vấn đề này, trong trường hợp của FLC-SDS-256/R, nhóm tác giả đề xuất thành phần biến đổi tuyến tính cụ thể (ma trận MDS 8×8) có cấu trúc truy hồi, đảm bảo giảm thiểu tài nguyên cài đặt, đã đưa ra một dạng biểu diễn đơn giản cho phép cài đặt hiệu quả và giảm thiểu tài nguyên cho kiểu cài đặt bitslice của thuật toán FLC-SDS-256/R. Qua quá trình thực nghiệm, nhóm tác giả nhận thấy rằng, tài nguyên và tốc độ nhận được của thuật toán vẫn có thể đáp ứng sử dụng trong các môi trường thực tế. Đối với cài đặt phần mềm sử dụng phương pháp tính trước các bảng tính trung gian, thuật toán FLC-SDS-256/R cho tốc độ thực thi có thể so sánh với thuật toán cùng loại (như Kalyna) với bộ nhớ lưu trữ các bảng tính là tương đương.

Hướng nghiên cứu tiếp theo của nhóm tác giả là tối ưu cài đặt thành phần phi tuyến (tầng S-box) ở dạng bitslice để giảm tối đa tài nguyên và tăng tốc độ thực thi trong cài đặt sử dụng ngôn ngữ phần cứng. Sẽ xem xét cài đặt phần cứng các thuật toán cùng loại để có những so sánh và đánh giá khách quan hơn.

TÀI LIỆU THAM KHẢO

[1]. Зензин, О. and М. Иванов, *Стардарт криптографической защиты-AES. Конечные поля*. КУДИЦ-ОБРАЗ, 2002.

[2]. Oliynykov, R., et al., *A new encryption standard of Ukraine: The Kalyna block cipher*. 2015. Online - https://eprint.iacr.org/2015/650.pdf.

[3]. Sovyn, Y. and V. Khoma, ПРОГРАМНА BITSLICED-ІМПЛЕМЕНТАЦІЯ ШИФРУ «КАЛИНА» ОРІЄНТОВАНА НА ВИКОРИСТАННЯ SIMD-ІНСТРУКЦІЙ МІКРОПРОЦЕСОРІВ З АРХІТЕКТУРОЮ X86-64 *(Software bitsliced implementation of kalyna cipher is oriented to use SIMD instructions for microprocessors with x86-64 architecture)*. Cybersecurity: Education, Science, Technique , 2020, № 7, p. 131-152, https://doi.org/10.28925/2663-4023.2020.7.131152.

[4]. Li, Y. and M. Wang. *On the construction of lightweight circulant involutory MDS matrices*. in International Conference on Fast Software Encryption. pp 121–139, 2016, Springer.

[5]. Pehlivanoğlu, M.K., et al., Generalisation of Hadamard matrix to generate involutory MDS matrices for lightweight cryptography. IET Information Security, 2018. 12(4): p. 348-355.

[6]. Augot, D. and M. Finiasz. *Direct construction of recursive MDS diffusion layers using shortened BCH codes*. in Fast Software Encryption. pp 3–17, 2014, Springer.

[7]. Gupta, K.C. and I.G. Ray, *On constructions of MDS matrices from companion matrices for lightweight cryptography*, in Security Engineering and Intelligence Informatics. 2013, Springer. p. 29-43.

[8]. Daemen, J. and V. Rijmen, The design of Rijndael: AES-the advanced encryption standard. 2002: Springer.

[9]. ГОСТ Р 34.12-2015: Криптографическая защита информации. Блочные шифры. 2015.

[10]. Z'aba, M.R., *Analysis of linear relationships in block ciphers*. Luận án tiến sĩ của Queensland University of Technology, 2010.

[11]. http://ccgi.gladman.plus.com/oldsite/AES/index.php.

[12]. https://github.com/Roman-Oliynykov/ciphers-speed.

[13]. Luong, T. T. (2022). *Building the dynamic diffusion layer for SPN block ciphers based on direct exponent and scalar multiplication*. Journal of Science and Technology on Information Security, 1(15), 38-45.

SƠ LƯỢC VỀ TÁC GIẢ

Trần Sỹ Nam

Đơn vị công tác: Viện Khoa học - Công nghệ mật mã, Ban Cơ yếu Chính phủ.

Email: transynam1989@gmail.com

Quá trình đào tạo: Tốt nghiệp chuyên ngành An toàn thông tin các hệ thống viễn thông tại Học viện FSO – Liên Bang Nga vào năm 2013; Tốt nghiệp Thạc sĩ chuyên ngành Kỹ thuật mật mã tại Học viện Kỹ thuật mật mã vào năm 2018.

Hướng nghiên cứu hiện nay: Nghiên cứu cài đặt hiệu quả các thuật toán mật mã; mã hóa dữ liệu lưu trữ; bảo mật mạng.

Nguyễn Văn Long

Đơn vị công tác: Viện Khoa học - Công nghệ mật mã, Ban Cơ yếu Chính phủ.

Email: longnv@bcy.gov.vn

Quá trình đào tạo: Tốt nghiệp chuyên ngành An toàn thông tin các hệ thống viễn thông vào năm 2008 và Tiến sĩ chuyên ngành Các phương pháp bảo vệ thông tin tại Học viện FSO – Liên Bang Nga vào năm 2015.

Hướng nghiên cứu hiện nay: Nghiên cứu, thiết kế các thuật toán mã đối xứng an toàn, hiệu quả trong cài đặt.

Nguyễn Bùi Cương

Đơn vị công tác: Viện Khoa học - Công nghệ mật mã, Ban Cơ yếu Chính phủ.

Email: nguyenbuicuong@gmail.com

Quá trình đào tạo: Tốt nghiệp chuyên ngành Toán học tại Đại học Sư phạm Hà Nội – Đại học Quốc gia Hà Nội vào năm 2004; Tốt nghiệp Thạc sĩ Toán học tại Đại học Khoa học Tự nhiên – Đại học Quốc gia Hà Nội vào năm 2008; Tiến sĩ Toán học tại Viện Khoa học và Công nghệ Quân sự vào năm 2018.

Hướng nghiên cứu hiện nay: Khoa học mật mã; mã hóa đối xứng.