XÂY DỰNG THUẬT TOÁN DẤU TIN MẬT TRONG ẢNH SỐ

Lê Hải Triều [*], Hồ Văn Canh [+]
* Viện Kỹ thuật điện tử và cơ khí nghiệp vụ, Bộ Công an
[+]Cục Kỹ thuật nghiệp vụ, Bộ Công an

***Abstract*—Kỹ thuật giấu tin (còn gọi là bảo mật tin) trong trong ảnh số yêu cầu cần thiết đối với sự phát triển của kỹ thuật mật mã. Hiện nay có có 2 hướng chính giấu tin là kỹ thuật giấu tin mật (steganography) và kỹ thuật thủy vân số (watermark).**

Trong bài báo này các tác giả tập trung tìm hiểu về kỹ thuật giấu tin mật trong ảnh kỹ thuật số dạng bitmap. Các tác giả giới thiệu thuật toán giấu tin đã được công bố, thuật toán cải tiến của nó và từ đó đề xuất 1 thuật toán giấu tin mật khác có hiệu quả cao hơn.

***Keywords*—giấu tin, ảnh số, steganography, watermark, LBS, BMP.**

I. ĐẶT VẤN ĐỀ

A. Nguyên lý của bảo mật tin trong các ảnh bitmap

Có nhiều thuật toán giấu các thông tin ẩn vào file ảnh. Nhưng phổ biến nhất hiện nay đang được ứng dụng rộng rãi trên thế giới là thuật toán chèn các thông tin ẩn vào các bit có ý nghĩa thấp(Least Significant Bit - LSB) trong phần dữ liệu ảnh của ảnh bitmap 24 bit màu, do việc thay đổi các bit LSB chỉ gây ra sự thay đổi rất nhỏ của các thành phần màu mà mắt thường khó có thể nhận biết đượcsự thay đổi đó. Hiện này người ta thấy rằng không chỉ những bit LSB mà cả những bit Most LSB(Với M=1,2) của phần dữ liệu ảnh bimap cũng không làm thay đổi đáng kể mà mắt thường khó phân biệt được sự thay đổi đó. Tuy nhiên việc phát hiện ảnh có chứa thông tin ẩn bằng thuật toán thống kê cấp 1 hoặc cấp 2 lại tỏ ra rất hiệu quả.Do đó chúng ta cần phải lưu ý đến vấn đề tiếp theo dưới đây [3].

B. Các tham số cần tính toán khi áp dụng thuật toán chèn bit LSB

Kích cỡ dữ liệu ẩn: Khi muốn nhúng(ẩn) một văn bản hoặc 1 file dữ liệu số nào đó vào một file ảnh bitmap(BMP) nào đó trước hết chúng ta cần đảm bảo rằng chất lượng và kích cỡ của file ảnh đó không bị thay đổi. Vì vậy độ dài tối đa của thông báo hoặc file dữ liệu ẩn so với độ dài của các LSB của một file dữ liệu ảnh BMP là:

$L_{max} \approx 12,5\% L_{LSB}$.

Trong đó L_{max} là độ dài tối đa của dữ liệu ẩn và L_{LSB} là độ dài các LSB của một file dữ liệu ảnh BMP. Nếu tính tất cả các bit của 1 file dữ liệu ảnh BMP thì độ dài $L_{max} \approx 100\%$ L^{-}_{BMP} (không vượt quá 100% dữ liệu ảnh của ảnh BMP). Xác định vị trí dữ liệu ẩn: Mỗi khi muốn đặt các bit thông tin ẩn vào 1 file ảnh BMP thì vấn đề đầu tiên là phải xem đặt thông tin ẩn bắt đầu từ vị trí nào của file ảnh là tốt nhất.

Để tăng độ bảo mật cho dữ liệu ản thì dữ liệu ẩn này nên được bắt đầu chèn vào phần dữ liệu ảnh tại một vị trí ngẫu nhiên liên quan đến mật khẩu:

$f(x) = f(C_1,C_2,..C_n)$, trong đó $(C_1,C_2,..,Cn)$ là một dãy con của dãy ký tự của mật khẩu độ dài n.

Thông thường người ta mã hóa bản tin trước khi nhúng vào ảnh số. Việc mã hóa này nhằm đảm bảo độ an toàn cao hơn cho bản tin cần giấu, đặc biệt đối với những thông tin liên quan đến an ninh - quốc phòng v.v... Khi đó cho dù đối phương có thể phát hiện được bản tin giấu thì vẫn còn một lớp mã hoá bảo vệ nó [9].

II. PHÂN TÍCH KHẢ NĂNG GIẤU TIN TRONG ẢNH BITMAP

A. Đánh giá khả năng giấu tin trong ảnh

Kết quả thực nghiệm cho thấy rằng: Việc giấu tin trong ảnh đen trắng đem lại hiệu quả thấp vì việc biến đổi một điểm ảnh từ đen (0) sang trắng (1) hoặc ngược lại từ trắng sang đen rất dễ tạo ra nhiễu của ảnh và do đó người ta dễ phát hiện được bằng thị giác của con người. Hơn nữa, tỷ lệ giấu trin trong ảnh đen trắng rất thấp. Chẳng hạn, một bức ảnh đen trắng kích cỡ 300x300 pixels chỉ có 2KB. Trong khi đó một ảnh 24 màu với kích cỡ tương tự có thể giấu được tới 200KB. Hơn nữa, ảnh đen trắng hiện nay rất ít được sử dụng thay vào đó là ảnh màu hoặc ảnh đa cấp xám. Để chọn ảnh màu ảnh đa cấp xám làm ảnh môi trường cho việc giấu tin. Chúng ta cần quan tâm đến các bit có ý nghĩa thấp nhấtmà ta sẽ ký hiệu LSB. LSB là bit có ít ảnh hưởng nhất đến việc quyết định màu sắc của mỗi một điểm ảnh. Do vậy, khi LSB bị thay đổi thì màu sắc của

ảnh đó sau khi thay đổi không khác nhau đáng kể so với màu sắc của ảnh ban đầu.

Nhưng làm thế nào để xác định được LSB của mỗi điểm ảnh? Việc xác định LSB của mỗi điểm ảnh trong một bức ảnh phụ thuộc vào định dạng của ảnh đó và số bit màu dành cho mỗi điểm ảnh đó.

Đối với ảnh 16 bit màu hoặc 24 bit màu thì việc xác định LSB tương đối đơn giản. Tuy nhiên đối với ảnh 8bit màu trở xuống. Những ảnh này có sử dụng bảng màu (palette màu) thì công việc trở lên rất phức tạp. Riêng ảnh đa cấp xám thì bảng màu của nó đã được sắp, trong đó những cặp màu trong bảng màu có chỉ số chênh lệch càng ít càng giống nhau. Vì vậy, đối với ảnh đa cấp xám LSB của mỗi điểm ảnh (pixel) là bit cuối cùng của điểm ảnh đó.

Quá trình tách LSB của các điểm ảnh đa cấp xám để tạo thành ảnh thứ cấp các bit này bằng thuật toán như thuật toán giấu tin trong ảnh đen trắng sẽ làm cho chỉ số màu của mỗi điểm màu thay đổi tăng hoặc giảm đi một đơn vị. Do đó điểm ảnh mới sẽ có độ sáng tối của ô màu liền trước hoặc sau ô màu của điểm ảnh của điểm ảnh môi trường (ảnh gốc). Bằng mắt thường người ta khó lòng phát hiện được sự thay đổi này. Thực nghiệm chỉ ra rằng, ngay cả khi ta đảo toàn bộ LSB của tất cả điểm dữ liệu ảnh trong một ảnh 8 bit đa cấp xám thì cũng không gây ra sự khác nhau nhiều. Vì vậy, nếu trong mỗi khối ảnh ta chỉ thay đổi nhiều nhất là 2 điểm ảnh thì khả năng phân biệt ảnh gốc và ảnh kết quả là rất khó khăn nếu không nói là "Không thể" bằng mắt thường [6,7].

a. Đối với ảnh số 8 bit màu

Những ảnh thuộc loại này gồm ảnh 16 màu (4 bit màu) và ảnh 256 màu(8 bit màu). Khác với ảnh đa cấp xám ảnh màu với số bit màu bé hơn hoặc bằng 8 không phải luôn luôn được sắp xếp bảng màu. Những màu ở liền kề nhau có thể rất khác nhau. Chẳnghạn, màu đen và màu trắng có thể được sắp xếp kề nhau trong bảng màu. Do đó việc xác định LSB là rất khó khăn. Nếu ta làm như đối với ảnh đa cấp xám, tức là vẫn lấy bit cuối cùng của mỗi điểm ảnh để tạo thành ảnh thứ cấp thì mỗi thay đổi 0 sang 1 hoặc 1 sang 0 trên ảnh thứ cấp thì có thể làm cho màu của ảnh môi trường và màu tương ứng của ảnh kết quả sẽ khác nhau rất xa đến mức mắt thường có thể phân biệt được, dù rằng chỉ số màu của chúng cũng chỉ tăng giảm đi 1 bit mà thôi.

Nhưng làm thế nào để biết được màu nào đã được dùng màu nào không được dùng đến? Để trả lời câu hỏi này trước hết ta phải duyệt toàn bộ các màu trong bảng màu và đánh dấu những màu có chỉ số xuất hiện trong dữ liệu ảnh đó là những màu đã được dùng. Giả sử có một màu C không dùng đến. Với mỗi điểm màu A khi tìm được màu B có sử dụng trong bảng màu để sắp cạnh A mà giá trị S(A,B) vẫn còn lớn hơn một ngưỡng nào đó thì ta sẽ chèn ô màu C vào giữa A và B đồng thời đổi lại màu của ô C sao cho giống màu A và B nhất có thể.

Trường hợp số màu được sử dụng bé hơn hoặc bàng 8 (đối với ảnh 256) hay bé hơn hoặc bằng 4 (đối với ảnh 16 màu) thì việc sắp xếp lại bảng màu theo thuật toán trên cho ta kết quả giấu tin rất tốt.

b. Đối với ảnh 16 bit màu

Ảnh 16 bit màu trong thực tế chỉ sử dụng 15 bit cho mỗi điểm ảnh trong đó 5 bit biển diễn cường độ tương đối của màu đỏ(Red); 5 bit biểu diễn cường độ tương đối của màu xanh lam (Green) và 5 bit biểu diễn cường độ tương đối của màu xanh lơ(blue). Một bit còn lại không được dùng đến đó là bit cao nhất của byte thứ hai trong mỗi cặp 2 byte biểu diễn một điểm ảnh. Đó chính là LSB của ảnh 16 bit màu. Tuy nhiên ta chỉ lấy những bit này để tạo thành ảnh thứ cấp thì lượng thông tin giấu được sẽ không nhiều. Để tăng tỷ lệ tin giấu đối với ảnh 16 bit màu, chúng ta có thể lấy được nhiều hơn 1 bit của mỗi điểm ảnh.

c. Đối với ảnh 24 bit màu

Ảnh 24 bit màu sử dụng 3 byte cho mỗi điểm ảnh, trong đó, mỗi byte biểu diễn một thành phần trong cấu trúc RGB. Trong mỗi byte, các bit càng thấp càng ít ảnh hướng tới màu sắc của mỗi điểm ảnh. Vì vậy đối với ảnh true color, 3 bit cuối cùng của 3 byte của mỗi điểm ảnh chính là LSB của điểm ảnh đó. Bằng kết quả thực nghiệm cho thấy: Việc thay đổi toàn bộ các bit cuối cùng của mỗi byte trong phần dữ liệu ảnh true color cũng không ảnh hưởng có ý nghĩa đến ảnh gốc(ảnh môi trường). Khi đó, nếu thay thế toàn bộ các bit này bằng các bit của dữ liệu ẩn thì tỷ lệ thông tin giấu được sẽ là 12,5% (hoặc 100% so với LSB của dữ liệu ảnh) [5,11].

B. Đánh giá chung

Một giá trị màu thông thường là một véc-tơ 3 thành phần trong không gian màu (tập các màu có thể) RGB [2]. Vì các màu đỏ, xanh lá cây, xanh nhạt là những màu nguyên thủy (primary – màu gốc). Mỗi màu được chỉ ra như là tổ hợp tuyến tính của các màu nguyên thủy đó. Như vậy một véc-tơ trong không gian RGB mô tả cường độ của các thành phần R, G, B đó.

Một không gian khác cũng được biết đến là Y, C_b, C_r. Nó phân biệt giữa độ sáng Y và 2 thành phần sáng tươi (C_b, C_r). Ở đây Y là thành phần sáng (chrominance) của một màu, còn C_b, $C_r$thì phân biệt mức độ màu. Một véc-tơ màu trong không gian màu RGB có thể được chuyển đổi thành Y, C_b, C_rbởi hệ thức sau đây:

$Y = 0.299R + 0.586G + 0.114B$

$C_b = 0.5 + \frac{1}{2}(B - Y)$

$C_r = 0.5 + \frac{1}{1,6}(R - Y)$

Còn mức xám G = 0.299R + 0.587R + 0.114B.

Do trong ảnh đa cấp xám bảng màu đã được sắp và với mỗi điểm ảnh thì bit cuối cùng là LSB của điểm ảnh (gồm 8 bít) đó. Cho nên chúng ta dễ dàng thực hiện việc giấu tin.

Do đó, trong phần tiếp theo tác giả chỉ đề cập đến ảnh 24 bit màu.

III. THUẬT TOÁN GIẤU TIN VÀ THUẬT TOÁN CẢI TIẾN

A. Thuật toán giấu tin phổ biến

a. Các tham số đầu vào:

Các ký hiệu: Gọi m là bức thông điệp cần giấu sau khi chuyển sang dãy bít bởi bộ mã ASCII mở rộng, ta được

* $m = m_1m_2 ...m_{l(m)}$với $m_i \in \{0,1\}$; i= 1,2,...,l(m) và l(m) là độ dài số bít biểu diễn của m.

* $C = C_1C_2...C_{l(c)}$ với $C_i \in \{0, 1\}$; i= 1,2,...,l(c), là ảnh được dùng để giấu thông điệp m.

* $S = S_1S_2...S_{l(c)}$ là ảnh Stego đã được giấu thông điệp m.

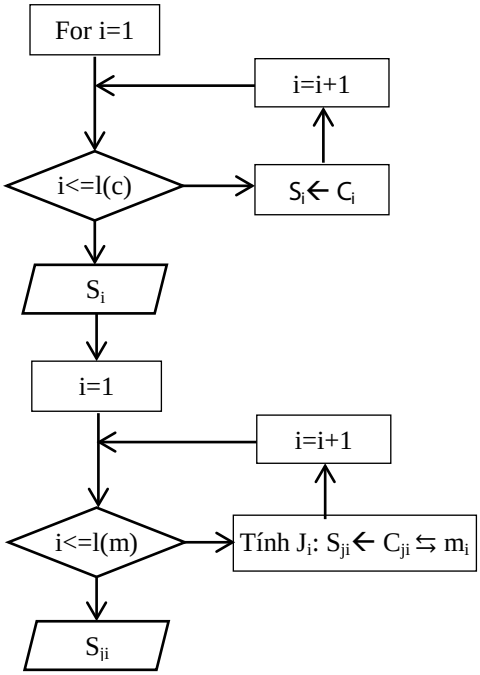
* $S = S_1S_2...S_{l(c)}$ là ảnh đã giấu tin.

b. Thuật toán giấu:

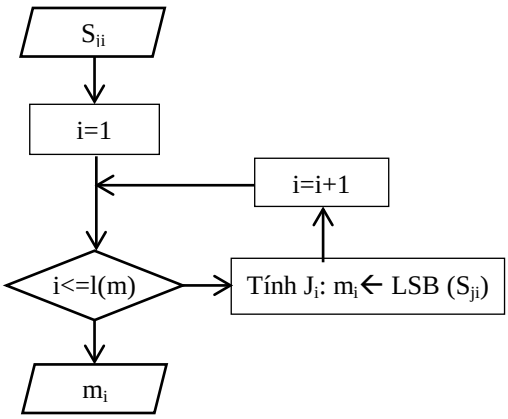
Đầu vào: m, c

Đầu ra: S

Quá trình thực hiện được trình bày trong lưu đồ sau:



c. Thuật toán trích chọn:



d. Đánh giá, nhận xét

Thuật toán này khá đơn giản. Tuy nhiên trong thực tế độ dài l(m) của bản tin thường bé hơn độ dài l(c) của ảnh môi trường, hơn nữa việc giấu tin lại tuần tự nên kẻ tấn công lợi dụng các nhược điểm này để có thể phát hiện được ảnh có giấu dữ liệu bên trong đó hay không bằng phân tích thống kê cấp 2.

B. Thuật toán cải tiến bằng phương pháp "khoảng ngẫu nhiên":

Để khắc phục nhược điểm đó người ta đã đưa ra thuật toán cái tiến được gọi là "Phương pháp khoảng ngẫu nhiên".Nội dung phương pháp như sau:

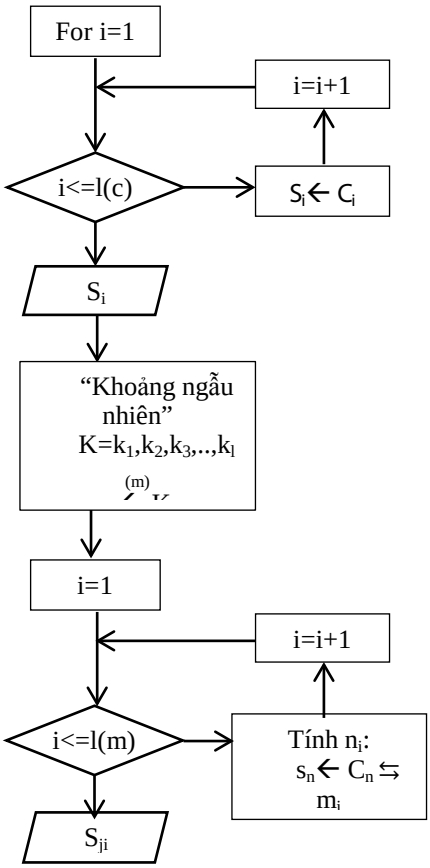
Giả sử hai người A và B trước lúc liên lạc với nhau, thống nhất dùng một khóa K, được gọi là mầm khóa (key seed). Từ mầm khóa K, người ta thống nhất sinh ra một dãy giả ngẫu nhiên (pseudo-random sequence)$k_1,k_2,k_3,..,k_{l(m)}$với (l(m) là độ dài bản thông báo m, quy đổi ra bit) và đặt:

$N_1 = K_1$

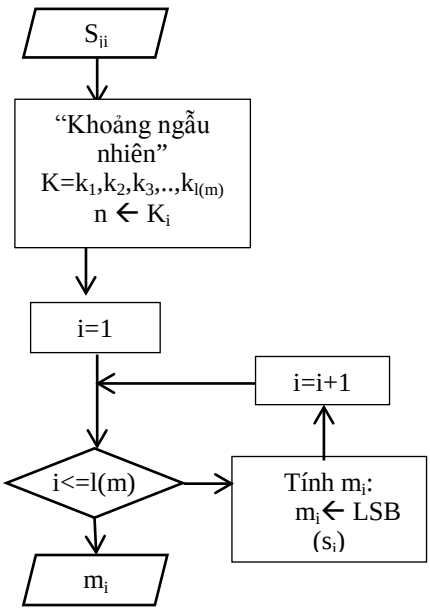
$N_i = N_{i-1} + K_i$ $i \geq 2$ tham gia vào việc truyền thông tin.

Như vậy, khoảng cách giữa 2 bit cần giấu được xác định một cách ngẫu nhiên. Từ đó, thuật toán cải tiến được thực hiện như sau đây.

a. Thuật toán giấu:



b. Thuật toán trích chọn.



c. Đánh giá, nhận xét.

Các thuật toán đã được trình bày ở trên cũng như nhiều thuật toán giấu tin khác đã được công bố đều khó có thể chống lại được các phương pháp phát hiện bằng thuật toán thống kê cấp 1 hoặc cấp 2 nếu số các LSB của dữ liệu ảnh bị thay đổi trên 30% so với tổng các LSB của dữ liệu ảnh [1][10].

Nhưng nếu vậy thì lượng thông tin giấu được vào một ảnh lại không đủ lớn khi kích cỡ ảnh nhỏ. Câu hỏi đặt ra ở đây là: có hay không một thuật toán giấu tin mật sao cho số lượng các LSB của ảnh môi trường bị thay đổi ít nhưng lượng thông tin giấu được nhiều hay không?

IV. ĐỀ XUẤT THUẬT TOÁN MỚI DỰA TRÊN MÃ HOÁ KHỐI

Qua phần III, thực tế cho thấy người ta không thể đồng thời cực tiểu hóa yêu cầu thứ nhất (giảm thiểu lượng LSB của dữ liệu ảnh số bị thay đổi) và cực đại hóa yêu cầu thứ 2 (tăng tối đa lượng bản tin giấu được vào ảnh số).

Tuy nhiên, ta có thể giảm tỷ lệ giấu xuống mức chống lại các tấn công bằng các thuật toán thống kê cấp 1 hoặc cấp 2 mà thông tin giấu được lại khá lớn.

Đó chính là Thuật toán mới được đề xuất trong bài báo này.

Để giấu được nhiều thông tin vào 1 ảnh bitmap mà không làm thay đổi đáng kể đến các LSB của dữ liệu ảnh và đảm bảo bí mật tác giả bổ sung thêm một lớp mã cho thông tin đó, tức làm cho tỷ lệ giấu tin đủ nhỏ mà lượng thông tin giấu được đủ lớn. Ở đây,chúng tôi xây dựng một bộ mã mới, bộ mã chỉ 5 bit chứ không phải là 8 bit như bộ mã ASCII (IV.C).

A. Một số kiến thức toán học bổ trợ.

Ta ký hiệu GF(q)[x] là tập hợp tất cả đa thức cấp n tùy ý $p(x) = a_0 + a_1x + a_2x + ... + x^n$ với $a_i \in GF(q)$ i = 0, 1, ..., n-1 còn q là số nguyên tố.

Ta có các định nghĩa sau đây:

- *Định nghĩa 1:* Đa thức $f(x) \in GF(q)[x]$ được gọi là bất khả qui (irreducible) trong trường GF(q) nếu f(x) không thể phân tích được thành tích các đa thức cấp nhỏ hơn cấp của f(x) trong trường GF(q).

- *Ví dụ 1:* Đa thức $f(x) = x^2 + x + 1$ là đa thức bất khả quy trong trường GF(2).

- *Định nghĩa 2:* Đa thức nguyên thủy (primitive polynomials). Một đa thức bất khả quy $p(x) \in GF(p)[x]$ có cấp m được gọi là đa thức nguyên thủy nếu số nguyên dương bé nhất n mà $x^n - 1$ chia hết cho p(x) là $n = p^m - 1$.

- *Ví dụ 2:* Đa thức $p(x) = x^3 + x + 1$ là đa thức nguyên thủy trong trường GF(2) vì nó là đa thức bất khả qui và số nguyên dương n bé nhất mà $2^n - 1$ chia hết cho $x^3 + x + 1$ là $n = 2^3 - 1 = 7$.

Thật vậy, $x^7 - 1 = (x^3 + x + 1)(x^4 + x^2 + x - 1)$ và không có một $n' < 7$ mà $x^{n'} - 1$ chia hết cho $x^3 + x + 1$.

Ta có các định lý sau đây:

- *Định lý 1:* Có $\phi(2^n - 1)/n$ đa thức nguyên thủy cấp n trong trường GF(2). Điều này được chứng minh trong [8].

- *Định lý 2:* Mọi nghiệm $\{\alpha_j\}$ của một đa thức nguyên thủy cấp m trong trường GF(p)[x] đều có cấp p^m-1. Điều này được chứng minh trong [7,8].

B. Bộ mã Hamming:

Người ta đã chứng minh được rằng [8,9], một bộ mã Hamming trên trường GF(2) thỏa mãn các điều kiện:

+ Độ dài $n = 2^m - 1$
+ Số các ký hiệu mang thông tin là $k = 2^m - m - 1$
+ Số các ký hiệu kiểm tra chẵn lẻ là m = m = k

Khi đó, khả năng sửa sai của bộ mã là t = 1.

C. Xây dựng bộ mã cho 26 ký tự La tinh (a, b, c,...,z)

Vận dụng một số kết quả ở trên, ta sẽ xây dựng bộ mã 26 chữ cái La tinh như sau: Giả sử $p(x) \in GF(2)[x]$ là một đa thức nguyên thủy cấp 5 trên trường GF(2). Lúc đó, ta biết rằng [8] sẽ có $\phi(2^5 - 1)/5$ đa thức nguyên thủy có cấp 5. Một trong những đa thức nguyên thủy cấp 5 trong trường GF(2) là $p(x) = x^5 + x^2 + 1$.

Ta gọi α là một nghiệm của p(x), tức là $p(\alpha) = 0$ hay $\alpha^5 + \alpha^2 + 1 = 0$.

Từ đó suy ra: $\alpha^5 = \alpha^2 + 1$ (1)

Trong không gian véc tơ nghiệm của đa thức p(x) có cấp 5, tức là có cực đại là 5 véc tơ độc lập tuyến tính. 5 véc tơ này sẽ tạo thành một cơ sở của không gian nghiệm.

Bằng cách trực chuẩn hóa cơ sở này, ta được một cơ sở của không gian nghiệm của nó là:

α^0 = 1 0 0 0 0
α^1 = 0 1 0 0 0
α^2 = 0 0 1 0 0
α^3 = 0 0 0 1 0
α^4 = 0 0 0 0 1

Từ (1) ta có α^5 = 1 0 1 0 0, tiếp tục $\alpha^6 = \alpha^3 + \alpha = \alpha^3 + \alpha^1$ = 0 0 0 1 0 + 0 1 0 0 0 = 0 1 0 1 0, .v.v.

Cuối cùng ta đã xây dựng bộ mã trong Bảng 1 sau đây: 5-bít

BẢNG 1. BỘ MÃ 5-bít

10000	01011	11000	11010
01000	10001	01100	01101
00100	11100	00110	10010
00010	01110	00011	01001
00001	00111	10101	
10100	10111	11110	
01010	11111	01111	
00101	11011	10011	
10110	11001	11101	

Nếu thêm vectơ 00000 vào bảng trên ta sẽ có bộ mã nhị phân gồm 32 từ mã. Với bộ mã này, ta lập tương ứng với 25 chữ cái Latinh(trừ chữ z) vì z có xác suất xuất hiện rất bé trong các bản tin (tỷ lệ khoảng 0,5%) nên ta sẽ sử dụng từ mã đó vào mục đích khác và sẽ được trình bày ở nội dung sau. Từ Bảng 1, ta tiếp tục xây dựng bảng 2 là bảng mã chữ cái tương ứng với bộ mã của bảng 1.

BẢNG 2. XÂY DỰNG BẢNG MÃ CHỮ CÁI

TT	Kí tự	Từ mã
0	Ông	00000
1	a	10000
2	b	01000
3	c	00100
4	d	00010
5	e	00001

6	f	10100
7	g	01010
8	h	00101
9	i	10110
10	j	01011
11	k	10001
12	l	11100
13	m	01110
14	n	00111
15	o	10111
16	p	11111
17	q	11011
18	r	11001
19	s	11000
20	t	01100
21	u	00110
22	v	00011
23	w	10101
24	x	11110
25	y	01111
26	(khóa mã), .	10011
27	y/c	11101
28	K/g	11010
29	tr/lời	01101
30	Gấp	10010
31	Người nhận	01001

Chú ý: Từ mã "10011" được dùng ở 2 chế độ là báo khóa cho nơi nhận biết trong trường hợp bản thông báo cần mã hóa trước lúc nhúng tin. Nếu không mã hóa thì từ mã này thay vì dấu "."(stop). Để chống lại việc phát hiện từ khóa, mỗi khi cần dùng nó để mã hóa(DES, hoặc AES hoặc bất cứ khóa mã nào) thì qui định nhóm "10011" xuất hiện đầu tiên(hoặc cuối cùng) sẽ là báo khóa và còn lại là dùng vì dấu "."(stop).

Ví dụ: Thông báo:"K/g Ông Lê Văn Thành" (dùng bộ gõ unicode "K/g Ong Lee Vawn Thanhf") thì bộ mã tương ứng là:

11010 00000 11100 00001 00001 00011 10000 10101 00111 01100 00101 10000 00111 00101 10100 10011.

Như vậy nếu viết đầy đủ thì sẽ là: "Kinhs guiwr oong Lee Vawn Thanhf". Riêng việc xây dựng bộ mã như trên đã giảm được 3 lần so với dùng bộ mã ASCII mở rộng (trong ví dụ này) như các thuật toán giấu tin đã được công bố cho đến này [4].

Trước khi xây dựng thuật toán giấu tin mới, ta xây dựng một ma trận H có cấp 5x31. Ở đây, Ma trận H được sử dụng dựa trên cơ sở bộ mã sửa sai Hamming trong thông tin liên lạc số. Ta biết rằng [8]: nếu Bộ mã Hamming độ dài n = 2m - 1, với ký hiệu mang tin là k = 2m - m -1 , số ký hiệu kiểm tra chẵn lẻ là = n - k = m thì khả năng sửa sai sẽ là t = 1. Ý nghĩa của ma trận H chính là ta chỉ làm sai 1 bít (nhúng 1 bít) đối với độ dài từ mã là 5 bít, hầm giảm tỷ lệ nhúng tin xuống nhưng đồng thời tăng được lượng tin giấu nhiều hơn.

D. Đề xuất thuật toán giấu tin mới

a. *Quá trình giấu tin*.

Trên cơ sở kết quả đã được trình bày ở trên, ta xây dựng được thuật toán giấu tin mới như sau:

Đầu vào:

+ Bản bản tin $m=m_1m_{2...}m_{l(m)}$ với $m_i \in \{0,1\}$ $i=1,2,..,l_{(m)}$

+ Ảnh cover $C=C_1C_2,\ldots C_{l(c)}$ với $C_i \in \{0,1\}$; $i=1,2,..,l_{(c)}$

Đầu ra:

+ Ảnh Stego S đã giấu tin, ta ký hiệu S=C(m)

Sau đây là các bước tiến hành:

Bước 1: Mã hóa bản tin m với thuật toán DES với khóa ở bảng 2 và kết quả ta nhận được bản mã $y=E_{DES}(m) = y_1y_2,\ldots,y_{l(m)} y_i \in \{0,1\}$ $i=1,2,..,l_{(m)}$. Nếu không cần mã hóa bản tin để tăng tính bảo mật trước lúc giấu thì ta bỏ qua bước 1 và sang bước 2 luôn.

Bước 2: Tạo ảnh thứ cấp $C_0 = x_{i0}, x_{i0+1},\ldots x_{i0+l(c)}$

$x_i \in \{0,1\}$, $i=i_0,\ldots.,i_{0+l(c)}$ bằng cách quy ước chọn 1 chỉ số i_0 nào đó của pixel dữ liệu ảnh cover C và trích chọn các LSB của các điểm ảnh có hệ số bắt đầu từ $i_0 = 1,2,\ldots l$(người gửi và người nhận thống nhất trước).

Bước 3: Chia C_0 thành từng block, mỗi block gồm 31 bít, tính từ khởi điểm x_{i0}, ta được

$C_0 = C_0(1)\ C_0(2)\ \ldots.\ C_0\ ([\frac{l(c)}{31}])$ $[\frac{l(c)}{31}]$ là phần nguyên

Bước 4: Chia căn bản mã y thành từng khối, mỗi khối 5 bit và được kết quả là:

$$Y = y(1)\ y(2)\ \ldots.\ (y[\frac{l(m)}{5}] + 1) \qquad (2)$$

Bước 5: Với i = 1, 2, ..., [l(m)/5] + 1, thực hiện $Z^T(i) = y^T(i) \oplus HC^T(i)$ (trong đó C^T là véc tơ chuyển vị của véc tơ C).

Bước 6: Với i = 1, 2, ... ,[l(m)] + 1;Tìm trong ma trận H, nếu tồn tại j_0, với j_0 = 1, 2, ..., 31 sao cho $y^T(i)= h_{j0}$ thì ta thực hiện đảo bit của véc tơ $C_0(i)$ tại vị trí j_0: $X'_{j0} = X_{j0} + 1$ và thay X'_{j0} vào vị trí của X_{j0} của véc tơ $C_0(i)$. Sau khi thay X'_{j0} ta có $C_0(i) = X'_0(i)$, với $X_0(i) + 1, \ldots, X_0(i) + 31$.

Nếu không tồn tại j_0 sao cho $y^T(i)= h_{j0}$ thì bỏ qua và quay lại Bước 5.

Bước 7: Ảnh thứ cấp mà ta đã thực hiện trên ký hiệu là C_1.

Bước 8: Trả lại ảnh thức cấp C_1 vào đúng vị trí ban đầu như khi ta trích chọn C_0. Cuối cùng ta nhận được ảnh Stego S.

E. Ví dụ:

Đầu vào: Bản tin m cần giấu "K/g ông X" và ảnh cover C.

Đầu ra: Bản tin m và ảnh C được khôi phục.

a. *Quá trình giấu:*

M="K/g ông X"<-> 11010 00000 11110 = (m1,m2,m3). Giả sử có 3 dãy LSB của ảnh C là(với giả thiết khởi điểm giấu là i_0=1):

$C_0(1)$=010011 00111 01000 11010 11100 10001

$C_0(2)$=100110 10100 01101 10000 10100 11010

$C_0(3)$=101110 10110 00111 10101 01101 10010

Ta có:

$y_1^T=m_1^T \oplus HC_0^T(1)=(11010)^T \oplus HC_0^T(1)=(11010)^T \oplus (11010)^T \oplus (01111)^T=(10101)^T$

Tồn tại y_1^T trùng với cột thứ 23 của ma trận H, ta thực hiện đảo bit của $C_0(1)$ tại vị trí 23 và ta có:

$C_0'(1)$= 010011 00111 01000 11010 10100 10001

$y_2^T=m_2^T \oplus HC_0^T(2)=m_2^T \oplus HC_0^T(2)=(00000)^T \oplus HC_0^T(2)=(00000)^T \oplus (11010)^T \oplus (01010)^T=(01010)^T$

Tiếp tục tồn tại y_7^Tcột thứ 7 của H vậy thành phần thứ 7 của $C_0(2)$ được đảo bit và do đó ta nhận được:

$C_0'(2)$= 100110 00100 01101 10000 10100 11010

Tương tự, vị trí cột 22 của $C_0(3)$ được đảo bít:

$C_0'(3)$=101110 10110 00111 10101 11101 10010

Đó là ảnh thứ cấp của ảnh Stego S đã giấu thông báo M= "K/g ông X". Sau khi trả lại các LSB tương ứng của ảnh Cover C, ta nhận được ảnh Stego S.

b. *Quá trình trích chọn:*

Đầu vào : ảnh Sstego S

Đầu ra: Bản tin M và ảnh C được khôi phục.

Tính $m_i^T = HC_0'(i)$ với i=1,2,3

Ta nhận được 11010 0000 11110↔"Kính gửi ông X".

F. Đánh giá, nhận xét

Thuật toán vừa được trình bày ở trên đơn giản cả cho việc nhúng và trích chọn. Có thể cải tiến thuật toán này để giảm tỷ lệ làm thay đổi ảnh gốc hơn.

Khả năng chịu tấn công bằng các phương pháp thống kê cấp 1 và cấp 2 rất hiệu quả, do tỷ lệ nhúng thấp.

Trong trường hợp ví dụ trên, tỷ lệ nhúng khoảng 3,2% (≈1/31). Chúng tôi tiếp tục cải tiến bảng mã để có thể giảm tỷ lệ nhúng xuống dưới 1,5% và từ đó đưa ra ứng dụng thực tế trong công tác của ngành An ninh.

V. NHẬN XÉT, KẾT LUẬN

Thuật toán giấu tin được đề xuất trong phần IV rất đơn giản và có ưu điểm lượng thông tin giấu được lớn nhưng các LSB được mã thay đổi ít nhất. Đây là tỷ lệ cho phép chống lại các thuật toán tấn công thông kê cấp 1 và cấp 2. Các thuật toán tấn công phát hiệu mù trên LSB của miền không gian, thuật toán phát hiện có ràng buộc [10] và các thuật toán tấn công đã được công bố trong [1]. Nếu tỷ lệ nhúng dưới 2% thì mọi phương pháp dò tìm bằng các thuật toán thống kê đều không có hiệu quả.

Cho đến nay, các tác giả cũng chưa tìm thấy một thuật toán nào có tỷ lệ tin giấu thấp hơn 3% mà lượng thông tin giấu được lại lớn như vậy.

Ngoài ra, trong thuật toán này, ma trận H có thể được mở rộng, chẳng hạn ma trận H có thể có kích cỡ 8 x 255 và như vậy tỷ lệ nhúng (số bit của các pixel bị đảo) còn bé hơn nữa mà vẫn đảm bảo lượng thông tin nhúng là khá lớn (có thể xuống cỡ 0,004).

Trong phạm vi bài này chúng tôi chỉ dừng lại ở kích cỡ ma trận H là 5 x 31 và coi như nó sẽ hài hòa giữa tốc độ tính toán và sự phức tạp của thuật toán.

Hiện nay, chúng tôi đang tiếp tục nghiên cứu cho trường hợp ma trận H với kích cỡ 6x63 để giảm tỷ lệ tin giấu xuống dưới 1,5%.

TÀI LIỆU THAM KHẢO

[1] Chunfang Yang, Xiangyang Luo, Fenlin Liu, "Embedding Ratio Estimating for Each Bit Plane of Image", Springer-Verlag Berlin Heidelberg, 2009.

[2] Foley, J.etal,"Computer Graphic: principles and practice". MA. Addison Wesley, 1990.

[3] Ker, A.D.,"Steganalysis of Embedding in Two Least-Significant Bits", IEEE Transactions on Information Forensics and Security 2,pp. 46-54, 2007.

[4] Lương Viết Nguyên, Nguyễn Thị Thu Thủy, Hồ Văn Canh, "Solving language recognition problems ", Tập san tại Hội nghị về những vấn đề chọn lọc trong công nghệ thông tin – truyền thông, tại trường Đại học Cần Thơ, pp. 171-179, 2011.

[5] M. Wu, E. Tang, and B. Liu (2000), "Data Hiding in Digital Images", IEEF International Conference on Multimedia, Expo (ICME), 2000.

[6] Moller, S. A Pfitzmann and I. Stirand, "Computer Based Stenography: How It Works and Why Therefore Any Restrictions on Cryptography Are Nonsense At Best", In Information Hiding Notes in Computer Science, Springer, pp. 7-21, 1996.

[7] Rongyue Zhang, Vasiliy Sachnev, Hyoung-Joong Kim Fast "BCH Syndrome Coding for Steganography",Lecture Notes in Computer Science, Volume 5806, CIST, Graduate School of Information Management and Security Korea University, Seoul, Korea, pp 48-58, 2009.

[8] Stephen B. Wicker,"Error Control Systems for Digital Communication and Storage", Prentice Hall - New Jersey, 2009.

[9] Stephan Katzenbeisser, Fabien A. P. Petitcolas : " Information Hiding Techniques for Steganography and Digital Watermarking "Artech House Boston - London 2000

[10] Y. Kim, Z. Duric, D. Richards,"Modified matrix encoding technique for minimal distortion steganography", In: Camenisch, J.L., Collberg, C.S.,Johnson, N.F., Sallee, P. (eds.) IH, 2006. (*LNCS, vol. 4437, Springer, Heidelberg, 2007*).

[11] Luận án TS Hồ Thị Hương Thơm, Hà Nội, 2012

[12] Lê Hải Triều, Hồ Văn Canh. "Kỹ thuật giấu tin mật trong truyền ảnh số", Kỷ yếu Hội thảo quốc gia lần thứ XIX: Một số vấn đề chọn lọc của Công nghệ thông tin và truyền thông, Đại học Sư phạm – ĐH Quốc gia Hà Nội, trang 187-193, 10/2016.

Lê Hải Triều sau khi tốt nghiệp Học viện Kỹ thuật Quân sự, anh bắt đầu làm việc tại Cục TC-ĐL-CL, Bộ Quốc Phòng từ năm 1996 – 2001. Hiện nay anh là Trưởng Phòng Kỹ thuật nghiệp vụ, Viện Kỹ thuật Điện tử và Cơ khí nghiệp vụ, Bộ Công an..

Hướng nghiên cứu chủ yếu của anh là ứng dụng công nghệ cao, nghiên cứu, chế tạo các thiết bị khống chế điện thoại di động, các thiết bị liên lạc bản tin hình ảnh và text có bảo mật vô tuyến, thiết bị nhận dang vân tay, các thiết bị thu thập và truyền âm thanh và hình ảnh dạng vô tuyến và hữu tuyến.

Anh có bằng Kỹ sư Vô tuyến Điện tử tại Học viện KTQS và CNTT tại ĐHBK Hà Nội. Năm 2004 anh tốt nghiệp Thạc sỹ Xử lý thông tin và Truyền thông, ĐHBK Hà Nội.

Hồ Văn Canh sinh năm 1944, nhận học vị tiến sỹ Toán-Lý năm 1986. Ông là nghiên cứu viên chính về bảo mật và thám mã. Ông nguyên là Phó Trưởng phòng phụ trách Đơn vị nghiên cứu và phát triển Nghiệp vụ, thuộc Cục Kỹ thuật nghiệp vụ I, Bộ Công an.

Ông tham gia rất nhiều hoạt động khoa học phục vụ lĩnh vực an ninh-quốc phòng; Lĩnh vực nghiên cứu chính của ông là thám mã và an ninh, an toàn và bảo mật thông tin, dữ liệu đa phương tiện, thông tin số.

Ông đã xuất bản nhiều sách, tài liệu tham khảo và tham gia giảng dạy, hướng dẫn sinh viên cao học, nghiên cứu sinh tại các trường đại học như ĐH Công nghệ, ĐH Mật mã, ĐH Hàng hải Hải phòng, ĐH Kỹ thuật – Hậu cần Công an, ĐH Thái Nguyên,v. . Đến nay, ông đã đào tạo được trên 40 Thạc sĩ chuyên ngành ATTT và đã có 32 bài báo khoa học được công bố ở trong và ngoài nước cho đến nay…

BẢNG 3. MA TRẬN H

1	0	0	0	0	1	0	0	1	0	1	1	0	0	1	1	1	1	1	0	0	0	1	1	0	1	1	1	0	1	0
0	1	0	0	0	0	1	0	0	1	0	1	1	0	0	1	1	1	1	1	0	0	0	1	1	0	1	1	1	0	1
0	0	1	0	0	1	0	1	1	0	0	1	1	1	1	1	0	0	0	1	1	0	1	1	1	0	1	0	1	0	0
0	0	0	1	0	0	1	0	1	1	0	0	1	1	1	1	1	0	0	0	1	1	0	1	1	1	0	1	0	1	0
0	0	0	0	1	0	0	1	0	1	1	0	0	1	1	1	1	1	0	0	0	1	1	0	1	1	1	0	1	0	1