TRƯỜNG ĐẠI HỌC LÂM NGHIỆP

KHOA HỆ THỐNG THÔNG TIN

========&&&========



**BÀI THI CUỐI KỲ**

**MÔN : AN TOÀN VÀ BẢO MẬT THÔNG TIN**

**CHỦ ĐỀ : NHÚNG DỮ LIỆU VÀO ẢNH**

**Giáo viên hướng dẫn : Mai Hà An**

**Nhóm thực hiện : Đỗ Thành Công**

**Nguyễn Văn Đức**

**Doàn Quang Khải**

**Vũ Tiến Đạt**

MỤC LỤC

[**LỜI MỞ ĐẦU** 3](#_Toc137563106)

[**GIỚI THIỆU ĐỒ ÁN** 3](#_Toc137563107)

[**LỊCH TRÌNH HOẠT ĐỘNG NHÓM** 4](#_Toc137563108)

[PHƯƠNG THỨC MÃ HÓA 5](#_Toc137563109)

1 Kỹ thuật chèn vào các bit ít quan trọng của một điểm ảnh (Least Significant Bits) 5

[2. Kỹ thuật mã hóa khối kết cấu bề mặt (Texture Block Coding)](#_Toc137563113) 9

[3. Kỹ thuật Patchwork 10](#_Toc137563114)

[4.Một số kỹ thuật khác ..11](#_Toc137563115)

[**CÂU LỆNH VÀ GIẢI THUẬT** 12](#_Toc137563118)

[**TỔNG KẾT** 16](#_Toc137563119)

**LỜI MỞ ĐẦU**

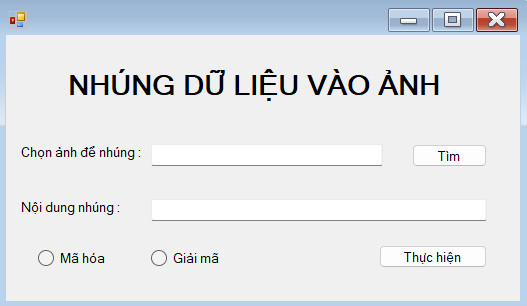
Trong thời đại số hóa ngày nay, vấn đề bảo mật thông tin ngày càng được coi trọng hơn bao giờ hết. Trong các hoạt động truyền thông trong mạng internet, mã hóa và giải mã dữ liệu là một trong những hoạt động quan trọng nhất, nhằm đảm bảo tính bảo mật và sự riêng tư của thông tin truyền tải. Các phương pháp mã hóa và giải mã hiện nay đã trải qua nhiều sự phát triển và cải tiến, từ các phương pháp mã hóa đơn giản sang các phương pháp mã hóa hiện đại sử dụng mật mã đối xứng, mật mã bất đối xứng và các thuật toán mã hóa khối như AES, DES, Blowfish... Ngoài ra, các phương pháp mã hóa thủ tục, như mã hóa mã nguồn đã trở thành phổ biến trong việc bảo mật ứng dụng và các trang web. Tuy nhiên, các phương pháp mã hóa và giải mã vẫn còn nhiều hạn chế. Trong một số trường hợp, các phương pháp mã hóa này vẫn dễ bị xâm nhập và tấn công bởi các hacker và chuyên gia bảo mật. Vì vậy, việc tìm kiếm những phương pháp mã hóa mới và hiệu quả là vô cùng cần thiết. Một trong những phương pháp mã hóa được giấu đi một cách kín đáo và không dễ bị phát hiện. Bài báo cáo này tập trung vào việc xây dựng một giao diện đơn giản để thực hiện mã hóa và giải mã dữ liệu bằng phương pháp giấu tin trong ảnh. Phương pháp này sử dụng kỹ thuật ẩn tin trong hình ảnh (Image steganography) để giấu thông tin vào trong các pixel của ảnh một cách bí mật và không dễ bị phát hiện. Bài báo cáo bao gồm mô tả chi tiết về thuật toán và kỹ thuật sử dụng, kèm theo đó là hướng dẫn cài đặt và sử dụng giao diện đơn giản để thực hiện quá trình mã hóa và giải mã. Ngoài ra, báo cá

Ngoài ra, báo cáo cũng đề cập đến một số ứng dụng của phương pháp giấu tin trong ảnh trong thực tế, giúp người đọc hiểu rõ hơn về tính ứng dụng và tính thực tiễn của kỹ thuật này. Với bài báo cáo này, mong muốn giúp người đọc có thể nắm bắt được cách thức thực hiện mã hóa và giải mã dữ liệu bằng phương pháp giấu tin trong ảnh một cách dễ dàng và hiệu quả, qua đó từ đó có thể ứng dụng vào thực tế để bảo vệ thông tin và tăng tính bảo mật trong các truyền thông trực tuyến.

**GIỚI THIỆU ĐỒ ÁN**

**Đề tài:Nhúng dữ liệu vào ảnh**

**Ngôn ngữ lập trình : C#**



**LỊCH TRÌNH HOẠT ĐỘNG NHÓM**

|  |  |  |  |
| --- | --- | --- | --- |
| Họ và Tên | 13/05 | 20/05 | 10/06 |
| Đỗ Thành Công | Tìm hiểu và làm quen visual code | Tìm hiểu giao diện | Viết code |
| Nguyễn Văn Đức | Tìm hiểu và làm quen visual code | Tìm hiểu phân tích dữ liệu thành bit, đọc tệp | Viết code |
| Đoàn Quang Khải | Tìm hiểu và làm quen visual code | Tìm hiểu nhúng các bit vào ảnh | Viết code |
| Vũ Tiến Đạt | Tìm hiểu và làm quen visual code | Tìm hiểu tách dữ liệu ra khỏi ảnh | Viết code |

# Phương thức mã hóa

### **1.**  **Kỹ thuật chèn vào các bit ít quan trọng của một điểm ảnh (Least Significant Bits)**

Thông thường khi chuyển một bức ảnh tương tự sang định dạng số, người ta thường chọn một trong 3 cách thể hiện màu sau :

- 24 bit màu : mỗi điểm ảnh có giá trị màu là một trong 224 màu, và những màu này là tổ hợp từ 3 màu cơ bản Red, Green, Blue có giá trị từ 0 đến 255.

- 8 bit màu : mỗi điểm ảnh có giá trị màu là một trong 256 màu, được chọn từ bảng màu.

- 8 bit xám : mỗi điểm ảnh có giá trị là một trong 256 cấp độ xám.

Các kỹ thuật giấu tin trong miền không gian ảnh được thực hiện bằng cách thay đổi trực tiếp giá trị màu hoặc cường độ sáng của các điểm ảnh được chọn. Một kỹ thuật giấu tin được sử dụng trong không gian ảnh là kỹ thuật chèn vào các bit ít quan trọng nhất của điểm ảnh. Phương pháp giấu tin LSB thực hiện bằng cách thay đổi các bit quan trọng của mỗi màu trong 24 bit màu hoặc chèn vào các bit ít quan trọng của ảnh 8 bit.

Ví dụ chữ A có mã ASCII là 65 ( thập phân ), có thể biểu diễn nhị phân là : 1000001.

Để giấu chữ A vào trong ảnh, ta cần 3 điểm ảnh liên tiếp trong ảnh 24 bit màu. Giả sử cho giá trị màu của 3 pixel trước khi chèn là :

10000000.10100100.10110101,10110101.11110011.10110111,11100111.10110011.00110011

Sau khi chèn, giá trị của các pixel màu sẽ như sau:

10000001.10100100.10110100.10110100.11110010.10110110.11100110.10110011.00110011 (những giá trị đậm là những giá trị bị thay đổi )

Ví dụ tương tự, cần giấu chữ A vào trong ảnh 8 bit màu, ta cần 8 pixel. Giả sử giá trị của 8 pixel ban đầu như sau :

10000000,10100100,10110101,10110101,11110011,10110111,11100111,10110011 sau khi chèn chữ A, giá trị của 8 điểm ảnh này sẽ như sau :

10000001,10100100,10110100,10110100,11110010,10110110,11100110,10110011 (những giá trị đậm là những giá trị bị thay đổi).

Từ những ví dụ trên, ta có thể suy luận ra rằng phương pháp chèn vào các bit ít quan trọng thường có 50% cơ hội để thay đổi một bit ít quan trọng trong 8 bit, vì thế không tạo ra sự thay đổi lớn đối với ảnh gốc. Hay nói cách khác, ý tưởng cơ bản của kỹ thuật này là thay đổi các bit biểu biễn các giá trị mà khả năng tri giác của mắt người là kém nhất. Ví dụ như thay đổi giá trị các bit biểu diễn thành phần màu xanh hay biểu diễn thành phần độ chói của điểm ảnh, là những giá trị mà với một sự thay đổi nhỏ thì ta khó phát hiện ra sự thay đổi.

Tỉ lệ dữ liệu

Vấn đề cơ bản nhất của kỹ thuật chèn vào các bit ít quan trọng đối với ảnh 24 bit màu là chèn được 3 bit / pixel. Vì vậy mỗi điểm ảnh 24 bit chúng ta có thể giấu 3 bit giấu/ 24 bit dữ liệu, tức 1 bit dữ liệu giấu / 8 bit dữ liệu. Trong trường hợp này, ta giấu được 1 bit trong 8 bit của ảnh chứa.

Chúng ta có thể thay đổi tỉ lệ giấu tin trong mỗi điểm ảnh bằng cách tăng hoặc giảm số bit thông tin giấu trong mỗi điểm. Có thể là chỉ giấu 1 bit thông tin trong 24 bit dữ liệu, hoặc 6 bit hoặc 9 bit, điều này phụ thuộc vào ứng dụng của chúng ta.

Để giấu 1 byte thông tin, tùy theo phương pháp chèn 1 bit, 2 bit, 3 bit mà ta phải cần có số byte dữ liệu chứa khác nhau. Với phương pháp chèn 1 bit, ta cần 8/1\*8 = 8 byte dữ liệu, đối với phương pháp chèn 2 bit ta cần 8/2\*8 = 4 byte và đối với phương pháp chèn 3 bit ta cần 8/ 3\*8 = 21.33 byte.

Tính bền vững

Phương pháp chèn vào các bit ít quan trọng rất dễ bị tấn công bởi các phép biến đổi, thậm chí là những phép thông thường và ít nguy hại nhất.

Các phương pháp nén mất mát thông tin, chẳng hạn như JPEG, gần như là phá vỡ các thông tin bị giấu. Vấn đề là lỗ hỗng của hệ thống tri giác của con người mà phương pháp chèn các bit ít quan trọng cố gắng khai thác – ít nhạy cảm với sự thay đổi nhỏ các giá trị màu của màu xanh hay độ chói - cũng giống với những gì mà các phương pháp nén mất mát thông tin dựa vào để có thể giảm dung lượng bức ảnh.

Các phép biến đổi hình học như di chuyển các điểm ảnh và đặc biệt là đổi chỗ các điểm ảnh so với lưới gốc ban đầu cũng phá hủy các thông tin được nhúng và cách duy nhất để lấy lại thông tin là phải làm các phép chuyển đổi ngược lại.

Các phép biến đổi các trên ảnh cũng dễ dàng làm mất các thông tin bị nhúng. Nói tóm lại, phương pháp chèn vào các bit ít quan trọng là phương pháp giấu tin rất ít bền vững.

Độ dễ của quá trình phát hiện và giải thông tin.

Không có điểm gì là đáng chú ý trong phương pháp chèn vào các bit ít quan trọng, phương pháp này chỉ làm thay đổi một ít độ nhiễu của kết cấu bề mặt bức ảnh. Nếu dùng phương pháp này thì rất dễ dàng để giải các tin giấu và kiểm tra lại thông tin giấu.

Thích hợp cho giấu tin mật hay thủy vân

Trước tiên, do đây là kỹ thuật rất dễ bị tấn công thậm chí là các biến đổi đơn giản, phương pháp chèn vào các bit ít quan trọng dường như không thích hợp đối với thủy vân số, nơi mà bức ảnh phải đối mặt với các thay đổi cố tình làm phá hoại chúng, cộng thêm các biến đổi khác như nén/giải nén (có mất mát thông tin) hoặc chuyển từ tương tự sang số và ngược lại.

Do tính chất tỉ lệ dữ liệu giấu cao nên phương pháp này thích hợp cho việc giấu thông tin mật, nơi mà tính bền vững ít được quan tâm.

Kỹ thuật giấu tin này được sử dụng cho các thuật toán thủy vân chống xuyên tạc do tính chất dễ bị phá vỡ trước các tấn công trên hệ thủy vân.

Vấn đề và các giải pháp

Có thể khẳng định rằng phương pháp chèn vào các bit ít quan trọng rất tốt cho giấu tin mật. Tuy nhiên một điểm yếu của phương pháp này là tính dễ dàng trong quá trình giải tin. Chúng ta không muốn một kẻ tấn công có chủ ý nào có thể đọc được thông tin mà chúng ta gửi đi. Điều này thì được hoàn thiện bởi hai kỹ thuật bổ trợ sau :

- Mã hóa thông điệp trứớc khi nhúng, điều này làm cho những ai sau khi lấy được thông tin thì phải tiến hành giải mã nó thì mới có thể hiểu được thông điệp.

- Xác định ngẫu nhiên vị trí các bit thay đổi dựa vào một hàm mật mã ngẫu nhiên, vì thế khó có thể thành lập lại thông điệp mà không biết cơ chế gieo của hàm ngẫu nhiên.

Theo cách này, thông điệp giấu được bảo vệ bởi hai khóa khác nhau, vì thế đạt được độ an toàn cao hơn trước đó. Điều này cũng gần như bảo vệ toàn vẹn thông điệp, làm cho việc giả mạo thông điệp trở nên khó khăn hơn (độ phức tạp tính toán cao, không triển khai được). Dù sao đi nữa, chúng ta không muốn thông điệp của chúng ta chỉ là bị mã hóa và thay đổi thì chúng ta phải tiến hành dưới các kênh truyền thông ẩn.

Có hai vấn đề quan trọng trong kỹ thuật này là : việc chọn các bức ảnh và chọn định dạng của ảnh (24 bit, 8 bit, nén hay không nén).

Trước tiên bức ảnh chọn là phải ngẫu nhiên, không có chủ định vì thế nó phải được chọn từ một tập các ảnh mà có lý do thích hợp để chuyển từ người gửi đến người nhận. Điều này tránh được sự nghi ngờ và không gây ra sự chú ý từ đối phương.

Tiếp theo đó, trong các bức ảnh được chọn phải có nhiều màu sắc khác nhau, nó phải “noisy” (không đơn điệu), vì thế mà việc thêm một ít các thông tin nhiễu vào cũng được che đậy bởi tính chất “noisy” trước đó của nó. Các ảnh có ít sự thay đổi màu sắc (có cấu trúc ảnh vững chắc) thì rất khó giấu thông tin vì khi giấu tin dễ gây sự chú ý của đối phương.

Vấn đề thứ hai liên quan đến kích thước file ảnh để giấu tin, điều này cũng bao gồm định dạng file ảnh. Thông thường để tăng thông tin giấu, ta sử dụng các file ảnh có kích thước lớn làm dữ liệu chứa vì các bức ảnh (được chứa trong file) này có nhiều không gian hơn để giấu tin. Tuy nhiên việc trao đổi các bức ảnh có kích thước lớn một cách không bình thường giữa hai người dễ gây ra sự tò mò chú ý của đối phương. Ví dụ, một bức ảnh thông thường trên internet có kích thước 500\*300 = 150000 điểm ảnh có kích thước là:

- Ảnh 24 bit màu : 150000 pixel \*24 bits/ pixel/8 bits/byte = 90000 byte.

- Ảnh 8 bit màu : 150000 pixel \*8 bits/ pixel/8 bits/byte = 15000 byte.

Quan sát kích thước file ảnh ta thấy rằng đối với một ảnh màu 24 bit không nén, kích thước 90000 byte là kích thước không bình thường bởi vì việc không nén nó và gửi đi là một điều lạ bởi vì thực tế rằng nếu bức ảnh được nén trước khi gửi đi thì chất lượng ảnh thay đổi không đáng kể trong khi đó kích thước file ảnh giảm sẽ tăng hiệu suất truyền tin ( thời gian truyền sẽ nhanh hơn ).

Để khắc phục điều này, ta có thể biến đổi một chút theo thuật toán nén JPEG là chèn các bit ít quan trọng vào những vùng mà việc nén ảnh làm mất mát thông tin ít nhất hoặc điều khiển việc làm tròn các hệ số của phép biến đổi Cosin rời rạc được sử dụng khi nén ảnh [9, 14,15].

Nếu muốn liên lạc giấu tin bằng phương pháp chèn vào các bit ít quan trọng, chúng ta cần phải có những bức ảnh có kích thước file nhỏ, hay nói cách khác chúng phải sử dụng đến các bức ảnh 8 bit vì kích thước các file ảnh này là bình thường và ít gây chú ý cho đối phương.

Đối với ảnh 256 màu là nó sử dụng một bảng màu được xác định theo chỉ số và việc thay đổi các bit ít quan trọng đồng nghĩa với việc chúng ta chuyển một pixel từ vị trí này sang một vị trí gần kề khác. Nếu có sự tương phản màu sắc giữa hai vị trí gần kề trong bảng màu thì điều này dẫn đến một điểm ảnh trong bức ảnh thay đổi màu một cách bất ngờ và vì thế thông điệp giấu trở nên dễ dàng phát hiện.

Để giải quyết vấn đề này, nhiều phương pháp đã được nghiên cứu như sắp xếp lại bảng màu sao cho giữa hai giá trị màu trong bảng màu không quá tương phản hay như giảm số màu trong bảng màu xuống và tái tạo lại các thành phần giống nhau trong bảng màu ở những vị trị gần kề nhau, vì thế sự khác nhau trước và sau khi nhúng thông điệp của ảnh chứa trở nên khó phát hiện.

Các kết quả thực nghiệm cho thấy rằng việc chèn vào các bit ít quan trọng nếu sử dụng ảnh 8 bit xám thì rất khó phát hiện bởi hệ thống thị giác của con người do có rất ít sự thay đổi các giá trị trong bảng màu của chúng [9, 14,17].

## 2. **Kỹ thuật mã hóa khối kết cấu bề mặt (Texture Block Coding)**

Kỹ thuật mã hóa khối kết cấu bề mặt được phát triển bởi Bender cùng các đồng nghiệp của ông và được xuất bản trong tạp chí IBM System năm 1996. Đây là một kỹ thuật có tỉ lệ bit rất thấp, khá bền với thay đổi affine trên ảnh nhưng điểm hạn chế chính của nó là nó cần sự can thiệp của con người trong quá trình mã hóa.

Kỹ thuật

Việc mã hóa khối kết cấu bề mặt được thực hiện thông qua việc sao chép (copy) một vùng từ một mẫu kết cấu bề mặt ngẫu nhiên trong bức ảnh khác mà có cùng kết cấu bề mặt .

Quá trình mã hóa được thực hiện bằng cách chọn thủ công một vùng để xử lý và sau đó sử dụng các mặt nạ để chọn vùng sao chép, ví dụ một vùng hình chữ (graphic text), để mà sau khi giải mã mặt nạ có thể trở nên nhìn thấy được (visible).

Quá trình giải mã có các bước sau:

- Tự động tạo sự tương quan cho bức ảnh. Điều này sẽ tạo ra các đỉnh (peak) ở mỗi điểm trong quá trình tự động tạo sự tương quan, nơi mà biểu thị vùng của bức ảnh bị chồng lên.

- Nâng vùng ảnh được biểu thị bằng các đỉnh trong bước 1. Ta tiến hành trừ ảnh từ vùng copy được nâng lên của nó, thêm vào các góc các giá trị 0 nếu cần.

- Bình phương kết quả và tạo ngưỡng để nhận lại các giá trị gần với 0. Vùng được sao chép trở nên hiển thị bằng các giá trị này.

Tỉ lệ dữ liệu

Một điều dễ nhận thấy là các vùng được copy phải khá lớn (ít nhất 16\*16 pixel) và chúng ta phải gắn các mặt nạ trên các vùng đó nên số lượng thông tin giấu được rất nhỏ. Thực ra chúng ta không thể thay đổi nhiều phần của bức ảnh theo cách này mà không làm cho nó trở nên phát hiện được.

Hơn nữa, không phải tất cả các phần nào của bức ảnh cũng thích hợp cho kỹ thuật này, chẳng hạn như những vùng ảnh có màu sắc thuần nhất, bởi vì dễ dàng thấy rõ (magnify) sự thay đổi của chúng.

Độ bền vững

Nếu kích thước khối để thực hiện thao tác copy đủ lớn, thì các thay đổi các phần bên trong mỗi khối sẽ giống nhau khi bức ảnh chịu tác động bởi các phép biến đổi không phải hình học, chẳng hạn như phép lọc, nén, xoay ảnh. Mặt nạ nhúng cũng có thể nhìn thấy được sau khi nhúng [8,18].

Việc xén hình sẽ phá hủy kết cấu bề mặt nếu như một trong 2 vùng được sao chép nằm ngoài vùng chọn xén (cropped area). Các phép biến đổi affine thường cũng biến đổi luôn mặt nạ vì thế có thể làm cho chúng không thể hiểu được. Nếu một trong hai vùng bị thay đổi bởi các tấn công có chủ ý thì thông tin giấu sẽ có thể bị phá hủy.

Độ dễ của việc phát hiện và giải thông tin

Thuật toán giải tin thì khá dễ thực hiện và nó không bao gồm bất cứ khóa nào trong quá trình mã vì thế việc giải tin là tương đối dễ dàng [8, 18].

Thích hợp cho giấu tin mật hay thủy vân

Do tỉ lệ dữ liệu giấu theo phương pháp này là ít và nó khá bền vững với các biến đổi không cố tình và do dễ dàng nhận lại được tin giấu nên kỹ thuật này thích hợp cho thủy vân hơn là giấu tin mật.

Vấn đề và các giải pháp

Hạn chế của kỹ thuật này là nó cần có sự can thiệp của con người trong việc chọn vùng để thực hiện sao chép và dán. Giải pháp khắc phục là lập trình cho máy làm việc này nhưng có sự giám sát của con người. 1.3 Các tiêu chí cần có ở AES

NIST đưa ra yêu cầu đối với AES đó là phải sử dụng phương pháp mã hóa khối với độ dài của key là 128, 192 và 256 bit để mã hóa và giải mã dữ liệu. Ngoài ra AES phải đáp ứng được những tiêu chí sau:

• Bảo vệ: Đây là một trong những tính năng hàng đầu AES cần phải có để đánh bại các đối thủ khác. Nó phải có khả năng chống lại các cuộc tấn công mạnh, quy mô lớn.

• Chi phí: AES mở ra nhiều cơ hội cho người dùng bằng cách phát hành trên toàn cầu và miễn phí bản quyền.

• Khả năng thực hiện: Linh hoạt, phù hợp và đơn giản chính là 3 yếu tố quan trọng hội tụ ở AES để đáp ứng trọn vẹn nhu cầu của người dùng.

## 3. Kỹ thuật Patchwork

Kỹ thuật

Patchwork là kỹ thuật giấu tin được phát triển bởi Bender cùng các đồng sự của ông và được xuất bản trong tạp chí IBM system năm 1996. Nó dựa trên mô hình giả ngẫu nhiên, thống kê.

Kỹ thuật giấu tin này dựa vào phân bố Gauss để nhúng thông tin vào ảnh. Hai tập hợp các điểm ảnh (patches) của bức ảnh đuợc chọn, được ký hiệu là A và B. Thuật toán được thực hiện bằng cách làm sáng một điểm ảnh ở tập A trong khi đó làm tối điểm ảnh ở tập B. Trong thuật toán này sử dụng bộ sinh số ngẫu nhiên được đáp ứng bằng khóa mật k. Khóa này được chia sẻ cho cả bên truyền và bên nhận để xác định được các điểm ảnh đã chọn để giấu tin và căn cứ vào đó để nhận lại thông tin giấu.

Ký hiệu S(n) = sum(a[i]-b[i], i = 1..n), trong đó a[i] và b[i] lần lượt là điểm thứ i trong trong tập patch A và B. Tổng S(n) này sẽ bằng 0 trong ảnh bình thường.

Sau khi thực hiện n lần các thao tác a[i] = a[i] + deltab[i] = b[i]- delta, giá trị này sẽ dần đến 2\*delta\*n.

Nếu như n đủ lớn và tính được giá trị S(n) của một bức ảnh, chúng ta có thể kết luận rằng ảnh đã được giấu tin bằng kỹ thuật Patchwork hay chưa.

Chúng ta có thể gán một “cấp độ chắc chắn” liên quan đến n trong quá trình mã hóa để có kết luận chắc chắn trong quá trình giải tin. Cũng trong phương pháp này, nếu chúng ta lặp lại càng nhiều thì thủy vân càng trở nên dễ nhìn thấy [8].

Tỉ lệ dữ liệu

Kỹ thuật giấu tin này có tỉ lệ dữ liệu thấp.

Độ vững chắc

Một trong những đặc tính quan trọng nhất của kỹ thuật Patchwork là nó có khả năng chịu đựng được phép xén ảnh và thay đổi tỉ lệ mức độ sáng và gamma.

Nếu giấu tin bằng kỹ thuật Patchwork, tin giấu sẽ bị phá hủy bởi các phép biến đổi affine như dịch chuyển, quay, thay đổi tỉ lệ ảnh.

Độ dễ của quá trình phát hiện và giải tin

Với kỹ thuật Patchwork, gần như không thể giải tin được nếu như không biết khóa và bộ sinh số giả ngẫu nhiên được sử dụng để tìm các điểm bị thay đổi.

Sự thích hợp cho giấu tin mật hay thủy vân

Do tính chất khá bền vững và tỉ lệ dữ liệu giấu khá thấp nên kỹ thuật giấu tin Patchwork khá thích hợp cho thủy vân.

Trong các hệ thống bảo vệ bản quyền, kỹ thuật này được sử dụng chủ yếu để xác nhận rằng bức ảnh có được nhúng thủy vân hay chưa bằng cách nhúng một chuỗi bit với độ chắc chắn đến 99.9999%, sau đó mới dùng khóa khác để nhúng vào các bit khác nhưng với cấp độ yếu hơn, vì thế thủy vân sẽ không làm ảnh hưởng đến bức ảnh nhiều. Cũng theo cách này, việc giải tin trước tiên sẽ tìm dòng bit xác nhận ảnh có nhúng thủy vân hay không, nếu tìm thấy thì tiếp tục tìm phần còn lại của thủy vân.

Vấn đề và các giải pháp:

Vấn đề đầu tiên là nếu ta sử dụng các điểm như một đơn vị của các patch, nhiễu mà chúng ta thêm vào ảnh nằm trên vùng tấn số cao. Các điểm này có thể bị lọc hay bị thay đổi bởi nhiều quá trình, từ phép nén mất mát đơn giản đến quá trình chuyển số - tương tự và ngược lại. Giải pháp là sử dụng các patches của một vùng nhỏ thay vì các điểm, với một phân bố delta sao cho trải nhiễu sang một băng rộng hoặc di chuyển nó đến vùng tần số thấp, làm cho nó ít bị loại bởi phép lọc.

Cả AES và DES đều là mật mã khối đối xứng nhưng AES mang đến hiệu quả cao bởi độ dài key mà nó sở hữu. Đem lên bàn cân có thể thấy các key 128 bit, 192 bit và 256 bit của AES mạnh gấp nhiều lần so với 56 bit của DES. Bên cạnh đó mã hóa AES cũng nhanh hơn DES.

## 4.Một số kỹ thuật khác

Ngoài các kỹ thuật giấu tin trên còn có một số kỹ thuật giấu tin khác được sử dụng trong giấu tin trong ảnh. Tùy vào các ứng dụng, các kỹ thuật này có thể được chọn để tăng cường tính bền vững hoặc tăng cường tỉ lệ dữ liệu dựa vào các đặc tính kỹ thuật của các phương pháp giấu tin.

Kỹ thuật thao tác trên các hệ số của phép biến đổi DCT

Kỹ thuật này được sử dụng chủ yếu trong các ứng dụng có liên quan đến nén mất mát thông tin. Dữ liệu ảnh trong không gian ảnh được biến đổi sang miền tần số. Năng lượng của bức ảnh được chia làm 3 vùng : vùng tần số cao, vùng tần số giữa, vùng tần số thấp. Để thông tin giấu không bị mất đi bởi các phép nén mất mát, thông thưòng ta chèn vào tấn số giữa. Thông tin được giấu trong vùng này sẽ tránh được phép lọc thông thấp, thông cao.

Kỹ thuật sử dụng mặt nạ tri giác (Perceptual Masking)

Kỹ thuật này nhúng dữ liệu vào những vùng được gắn mặt nạ bởi các vùng khác dựa vào hệ thống thị giác của con người. Việc nhúng có thể thực hiện trên miền không gian ảnh hoặc miền tần số. Tỉ lệ dữ liệu nhúng được trong kỹ thuật này phụ thuộc vào ảnh chứa và thông thường việc chọn vùng để giấu tin được thực hiện dưới sự giám sát của con người.

**CÂU LỆNH VÀ GIẢI THUẬT**

//thư viện

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Diagnostics;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp4

{

public partial class Form1 : Form

{

private object bit;

private object data;

private readonly int valueR;

public Form1()

{

InitializeComponent();

}

private void radGiaiMa\_CheckedChanged(object sender, EventArgs e)

{

if (radGiaiMa.Checked)

{

radMaHoa.Checked = false;

txtDuLieu.ReadOnly = true;

}

}

private void radMaHoa\_CheckedChanged(object sender, EventArgs e)

{

if (radMaHoa.Checked)

{

radGiaiMa.Checked = false;

txtDuLieu.ReadOnly = false;

}

}

// Chon ảnh để nhúng

private void btnTim\_Click(object sender, EventArgs e)

{

OpenFileDialog od = new OpenFileDialog();

od.Filter = "Ảnh định dạng bmp (\*.bmp) | \*.bmp";

od.Multiselect = false;

if (od.ShowDialog() == DialogResult.OK)

{

txtDuongDan.Text = od.FileName;

}

}

private void btnThucHien\_Click\_1(object sender, EventArgs e)

{

//Thực hiện nhúng ảnh va giải mã

if (radMaHoa.Checked)

{

if (!File.Exists(txtDuongDan.Text))

{

MessageBox.Show("Bạn phải chọn tệp tin ảnh");

btnTim.Focus();

return;

}

if (String.IsNullOrEmpty(txtDuLieu.Text))

{

MessageBox.Show("Bạn phải nhập dữ liệu cần nhúng vào ảnh");

txtDuLieu.Focus();

return;

}

//phân tích dữ liệu thành các bit

String bits = StringToBinary(txtDuLieu.Text);

Bitmap bmp = new Bitmap(txtDuongDan.Text);//đọc tệp ảnh

for (int i = 0; i < bits.Length; i++)

{

int x = i % bmp.Width;

int y = i / bmp.Width;

Color px = bmp.GetPixel(x, y);

int valueREncrypt = EncryptPixel(px.R, bits[i].ToString());

Color pxe = Color.FromArgb(px.A, valueREncrypt, px.G, px.B);

bmp.SetPixel(x, y, pxe);

}

// Ghi do dai chuoi vao pha R cua pixel cuoi cung

int x1 = bmp.Width - 1;

int y1 = bmp.Height - 1;

Color px1 = bmp.GetPixel(x1, y1);

Color pxe1 = Color.FromArgb(px1.A, txtDuLieu.Text.Length, px1.G, px1.B);

bmp.SetPixel(x1, y1, pxe1);

Color px2 = bmp.GetPixel(x1, y1);

// Luu anh sau khi nhung

SaveFileDialog sd = new SaveFileDialog();

sd.Filter = "Ảnh định dạng bmp (\*.bmp) | \*.bmp";

if (sd.ShowDialog() == DialogResult.OK)

{

bmp.Save(sd.FileName);

txtDuLieu.Text = "";

txtDuongDan.Text ="";

bmp.Dispose();

Bitmap bmp1 = new Bitmap(sd.FileName);

Color px3 = bmp1.GetPixel(x1, y1);

int r = px3.R;

}

}

else

{

// Tach du lieu ra khoi anh

if (!File.Exists(txtDuongDan.Text))

{

MessageBox.Show("Bạn phải chọn tệp tin ảnh");

btnTim.Focus();

return;

}

Bitmap bmp = new Bitmap(txtDuongDan.Text);

Color px1 = bmp.GetPixel(bmp.Width - 1, bmp.Height - 1);

int chieuDaiBit = px1.R \* 8;

String bits = "";

for (int i = 0; i < chieuDaiBit; i++)

{

int x = i % bmp.Width;

int y = i / bmp.Width;

Color px = bmp.GetPixel(x, y);

bits += DecryptPixel(px.R);

}

String duLieu = BinaryToString(bits);

txtDuLieu.Text = duLieu;

}

}

private int EncryptPixel(byte r, string v)

{

if (bit.Equals("0"))

{

if (valueR % 2 == 0)

{

return valueR;

}

else

{

return (valueR + 1) % 256;

}

}

else

{

if (valueR % 2 == 0)

{

return (valueR + 1) % 256;

}

else

return valueR;

}

throw new NotImplementedException();

}

private string DecryptPixel(byte r)

{

if (valueR % 2 == 0)

return "0";

else

return "1";

throw new NotImplementedException();

}

private string StringToBinary(string bits)

{

StringBuilder sb = new StringBuilder();

foreach (char b in bits.ToCharArray())

{

sb.Append(Convert.ToString(b, 2).PadLeft(8, '0'));

}

return sb.ToString();

throw new NotImplementedException();

}

private string BinaryToString(string bits)

{

List<Byte> byteList = new List<Byte>();

for (int i = 0; i < bits.Length; i += 8)

{

byteList.Add(Convert.ToByte(bits.Substring(i, 8), 2));

}

return Encoding.ASCII.GetString(byteList.ToArray());

throw new NotImplementedException();

}

}

}

**TỔNG KẾT**

Chúng tôi đã giới thiệu về chương trình, cách hoạt động, và thuật toán được sử dụng. Hi vọng qua bài báo cáo này bạn sẽ hiểu được đôi chút về tầm quan trọng của việc bảo mật thông tin cá nhân trong thời đại hiện nay. Nhưng không có gì là hoàn hảo cả, chương trình của chúng tôi cũng có những vấn đề đã giải quyết được và những vẫn đề còn tồn đọng.

Vấn đề đã được giải quyết: thiết kế được 1 giao diện đơn giản thân thiện với người dùng, giải quyết được vấn đề mã hóa và giải mã nhúng dữ liệu vàoảnh

Vấn đề còn tồn đọng: hiện tại chương trình chỉ mã hóa và giải mã được những ảnh .bmp chứ không hỗ trợ những ảnh khác và dữ liệu được nhúng vào còn bọ hạn chế

Trên đây là tất cả những gì chúng tôi đã làm để xây dựng lên 1 chương trình mã hóa để giúp người dùng có thể tự bảo mật những thông tin quan trọng của họ trong thời đại công nghệ 4.0 hiện nay.