

BỘ CÔNG THƯƠNG

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

**----🙣🕮🙡----**



**BÁO CÁO**

**Thuật Toán AES và One-Time Pad**

***Giảng viên hướng dẫn* : GV. Võ Ngọc Tấn Phước**

***Sinh viên thực hiện* : Đặng Minh Vương - 21026071**

**Lê Quốc Bảo - 21075561**

**Đặng Công Minh - 18051991**

**Nguyễn Văn Long - 20002975**

***TP.HCM, ngày tháng 5 năm 2023***

Mục lục

[**PHẦN MỞ ĐẦU** 3](#_Toc134459763)

[**I: Lý do chọn đề tài:** 3](#_Toc134459764)

[**II: Mã hóa AES:** 4](#_Toc134459765)

[**1: Khái niệm và nguyên lý hoạt động:** 4](#_Toc134459766)

[**2: Demo chương trình:** 12](#_Toc134459767)

[**III: Thuật toán One-Time Pad:** 13](#_Toc134459768)

[**1: Khái niệm và nguyên lý hoạt động:** 13](#_Toc134459769)

[**2: Demo chương trình:** 19](#_Toc134459770)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 20](#_Toc134459771)

[**PHỤ LỤC** 21](#_Toc134459772)

# **PHẦN MỞ ĐẦU**

## **I: Lý do chọn đề tài:**

Mã hóa là một quá trình biến đổi thông tin từ một dạng dữ liệu gốc sang một dạng dữ liệu khác, không thể đọc được bởi người không có quyền truy cập. Mục đích chính của mã hóa là bảo vệ tính bảo mật và riêng tư của thông tin trong quá trình truyền và lưu trữ.

Thông tin sau khi được mã hóa sẽ được biến đổi thành một chuỗi các ký tự hoặc bit không có nghĩa, chỉ có người có khóa mã hóa mới có thể giải mã để đọc được thông tin gốc. Điều này giúp ngăn chặn những người không được ủy quyền truy cập thông tin từ việc đọc hoặc xâm nhập vào dữ liệu quan trọng.

Mã hóa được sử dụng rộng rãi trong các ứng dụng bảo mật, chẳng hạn như các giao dịch tài chính trực tuyến, truyền tải email và tin nhắn, lưu trữ dữ liệu quan trọng, và trong các mạng máy tính.

Mã hoá dữ liệu là một công việc cơ bản và cần thiết đối với các doanh nghiệp đang hoạt động hiện nay. Bởi nó mang lại những lợi ích nhất định cho doanh nghiệp:

• Mã hoá được xem là giải pháp hiệu quả giúp mọi thông tin của doanh nghiệp khi truyền tải trên internet luôn được đảm bảo an toàn và toàn vẹn nhất.

• Thực hiện mã hoá giúp doanh nghiệp có thể dễ dàng ngăn chặn được các truy cập bất hợp pháp vào hệ thống thông tin đã được thực hiện việc bảo vệ. Bởi chỉ những người có mật khẩu hoặc có quyền truy cập vào khoá giải mã thì mới có thể đọc và hiểu được các dữ liệu, thông tin đã được mã hoá.

• Các thuật toán mã hoá sẽ cung cấp cho doanh nghiệp các yếu tố bảo mật then chốt như tính xác thực cho phép xác minh nguồn dữ liệu, tính toàn vẹn giúp đảm bảo các thông tin không bị thay đổi khi được gửi đi và không thu hồi để đảm bảo việc gửi dữ liệu không bị huỷ.

• Thực hiện mã hoá tương tự như việc gia tăng thêm mức độ bảo mật cho thông tin. Chính vì vậy, cho dù dữ liệu của bạn bị đánh cắp thì việc giải mã cũng rất khó khăn và tốn nhiều thời gian, công sức.

Từ những lý do trên đồng thời đề đảm bảo kiến thức môn học Nhập môn an toàn thông tin nên nhóm 18 chúng em đã thực hiện đề tài “Nghiên cứu, viết chương trình demo mã AES và One-Time Pad”.

## **II: Mã hóa AES:**

### **1: Khái niệm và nguyên lý hoạt động:**

AES là một thuật toán “mã hóa khối” (block cipher) ban đầu được tạo ra bởi hai nhà mật mã học người Bỉ là Joan Daemen và Vincent Rijmen. Kể từ khi được công bố là một tiêu chuẩn, AES trở thành một trong những thuật toán mã hóa phổ biến nhất sử dụng khóa mã đối xứng để mã hóa và giải mã (một số được giữ bí mật dùng cho quy trình mở rộng khóa nhằm tạo ra một tập các khóa vòng). Ở Việt Nam, thuật toán AES đã được công bố thành tiêu chuẩn quốc gia TCVN 7816:2007 năm 2007 về Thuật toán mã hóa dữ liệu AES.

Diagram

Description automatically generated

**â**

AES là một thuật toán mã hóa khối đối xứng với độ dài khóa là 128 bít (một chữ số nhị phân có giá trị 0 hoặc 1), 192 bít và 256 bít tương ứng dọi là AES-128, AES-192 và AES-256. AES-128 sử dụng 10 vòng (round), AES-192 sử dụng 12 vòng và AES-256 sử dụng 14 vòng.

A picture containing text, screenshot, font, number

Description automatically generated

Vòng lặp chính của AES thực hiện các hàm sau: SubBytes(), ShiftRows(), MixColumns() và AddRoundKey(). Ba hàm đầu của một vòng AES được thiết kế để ngăn chặn phân tích mã bằng phương thức “mập mờ“ (confusion) và phương thức “khuếch tán“ (diffusion), còn hàm thứ tư mới thực sự được thiết kế để mã hóa dữ liệu.

Trong đó “khuếch tán“ có nghĩa là các kiểu mẫu trong bản rõ (Dữ liệu đầu vào của phép mã hóa hoặc dữ liệu đầu ra của phép giải mã) được phân tán trong các bản mã (Dữ liệu đầu ra của phép mã hóa hoặc dữ liệu đầu vào của phép giải mã), “mập mờ“ nghĩa là mối quan hệ giữa bản rõ và bản mã bị che khuất.

Đối với phép mã hóa và phép giải mã, thuật toán AES sử dụng một hàm vòng gồm bốn phép biến đổi byte như sau: phép thay thế byte (một nhóm gồm 8 bít) sử dụng một bảng thay thế (Hộp-S), phép dịch chuyển hàng của mảng trạng thái theo các offset (số lượng byte) khác nhau, phép trộn dữ liệu trong mỗi cột của mảng trạng thái, phép cộng khóa vòng và trạng thái.

**Phép mã hóa:**

Tại thời điểm bắt đầu phép mã hóa, đầu vào được sao chép vào mảng trạng thái sử dụng các quy ước. Sau phép cộng khóa vòng khởi đầu, mảng trạng thái được biến đổi bẳng cách thực hiện một hàm vòng liên tiếp với số vòng lặp là 10, 12 hoặc 14 (tương ứng với độ dài khóa), vòng cuối cùng khác biệt không đáng kể với Nr-1 vòng đầu tiên. Trạng thái cuối cùng được chuyển thành đầu ra. Hàm vòng được tham số hóa bằng cách sử dụng một lược đồ khóa – mảng một chiều chứa các từ 4 byte nhận từ phép mở rộng khóa.

Phép biến đổi cụ thể gồm SubBytes(), ShiftRows(), MixColumns() và AddRoundKey() dùng để xử lý trạng thái.

**SubBytes():**

Phép biến đổi dùng trong phép mã hóa áp dụng lên trạng thái (kết quả mã hóa trung gian, được mô tả dưới dạng một mảng chữ nhật của các byte) sử dụng một bảng thay thế byte phi tuyến (Hộp S – bảng thay thế phi tuyến, được sử dụng trong một số phép thay thế byte và trong quy trình mở rộng khóa, nhằm thực hiện một phép thay thế 1-1 đối với giá trị mỗi byte) trên mỗi byte trạng thái một cách độc lập.

A picture containing text, number, line, font

Description automatically generated

**ShiftRows():**

Phép biến đổi dùng trong phép mã hóa áp dụng lên trạng thái bằng cách chuyển dịch vòng ba hàng cuối của trạng thái theo số lượng byte các offset khác nhau.

A picture containing sketch, diagram, line, pattern

Description automatically generated

**MixColumns():**

Phép biến đổi trong phép mã hóa thực hiện bằng cách lấy tất cả các cột trạng thái trộn với dữ liệu của chúng (một cách độc lập nhau) để tạo ra các cột mới.

A picture containing text, font, number, screenshot

Description automatically generated

**AddRoundKey():**

Phép biến đổi trong phép mã hóa và phép giải mã. Trong đó, một khóa vòng (các giá trị sinh ra từ khóa mã bằng quy trình mở rộng khóa) được cộng thêm vào trạng thái bằng phép toán XOR (phép toán hoặc và loại trừ). Độ dài của khóa vòng bằng độ dài của trạng thái.

**Mở rộng khóa:**

Thuật toán AES nhận vào một khóa mã K và thực hiện phép mở rộng khóa để tạo ra một lược đồ khóa. Phép mở rộng khóa tạo ra tổng số Nb(Nr+1) từ. Thuật toán yêu cầu một tập khởi tạo gồm Nb từ và mỗi trong số Nr vòng đòi hỏi Nb từ làm dữ liệu khóa đầu vào. Lược đồ khóa kết quả là một mảng tuyến tính các từ 4 byte.

A diagram of a block diagram

Description automatically generated with low confidence

**Phép giải mã:**

Các phép biến đổi trong phép mã hóa có thể được đảo ngược và sau đó thực hiện theo chiều ngược lại nhằm tạo ra phép giải mã trực tiếp của thuật toán AES. Các phép biến đổi sử dụng trong phép giải mã gồm: InvShiftRows(), InvSubBytes(), InvMixColumns() và AddRoundKey().

**InvSubBytes():**

Phép biến đổi InvSubBytes() là nghịch đảo của phép thay thế theo byte SubBytes(), trong đó sử dụng một hộp-S nghịch đảo áp dụng cho mỗi byte của trạng thái.

**InvShiftRows():**

Phép biến đổi InvShiftRows() là phép biến đổi ngược của ShiftRows(). Các byte trong ba từ cuối của trạng thái được dịch vòng theo số byte khác nhau. Ở hàng đầu tiên (r=0) không thực hiện phép chuyển dịch, ba hàng dưới cùng được dịch vòng Nb-shift(r,Nb) byte.

**InvMixColumns():**

Phép biến đổi InvMixColumns() là phép biến đổi ngược của MixColumns(). Nó thao tác theo từng cột của trạng thái, xem mỗi cột như một đa thức bốn hạng tử.

**Biến đổi nghịch AddRoundKey():**

Phép biến đổi AddRoundKey() là phép biến đổi thuận nghịch vì nó chỉ áp dụng một phép toán XOR nên nó được thực hiện như nhau ở cả phép mã hóa và phép giải mã.

Ngoài các phép giải mã trên, thuật toán AES còn cho phép thực hiện một phép giải mã tương đương có cùng thứ tự các phép biến đổi như trong phép mã hóa (các biến đổi được thay bằng các phép biến đổi ngược). Có thể thực hiện được điều này là nhờ một thay đổi trong lược đồ khóa. Hai tính chất tạo nên một phép giải mã tương đương là: Tính giao hoán giữa hai phép biến đổi SubBytes() và ShiftRows() (tính chất này cũng đúng với phép nghịch đảo InvSubBytes() và InvShiftRows()), Các phép toán trộn cột MixColumns() và InvMixColumns() là tuyến tính đối với đầu vào cột. Các tính chất này cho phép đảo ngược thứ tự của các phép biến đổi InvSubBytes() và InvShiftRows(). Thứ tự của các phép biến đổi AddRoundKey() và InvMixColumns() cũng có thể đảo ngược với điều kiện đảm bảo rằng các cột của lược đồ khóa giải mã được chỉnh sửa bằng cách sử dụng phép biến đổi InvMixColumns().

**Vấn đề thực hiện khóa:**

*Yêu cầu về độ dài khóa:*

Việc thực hiện khóa của thuật toán AES sẽ hỗ trợ ít nhất một trong ba độ dài khóa là 128 bít, 192 bít và 256 bít. Việc thực hiện khóa có thể tùy chọn hỗ trợ hai hoặc ba độ dài khóa, nhằm tăng thêm tính tương tác cho các thực hiện thuật toán.

*Tham số hóa độ dài khóa, kích thước khối và số vòng:*

AES quy định cụ thể các giá trị được phép dùng cho chiều dài khóa, kích thước khối và số vòng. Tuy nhiên, các giá trị này có thể thay đổi trong tương lai. Do đó, những nhà triển khai thuật toán AES có thể lựa chọn thiết kế linh hoạt với mong muốn của họ.

**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.

**Tấn công mã hóa AES:**

Mặc dù mã hóa AES là thuật toán mạnh mẽ nhưng không tránh khỏi các cuộc tấn công. Nhiều nhà nghiên cứu đã công bố danh sách các cuộc tấn công, chẳng hạn như:

Năm 2009 cryptanalysis (công cụ phân tích mật mã) đã tìm cách bẻ key của hệ thống AES.

Năm 2009, AES-128 đã trở thành mục tiêu của một cuộc tấn công, tuy nhiên mối đe dọa này không gây thiệt hại gì.

Đáng quan ngại nhất với mã hóa AES là phải đối mặt với side-channel attack (tấn công kênh phụ). Để xác định được key của mã hóa AES các cuộc tấn công này đã sử dụng những thông tin về thời gian, rò rỉ điện từ, manh mối âm thanh…

**AES có an toàn không?**

AES nếu được triển khai đúng quy trình thì sẽ đảm bảo an toàn tuyệt đối. Thế nhưng một điều cần lưu ý đó là bất kỳ một hệ thống nào cũng có thể bị tấn công nếu hacker biết được key mã hóa. Do đó các key mã hóa AES phải được bảo vệ bằng nhiều cách khác nhau như dùng mật khẩu mạnh, xác thực, tường lửa hay phần mềm chống độc hại. Ngoài ra các tổ chức phải đào tạo nhân viên để nâng cao nhận thức của họ trong việc bảo mật thông tin, tránh bị kẻ xấu lợi dụng.

**Một số ứng dụng của AES**

Bên cạnh ứng dụng trong việc đảm đảm an toàn cho các tài liệu chính phủ, AES cũng được áp dụng nhiều trong các lĩnh vực khác nhau. Cụ thể:

Ứng dụng cho tất cả người dùng phổ thông trong quá trình giải mã dữ liệu bằng cách truy cập vào trang web AES Encryption, nhập dữ liệu và áp mã khoá. Tuy nhiên, đây chỉ là phương pháp áp dụng cho các tác vụ thông thường và tính bảo mật thường không cao.

Mã hoá thông tin trên phần mềm với các ngôn ngữ lập trình như C/C++, Java hay Assembler. AES hỗ trợ rất nhiều cho các hệ điều hành như Linux hay Windows.

AES áp dụng cho các thiết bị phần cứng bao gồm dòng thiết bị dựa trên hoạt động của hệ vi xử lý và dòng thiết bị cắm qua cổng USB hoặc thẻ thông minh Smart Card.

Ứng dụng trong truyền thông tin qua Internet thông qua kết nối HTTPS: Dữ liệu sẽ được mã hoá và giải mã thông qua thuật toán AES, thông tin được bảo mật tốt hơn khi so sánh với kết nối HTTP. Bên cạnh đó, wifi hiện nay cũng được ứng dụng thuật toán AES, khi kết hợp với giao thức WPA2, giao tiếp này trở nên an toàn, hiệu quả hơn nhiều và ngăn chặn tấn công trung gian. Bên cạnh đó, AES cũng được sử dụng để mã hoá wifi trên router, kết hợp với giao thức phổ biến WPA2 được gọi là AES/WPA2. AES còn được sử dụng nhằm hỗ trợ mã hoá SSL.

### **2: Demo chương trình:**

A screenshot of a computer

Description automatically generated

## **III: Thuật toán One-Time Pad:**

### **1: Khái niệm và nguyên lý hoạt động:**

**One-time pad (OTP)** là một kỹ thuật mã hóa không thể bị bẻ khóa, nhưng yêu cầu sử dụng khóa chia sẻ trước một lần có cùng kích thước hoặc dài hơn thông điệp được gửi. Trong kỹ thuật này, một bản rõ được ghép nối với một Khóa bí mật ngẫu nhiên (còn được gọi là One-time pad). Sau đó, mỗi bit hoặc ký tự của bản rõ được mã hóa bằng cách kết hợp nó với bit hoặc ký tự tương ứng từ vùng đệm bằng cách sử dụng phép cộng mô-đun. Bản mã thu được sẽ không thể giải mã hoặc phá vỡ nếu bốn điều kiện sau được đáp ứng:

**Chìa khóa** phải thực sự ngẫu nhiên

**Khóa** ít nhất phải dài bằng bản rõ

**Chìa khóa** không bao giờ được sử dụng lại toàn bộ hoặc một phần

**Chìa khóa** phải được giữ bí mật hoàn toàn

Nó cũng đã được chứng minh rằng bất kỳ mật mã nào có thuộc tính bảo mật hoàn hảo đều phải sử dụng các khóa với các yêu cầu tương tự như khóa OTP.

**Lịch sử của One-Time Pad**

OTP xuất hiện từ đầu thế kỉ 20 và còn có tên gọi khác là Vernam Cipher, nó được mệnh danh là cái chén thánh của ngành mã hóa dữ liệu.

OTP là thuật toán duy nhất chứng minh được về lý thuyết là không thể phá được ngay cả với tài nguyên vô tận (tức là có thể chống lại kiểu tấn công brute-force).

Frank Millernăm 1882 là người đầu tiên mô tả hệ thống đệm dùng one-time pad để bảo mật điện báo.

**Cơ sở mã hoá One-time Pad**

Cơ sở xác định bản mã và bản rõ của One-time Pad hết sức đơn giản. Ta chỉ cần thực hiện phép XOR bản mã (hoặc bản rõ) với khoá.

A picture containing text, font, white, handwriting

Description automatically generated

A picture containing text, font, screenshot, line

Description automatically generated

Với OTP, khóa k phải đáp ứng 3 điều kiện sau đây:

Độ dài của khóa phải bằng kích thước bản rõ.

Khóa phải được chọn hoàn toàn ngẫu nhiên (truly random).

Và khóa chỉ được sử dụng một lần.

Nếu thỏa mãn 3 điều kiện trên, hệ mã OTP sẽ là an toàn tuyệt đối (perfect security) theo định lý của Clause Shannon. OTP sẽ cho ta tốc độ tính toán rất nhanh

**Biến thể OTP trong thực tế:**

A picture containing text, font, white, handwriting

Description automatically generated

Trong đó, k có kích thước nhỏ hơn rất nhiều so với m.

**Cách hoạt động của mã hoá One-time Pad:**

Ý tưởng thực hiện sẽ lần lượt triển khai theo các bước như sau:

Chuyển dữ liệu sang dạng nhị phân (ta gọi đây là plaintext).

Sinh ngẫu nhiên một mảng dữ liệu nhị phân với chiều dài bằng chiều dài của plaintext (ta gọi đây là pad). Chú ý pad ở đây phải là truly random.

XOR từng bit trong plaintext với bit ở vị trí tương ứng trong pad để được dữ liệu mã hóa (ta gọi đây là cipher).

Để lấy plaintext từ cipher, ta chỉ cần thực hiện XOR cipher với pad.

A screen shot of a computer code

Description automatically generated with low confidence

A picture containing text, screenshot, font, number

Description automatically generated

**One-Time Pad – Hoàn hảo nhưng không khả thi:**

Nếu OTP vừa đơn giản là vừa bảo mật thì tại sao ta lại cần tới những phương pháp mã hóa khác như AES hay RSA? Đó là vì để đạt được độ bảo mật hoàn hảo, OTP cần được sử dụng đúng cách. Ta cần đáp ứng được các điều kiện sau.

Pad phải là truly random và phải có độ dài ít nhất bằng với plaintext.

Phía gửi và phía nhận đều phải biết nội dung của pad và phải giữ kín không để lộ cho bên thứ ba.

Không được tái sử dụng pad.

Ta sẽ phân tích từng điều kiện này.

**Pad phải là truly random và phải có độ dài ít nhất bằng với plaintext**

Việc tạo ra dữ liệu ngẫu nhiên một cách truly random không dễ như ta tưởng, nhất là khi ta cần lượng dữ liệu lớn. Trong phần lớn các trường hợp, máy tính sử dụng các giải thuật để tạo dữ liệu có vẻ là ngẫu nhiên, nhưng thực ra các dữ liệu đó phụ thuộc vào một dữ liệu mồi (seed). Dữ liệu mồi đó có thể được người dùng thiết lập trước, hoặc được lấy từ những nguồn khó đoán như thời gian hệ thống hiện tại.

Một số giải thuật sinh dữ liệu giả ngẫu nhiên như CryptGenRandom, Yarrow hay Fortuna được gọi là cryptographically-secure. Tức là dữ liệu do chúng sinh ra hầu như không khác gì dữ liệu thực sự ngẫu nhiên, và ta có thể dùng chúng cho mục đích bảo mật.

Vậy ta có thể dùng giải thuật có tính chất cryptographically-secure để sinh pad cho OTP không? Rất tiếc là không, vì một sợi xích dù chắc đến đâu cũng có thể bị gãy chỉ vì một mắt xích yếu. Trong trường hợp này mã hóa của ta chỉ đạt được mức độ bảo mật là cryptographically-secure. Mức bảo mật này đủ để sử dụng trong thực tế, nhưng ta chưa thể gọi đó là hoàn hảo.

Ta cũng có một vài phương pháp để sinh dữ liệu thực sự ngẫu nhiên. Ta có thể đo một nguồn dữ liệu ngẫu nhiên như áp suất khí quyển, nhiệt độ môi trường hay các hiện tượng lượng tử rồi chuyển đổi nó thành dạng số. Tuy nhiên phương pháp này chậm và phức tạp. Mỗi khi cần sinh dữ liệu, ta cần đợi để thu thập đủ lượng dữ liệu ngẫu nhiên từ môi trường, và ta phải có phương án để loại trừ những sai số mang tính hệ thống trong quá trình đo đạc.

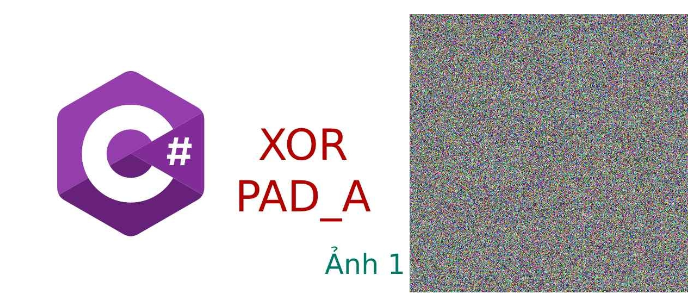
**Phía gửi và phía nhận đều phải biết nội dung của pad và phải giữ kín không để lộ cho bên thứ ba:**

Điều kiện này nghe có vẻ đơn giản nhưng lại không hề dễ thực hiện. Nếu như ta có cách để chuyển một pad dài bằng plaintext cho người nhận mà không sợ bị lộ thì sao ta không dùng cách đó để chuyển luôn plaintext, sao còn phải dùng OTP? Đây chính là nhược điểm lớn nhất và nó làm hạn chế đáng kể tính ứng dụng của OTP.

Vậy ta có thể áp dụng các giải thuật trao đổi khóa (key exchange) vào OTP được không? Đúng là việc trao đổi dữ liệu một cách bảo mật trên kênh thông tin không an toàn không phải là một vấn đề mới. Những giải thuật mã hóa dùng public key như RSA đã xử lý vấn đề này tương đối ổn bằng các giải thuật như Diffie–Hellman hay ECDH. Tuy nhiên khi áp dụng chúng vào OTP, ta gặp phải hai trở ngại lớn. Trở ngại thứ nhất là các giải thuật trao đổi khóa được thiết kế cho lượng dữ liệu nhỏ, chúng không hiệu quả khi áp dụng với dữ liệu lớn như pad (nhớ là pad cần dài ít nhất bằng plaintext). Trở ngại thứ hai là không giải thuật trao đổi khóa nào là hoàn hảo, nên nếu dùng chúng thì OTP cũng không còn hoàn hảo nữa.

**Không được tái sử dụng pad:**

Ta sẽ thực hiện một thử nghiệm nhỏ để thấy vì sao không được tái sử dụng pad.



A picture containing design, typography

Description automatically generated with medium confidence

A picture containing graphics, graphic design, logo, clipart

Description automatically generated

Như hình trên, ta thấy khi 2 hình ảnh riêng lẻ lần lượt mã hoá OTP cùng 1 khoá pad thì kết quả trả về khá là bảo mật.

Tuy từ từng cipher riêng lẻ ta không biết được plaintext là gì, sau khi XOR hai cipher với nhau ta lại biết được tương đối nhiều điều về cả hai plaintext. Các bạn có thể tự mình thực hiện thử nghiệm này bằng script dưới đây. Tôi dùng urandom để sinh pad nên đây không đúng 100% là OTP, nhưng như thế là đủ cho thử nghiệm này.

**One-time Pad có ứng dụng nào trong thực tế?**

Thật tiếc là một phương pháp mã hóa đơn giản mà an toàn như OTP lại không mấy hữu ích trong thực tế. Liệu có trường hợp nào mà OTP là phù hợp không? Thực ra là có, trong thực tế lúc ta có kênh bảo mật thì chưa chắc đã có dữ liệu để trao đổi, mà lúc có dữ liệu để trao đổi thì có khi kênh bảo mật đó đã không còn tồn tại. Nếu ta gặp trực tiếp đối tác và đưa cho họ một USB chứa sẵn dữ liệu thực sự ngẫu nhiên thì sau này ta có thể dùng dữ liệu ngẫu nhiên đó làm pad để trao đổi thông tin mà hoàn toàn không sợ bị nghe trộm. Có lẽ các bạn đã đoán được rằng OTP rất hữu ích cho hoạt động gián điệp :).

Đoạn trích dưới đây được lấy từ Crypto Museum:

A picture containing text, screenshot, font, number

Description automatically generated

### **2: Demo chương trình:**

A screenshot of a computer

Description automatically generated with medium confidence

# **DANH MỤC TÀI LIỆU THAM KHẢO**

**\*Website:**

**1:** <https://viblo.asia/p/cau-truc-va-thuat-toan-advanced-encryption-standard-chuan-ma-hoa-nang-cao-924lJYe8ZPM>

**2:** <http://iottuonglai.com/aes-encryption-la-gi.html>

**3:** <https://nam.name.vn/bai-13-one-time-pad-hoan-hao-nhung-khong-kha-thi.html>

**4:** <https://codelearn.io/sharing/ma-mot-lan-bao-mat-tuyet-doi>

5: <https://viblo.asia/p/tim-hieu-ve-he-ma-one-time-pad-yMnKM62NZ7P>

# **PHỤ LỤC**

**TRƯỜNG ĐH CÔNG NGHIỆP TP.HCM CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**Độc lập – Tự do – Hạnh phúc**

**BẢNG ĐÁNH GIÁ KẾT QUẢ LÀM VIỆC NHÓM**

LỚP HỌC PHẦN: DHKTPM16B - 420300100407

NHÓM: 18

1. Phân công công việc:

Nhóm có tổ chức 1 buổi họp online:

- Thời gian bắt đầu: 20h00 ngày 22/4/2023

- Thời gian kết thúc: 21h30 ngày 22/4/2023

- Chủ trì: Nguyễn Văn Long.

- Thành phần tham dự gồm: Đặng Minh Vương, Lê Quốc Bảo, Nguyễn Văn Long.

Qua cuộc họp, nhóm đã thảo luận và cùng nhau trao đổi về chủ đề được giao (AES và One-Time Pad). Được sự thống nhất của tất cả các thành viên trong nhóm, nhóm trưởng đã phân công công việc cho các thành viên như sau:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Họ Tên | MSSV | Vai trò | Công việc thực hiện |
| 1 | Đặng Minh Vương | 21026071 | Thành viên | Code thuật toán One-TimePad |
| 2 | Lê Quốc Bảo | 21075561 | Thành viên | Code thuật toán One-TimePad |
| 3 | Đặng Công Minh | 18051991 | Thành viên |  |
| 4 | Nguyễn Văn Long | 20002975 | Trưởng nhóm | Code thuật toán AES, làm file báo cáo tổng hợp. |

2. Kết quả đánh giá:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Họ Tên | Vai Trò | Mức độ  đóng  góp | Nhận xét, góp  ý của nhóm | Điểm |
| 1 | Đặng Minh Vương | Thành viên |  |  |  |
| 2 | Lê Quốc Bảo | Thành viên |  |  |  |
| 3 | Đặng Công Minh | Thành viên |  |  |  |
| 4 | Nguyễn Văn Long | Nhóm trưởng |  |  |  |

# 