

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

MẬT MÃ ỨNG DỤNG

Project #1: Password manager

|  |  |
| --- | --- |
| Nguyễn Tiến Lâm | 20215606 |
|  |  |
|  |  |

**HANOI, 10/2024**

**1 Short-answer Questions**

1. **Question:** Briefly describe your method for preventing the adversary from learn- ing information about the lengths of the passwords stored in your password manager.

**Answer:** Ta sử dụng hàm createCipheriv trong thư viện crypto với tùy chọn setAutoPadding() mặc định, vì vậy tất cả các mật khẩu sẽ có cùng độ dài.

1. **Question:** Briefly describe your method for preventing swap attacks. Provide an argument for why the attack is prevented in your scheme.

# Answer:

* Kho lưu trữ khóa-giá trị trong trình quản lý mật khẩu lưu trữ một cặp bao gồm tên miền đã được HMAC hash và mật khẩu được mã hóa kèm theo HMAC hash của tên miền đó (trong trường hợp này là AES-256/GCM với IV).
* keychain = {**HMAC(tên miền), IV || AES-256-GCM(HMAC(tên miền) || mật khẩu**)}

Chứng minh phản chứng: Giả sử có một kẻ tấn công có thể thực hiện thành công cuộc tấn công tráo đổi. Điều này có nghĩa là kẻ tấn công có thể tráo đổi bản ghi cho **HMAC(tên miền), IV1 || AES-256-GCM(HMAC(tên miền1) || mật khẩu1)** và **HMAC(tên miền), IV2 || AES-256-GCM(HMAC(tên miền2) || mật khẩu2)** sao cho các bản ghi mới là **HMAC(tên miền), IV || AES-256-GCM(HMAC(tên miền1) || mật khẩu2)** và **HMAC(tên miền), IV || AES-256-GCM(HMAC(tên miền2) || mật khẩu1)**.

Tuy nhiên, điều này có nghĩa là kẻ tấn công đã phá vỡ **AES-256-GCM() (tức là AES-256/GCM)** vì tính toàn vẹn của mã hóa đã bị vi phạm khi hắn có thể gửi một bản mã mới

**IV || AES-256-GCM(HMAC(tên miền1) || mật khẩu2)**

hoặc **IV || AES-256-GCM(HMAC(tên miền2) || mật khẩu1)**

mà thách thức đã chấp nhận. Điều này dẫn đến một mâu thuẫn. Do đó, phương pháp này bảo vệ khỏi cuộc tấn công tráo đổi.

1. **Question:** In our proposed defense against the rollback attack, we assume that we can store the SHA-256 hash in a trusted location beyond the reach of an adversary. Is it necessary to assume that such a trusted location exists, in order to defend against rollback attacks? Briefly justify your answer.

**Answer:** Cần thiết phải giả định rằng tồn tại một vị trí lưu trữ đáng tin cậy. Điều này là do chiến lược bảo toàn tính toàn vẹn của chúng tôi, bao gồm việc đệm **HMAC(tên miền)** với mật khẩu đã đệm, chỉ bảo vệ được chống lại các cuộc tấn công tráo đổi nếu chúng tôi giả định rằng hash không được lưu trữ trong một nơi an toàn. Kẻ tấn công có thể thay thế mục nhập **"AES-256-GCM(mật khẩu || HMAC(tên miền)"** hiện tại bằng một mục nhập **"AES-256-GCM(mật khẩu || HMAC(tên miền)"** cũ hơn, và vì **HMAC(tên miền)** vẫn khớp, trình quản lý mật khẩu sẽ không thể phát hiện ra điều này.

1. **Question:** Because HMAC is a deterministic MAC (that is, its output is the same if it is run multiple times with the same input), we were able to look up domain names using their HMAC values. There are also randomized MACs, which can output different tags on multiple runs with the same input. Explain how you would do the look up if you had to use a randomized MAC instead of HMAC. Is there a performance penalty involved, and if so, what?

# Answer:

Vì HMAC là một hàm PRF và không thể bị giả mạo, nó đáp ứng các tiêu chí để tạo khóa. Có thể thay thế HMAC bằng một MAC ngẫu nhiên nếu nó đáp ứng các yêu cầu này. Tuy nhiên, để tra cứu tên miền khi sử dụng MAC ngẫu nhiên, cơ sở dữ liệu sẽ cần lưu trữ cả tên miền và các giá trị MAC đã được tạo cho nó. Khi tra cứu một tên miền, giá trị MAC của nó sẽ cần được tính toán lại và so sánh với các giá trị MAC được lưu trữ để tìm các kết quả phù hợp, thay vì chỉ thực hiện tra cứu trực tiếp.

Điều này sẽ dẫn đến việc giảm hiệu suất so với việc sử dụng MAC xác định như HMAC. Với HMAC, quá trình tra cứu đơn giản là so sánh giá trị. Nhưng với MAC ngẫu nhiên, cần thực hiện thêm các bước sau cho mỗi lần tra cứu:

* Tính lại giá trị MAC cho tên miền.
* Quét cơ sở dữ liệu, lấy và tính lại giá trị MAC cho tất cả các mục.
* So sánh giá trị MAC vừa tính với mỗi giá trị MAC đã lưu để tìm kết quả phù hợp.

Điều này tốn nhiều tài nguyên tính toán hơn so với chỉ một lần tra cứu đơn giản. Nó cũng yêu cầu nhiều lượt truy cập bộ nhớ hơn, vì giá trị MAC của từng mục trong cơ sở dữ liệu cần được tính lại. Do đó, sẽ có sự gia tăng cả về thời gian tính toán lẫn sử dụng bộ nhớ/lưu trữ so với việc sử dụng MAC xác định như HMAC.

1. **Question:** In our specification, we leak the number of records in the password manager. Describe an approach to reduce the information leaked about the number of records. Specifically, if there are k records, your scheme should only leak log2

(k) (that is, if k1 and k2 are such that log2 (k1) = log2 (k2), the attacker should not be able to distinguish between a case where the true number of records is k1 and another case where the true number of records is k2).

**Answer:** Một cách tiếp cận có thể là thiết lập một giới hạn tối đa cao về tổng số mật khẩu được phép lưu trữ. Trình quản lý mật khẩu sẽ luôn chiếm cùng một lượng không gian đĩa, đệm các vùng chưa sử dụng bằng các bit ngẫu nhiên. Tuy nhiên, thiết kế này sẽ không tối ưu về việc sử dụng bộ nhớ và lưu trữ. Điều quan trọng cần lưu ý là việc đặt giới hạn trên về số lượng mục sẽ giới hạn số mật khẩu có thể lưu trữ trong trình quản lý. Nhưng giới hạn này sẽ chấp nhận được miễn là nó đủ cao để người dùng khó có thể đạt tới mức tối đa này.

1. **Question:** What is a way we can add multi-user support for specific sites to our password manager system without compromising security for other sites that these users may wish to store passwords of? That is, if Alice and Bob wish to access one stored password (say for nytimes) that either of them can get and update, without allowing the other to access their passwords for other websites.

**Answer:** Mỗi mật khẩu của người dùng nên được mã hóa bằng một khóa mã hóa riêng biệt, được tạo từ mật khẩu chính của họ hoặc một cơ chế xác thực an toàn khác. Điều này đảm bảo rằng mật khẩu của từng người dùng vẫn an toàn và không thể truy cập bởi người khác. Để quản lý quyền truy cập vào các trang web dùng chung, chúng ta có thể triển khai cơ chế kiểm soát truy cập, ví dụ như kiểm soát truy cập dựa trên vai trò (RBAC) hoặc danh sách kiểm soát truy cập (ACL).

Đối với các mật khẩu của các trang web dùng chung, tạo ra một khóa mã hóa chung và chia sẻ nó giữa những người dùng được ủy quyền. Khóa mã hóa chung này nên được phân phối một cách an toàn cho những người dùng được phép, có thể thông qua một giao thức trao đổi khóa hoặc một kênh liên lạc bảo mật.