ỨNG DỤNG CHIA SẺ GHI CHÚ

Bộ môn Công nghệ Tri thức

Mục tiêu

Trong bài tập này, sinh viên sẽ xây dựng một nền tảng chia sẻ ghi chú bảo mật với tính năng kiểm soát thời gian truy cập. Mô tả chi tiết sẽ được đề cập bên dưới, sau đây là tổng quan các mục tiêu cơ bản mà sinh viên cần phải hoàn thành:

- Thiết kế hệ thống client-server đơn giản, chỉ cần hoạt động được trong mạng nội bộ hoặc trên cùng một máy.
- Triển khai các hàm thực hiện mã hóa ở client để bảo vệ dữ liệu trước khi tải lên server.
- Xây dựng cơ chế xác thực người dùng, như cơ chế xác thực thông qua mật khẩu, đồng thời quản lý phiên làm việc thông qua token bảo mật.
- Cài đặt tính năng chia sẻ ghi chú an toàn. Yêu cầu cơ bản là các ghi chú được chia sẻ thông qua url tạm thời trong thời gian định trước, nhằm đảm bảo tính bảo mật và kiểm soát truy cập dữ liệu.
- Báo cáo chi tiết về thiết kế, các thách thức gặp phải, giải pháp và phương pháp kiểm thử, cùng với đề xuất các cải tiến trong tương lai.

Giới thiệu bài tập

Bài tập này yêu cầu sinh viên cần cách kết hợp các kỹ thuật sau nhằm tạo ra nền tảng đảm bảo việc chia sẻ ghi chú an toàn và riêng tư:

Client-side encryption Client-side encryption (mã hóa phía máy khách) đảm bảo rằng dữ liệu nhạy cảm được mã hóa trước khi được gửi ra khỏi máy của người dùng, bảo vệ quyền riêng tư và tính toàn vẹn dữ liệu. Phương pháp này ngăn chặn máy chủ truy cập vào dữ liệu gốc, giảm nguy cơ từ các cuộc tấn công từ máy chủ độc hại/bị điều khiển. AES là thuật toán mã hóa đối xứng được tin dùng rộng rãi để mã hóa dữ liệu nhỏ một cách hiệu quả. Tuy nhiên, để vừa đảm bảo dữ liệu được mã hóa đúng đắn, vừa đảm bảo tính toàn vẹn của dữ liệu, các bạn sinh viên cần tìm hiểu các mode của AES và chọn mode phù hợp. Mỗi ghi chú nên được mã hóa bằng một khóa riêng, nhằm củng cố tính bảo mật.

User authentication User authentication (xác thực người dùng) cho phép chỉ những người dùng hợp lệ truy cập hệ thống. Mật khẩu nên được lưu trữ trên máy chủ một cách an toàn bằng cách hashing và các kỹ thuật hỗ trợ như salting/peppering. JWT giúp duy trì các phiên đăng nhập an toàn mà không làm giảm trải nghiệm người dùng. Các cơ chế này cân bằng giữa tính thân thiện với người dùng và tính bảo mật.

End-to-end encryption End-to-end encryption (mã hóa đầu-cuối) đóng vai trò đảm bảo tính bảo mật khi chia sẻ dữ liệu giữa các thiết bị hay giữa các người dùng. Tính năng này bao gồm việc sử dụng trao đổi khóa Diffie-Hellman để tạo ra một **khóa phiên** duy nhất nhằm mã hóa và giải mã các ghi chú được chia sẻ, khóa này sẽ bị hủy sau khi sử dụng.

Time-sensitive access control Tính năng truy cập tạm thời giúp kiểm soát thời gian truy cập tài nguyên, gia tăng tính bảo mật và tính thực tiễn. Ví dụ, các liên kết này có thể được sử dụng để chia sẻ ghi chú công khai hoặc cung cấp quyền truy cập ngắn hạn vào dữ liệu được lưu trên cloud. Quyền truy cập bị giới hạn thông qua metadata như thời gian hết hạn truy cập, số lượt truy cập, yêu cầu xác thực để truy cập... Máy chủ thực thi các quy tắc này và có thể tự động xóa dữ liệu đã hết hạn.

1 Gợi ý thực hiện

1.1 Chức năng cơ bản

Các chức năng cơ bản cần được triển khai:

1. **Xác thực người dùng**: Triển khai đăng ký và đăng nhập. Lưu trữ mật khẩu và quản lý phiên làm việc bằng token bảo mật.
2. **Mã hóa/giải ghi chú**: Mã hóa ghi chú ở client bằng AES trước khi tải lên server, giải mã ghi chú cá nhân hoặc ghi chú được chia sẻ.
3. **Giao diện điều khiển chương trình**: Cho phép người dùng liệt kê ghi chú, hủy chia sẻ ghi chú hoặc xóa các ghi chú.
4. **Giới hạn thời gian truy cập**: Tạo và xác thực URL tạm thời để truy cập ghi chú. Đảm bảo các ghi chú không thể truy cập sau khi hết hạn.

Lưu ý: Không cần cài đặt nền tảng tạo ghi chú, ghi chú ở đây có thể là một tập tin bất kỳ như `.docx`,`.pdf`,`.txt`,... Bài tập này tập trung vào các chức năng mã hóa, xác thực và chia sẻ dữ liệu, không yêu cầu cụ thể GUI hay CLI.

1.2 Giới hạn thời gian truy cập

Phát triển tính năng chia sẻ ghi chú qua URL các chức năng cơ bản sau:

- **Tạo URL**: Tạo URL duy nhất từ ID ghi chú, khóa mã hóa và thời gian hết hạn (và các metadata khác nếu cần thiết).
- **Xác thực URL**: Kiểm tra tính hợp lệ của URL và kiểm tra các quy tắc qui định bởi metadata như thời gian hết hạn, yêu cầu xác thực nếu có,...

Lưu ý: Đây là các gợi ý. Bạn có thể tự thiết kế miễn là đạt được chức năng yêu cầu.

1.3 Ngôn ngữ lập trình và thư viện

Trong bài tập này, sinh viên có thể dùng bất kỳ ngôn ngữ lập trình nào cũng như được phép sử dụng lại **các thư viện mã hóa chuẩn và phổ biến** thay vì tự viết các sơ đồ mã hóa từ đầu. Điều này giúp đảm bảo ứng dụng được làm ra có tính an toàn cao, có thể sử dụng trong thực tiễn. Đồng thời, việc sử dụng các thư viện và ngôn ngữ lập trình được công nhận rộng rãi giúp giảm thiểu rủi ro bảo mật, dễ bảo trì, đảm bảo an toàn dữ liệu và tuân thủ các tiêu chuẩn bảo mật mới nhất.

Tuy nhiên, các thư viện được sử dụng **phải được liệt kê trong báo cáo**, cũng như cần được kiểm tra kỹ về việc thư viện đó có đạt chuẩn hay không, có được khuyến khích sử dụng trong các ứng dụng thực tế không. Ngoài ra, sinh viên cần nắm rõ cách sử dụng thư viện, tránh trường hợp dùng sai, hoặc cấu hình sai dẫn đến không đảm bảo tính an toàn và bảo mật của ứng dụng.

1.4 Tài liệu đọc thêm

- **Tổng quan:**
  - Secure Coding Practices - OWASP Foundation: Hướng dẫn cơ bản về các nguyên tắc lập trình an toàn.
  - Understanding Authentication: Bài viết cung cấp cái nhìn sâu sắc về cơ chế xác thực và thực tiễn tốt nhất.
- **C++:**
  - Secure Coding in C and C++: Hướng dẫn cơ bản về lập trình C/C++ an toàn.
- **Rust:**
  - Rust Cookbook: Các ví dụ thực tiễn, bao gồm mật mã học và triển khai máy chủ web.

2 Yêu cầu nộp bài

2.1 Kiểm thử

Tạo thư mục test chứa các tập tin hỗ trợ kiểm thử. Các yêu cầu kiểm thử cần bao gồm:

- **Xác thực**: Kiểm tra đăng ký và đăng nhập, bao gồm các trường hợp lỗi.
- **Mã hóa/giải mã**: Xác minh mã hóa/giải mã chính xác và đảm bảo khóa mã hóa được bảo vệ.
- **Giới hạn truy cập**: Đảm bảo các liên kết hết hạn không thể truy cập.
- **Mã hóa đầu-cuối**: Kiểm thử chức năng trao đổi khóa và sử dụng khóa phiên.

2.2 Báo cáo

Báo cáo cần đảm bảo các phần sau:

- **Tổng quan ứng dụng**: Mô tả mục tiêu ứng dụng, cách chạy chương trình từ mã nguồn nộp kèm, các chức năng đã triển khai và tính năng nâng cao (nếu có).
- **Thiết kế và kiến trúc**: Mô tả kiến trúc hệ thống, mục đích thiết kế, các thành phần chính và công nghệ sử dụng. Mô tả kèm sơ đồ thiết kế, sơ đồ luồng hoạt động, **không thêm các đoạn mã nguồn vào báo cáo**.
- **Chi tiết cài đặt**: Giải thích các trick hay những phần optimize **nếu có** trong quá trình thực hiện, nêu lý do và kết quả thực tiễn (như nhanh hơn hay tốn ít dữ liệu hơn bao nhiêu phần trăm).
- **Thách thức và giải pháp**: Trình bày các vấn đề đã gặp và cách giải quyết.
- **Kiểm thử**: Phương pháp, công cụ/framework sử dụng và kết quả kiểm thử.
- **Cải tiến tương lai**: Đề xuất các cải tiến hoặc tính năng bổ sung.

2.3 Các quy định khác

- Bài tập được theo **nhóm tối đa 4 sinh viên**.
- Thời gian thực hiện là 3 tuần tính từ lúc được đăng trên trang môn học.
- Cấu trúc bài nộp:

```
<MSSV1-MSSV2-MSSV3-MSSV4>.zip
├── project_02_source
│   └── *.cpp
├── project_02_report
│   └── *.pdf
└── project_02_test
    └── *..
```

  Trong đó:

  - `<MSSV1-MSSV2-MSSV3-MSSV4>` là các mã số sinh viên của nhóm sinh viên, sắp xếp theo thứ tự trong danh sách lớp.
  - `.zip` là định dạng nén cho bài làm (định dạng ZIP).
  - `project_02_source` là nơi chứa mã nguồn của bài tập, ngôn ngữ lập trình có quy định ở trên.
  - `project_02_report` là nơi chứa (các) báo cáo ở định dạng pdf.
  - `project_02_test` chứa các tập tin hỗ trợ kiểm thử.

- Chương trình thực hiện yêu cầu nào không biên dịch được hoặc lỗi thì phần đó không có điểm.
- Mọi thắc mắc vui lòng gửi về email: `ndhy@fit.hcmus.edu.vn`