

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----🙞🙜🕮🙞🙜-----

**BÁO CÁO BÀI TẬP LỚN MÔN HỌC**

**AN TOÀN VÀ BẢO MẬT THÔNG TIN**

|  |  |  |
| --- | --- | --- |
| **Đề tài:** | **XÂY DỰNG PHẦN MỀM CƠ SỞ DỮ LIỆU VỀ LỖ HỔNG BẢO MẬT Web/App/Os** | |
| **Lớp:** [**20231IT6043009**](https://qlht.haui.edu.vn/course/view.php?id=26796)  **Nhóm thực hiện :** | | **09** |
| **Thành viên trong nhóm :** | | **Trịnh Xuân Mạnh Nguyễn Quý Long**  **Bùi Quang Minh Nguyễn Văn Mùi**  **Bùi Huy Nam** |
| **Giảng viên hướng dẫn :** | | **Ths Lê Thị Anh** |
| **Hà Nội, tháng 2 năm 2023** | | |

# LỜI CẢM ƠN

*Để hoàn thành tốt bài tập lớn này, chúng em - Nhóm 09 xin bày tỏ lòng biết ơn đến Ths Lê Thị Anh, đã tận tình hướng dẫn, giúp đỡ và chỉ bảo nhóm trong suốt thời gian thực hiện đề tài. Cô luôn sẵn lòng lắng nghe những thắc mắc của nhóm và giải đáp một cách rõ ràng và chi tiết, luôn hướng cho chúng em một hướng đi đúng đắn và có nhiều cơ hội rộng mở nhất. Nhờ vậy, nhóm 09 chúng em đã hiểu rõ hơn về chủ đề nghiên cứu và có thể tiến hành thực hiện bài tập một cách hiệu quả. Với vốn kiến thức được tiếp thu trong thời gian học không chỉ là nền tảng cho quá trình làm đề tài mà còn là hành trang quý báu để chúng em tiếp tục quá trình học tập và làm việc với tâm thế vững chắc và tự tin. Bên cạnh đó, nhóm 09 cũng xin chân thành cảm ơn đến các bạn trong lớp, trong quá trình học đã giúp đỡ và tạo điều kiện thuận lợi để nhóm hoàn thành bài báo cáo. Và cuối cùng chúng em xin kính chúc cô sức khỏe và thành công trong sự nghiệp. Chúc các bạn luôn dồi dào sức khỏe, đạt được nhiều thành công trong quá trình học tập.*

#### Chúng em xin chân thành cảm ơn!

Nhóm thực hiện

*Nhóm 09*

**MỤC LỤC**

[LỜI CẢM ƠN 2](#_Toc159859842)

[LỜI NÓI ĐẦU 4](#_Toc159859843)

[CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI 5](#_Toc159859844)

[1.1. Giới thiệu về lỗ hổng bảo mật 5](#_Toc159859845)

[1.1.1 Định nghĩa về lỗ hổng bảo mật 5](#_Toc159859846)

[1.1.2 Tác động tiêu cực 5](#_Toc159859847)

[1.2. Mục tiêu, đối tượng và phạm vi nghiên cứu 6](#_Toc159859848)

[1.3. Phương pháp nghiên cứu 8](#_Toc159859849)

[1.4. Cấu trúc báo cáo 8](#_Toc159859850)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT VỀ AN TOÀN THÔNG TIN TRÊN NỀN TẢNG WEB/APP/OS 9](#_Toc159859851)

[2.1. Website 9](#_Toc159859852)

[2.2.APP 11](#_Toc159859853)

[2.3. OS 12](#_Toc159859854)

[2.4 Các khái niệm cơ bản như HTTP, HTTPS, cookies, và sessions. 13](#_Toc159859855)

[2.5 Mã hóa và chữ ký số 16](#_Toc159859856)

[CHƯƠNG 3: TỔNG QUAN VỀ CÁC LỖ HỔNG BẢO MẬT WEB/APPOS 17](#_Toc159859857)

[3.1. Các lỗ hổng bảo mật Web 17](#_Toc159859858)

[3.2. Lỗ hổng bảo mật App 28](#_Toc159859859)

[3.3.Lỗ hổng bảo mật OS 31](#_Toc159859860)

[CHƯƠNG 4: THIẾT KẾ GIAO DIỆN VÀ CÁC CHỨC NĂNG 34](#_Toc159859861)

[4.1. Mô hình hóa chức năng 34](#_Toc159859862)

[4.2. Giao diện người dung 35](#_Toc159859863)

[4.3. Tổng kết lại các biện pháp bảo mật 38](#_Toc159859864)

[KẾT LUẬN 40](#_Toc159859865)

[TÀI LIỆU THAM KHẢO 41](#_Toc159859866)

[Tiếng Việt 41](#_Toc159859867)

[Tiếng Anh 41](#_Toc159859868)

# LỜI NÓI ĐẦU

Trong thời đại mà sự kết nối qua mạng Internet trở thành phổ biến, bảo mật web trở thành một trong những vấn đề quan trọng nhất đối với cả cá nhân và doanh nghiệp. Cùng với sự phát triển nhanh chóng của các ứng dụng web và dịch vụ trực tuyến, các mối đe dọa liên quan đến an ninh trực tuyến cũng ngày càng trở nên phức tạp và nguy hiểm.

Báo cáo này đưa ra một cái nhìn tổng quan về các vấn đề bảo mật web đương đại, tập trung vào việc nghiên cứu và phân tích các mô hình tấn công phổ biến, cũng như những biện pháp bảo mật mà cộng đồng kỹ thuật đang tích cực thực hiện. Chúng ta sẽ khám phá những thách thức mới mẻ mà các nhà phát triển và quản trị hệ thống đối mặt, đồng thời đề xuất những giải pháp sáng tạo để bảo vệ thông tin quan trọng và đảm bảo tính an toàn của ứng dụng web.

Đề tài này không chỉ nhằm mục đích cải thiện hiểu biết của chúng ta về bảo mật web, mà còn nhấn mạnh tầm quan trọng của việc áp dụng các biện pháp bảo mật hiệu quả trong môi trường số ngày nay. Chúng ta hãy bắt đầu hành trình khám phá sâu rộng vào thế giới phức tạp này, nơi mà sự an toàn của thông tin trở thành trách nhiệm chung của chúng ta.

# CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI

### Giới thiệu về lỗ hổng bảo mật

### Định nghĩa về lỗ hổng bảo mật

* Lỗ hổng bảo mật (ví dụ: security vulnerability) là một điểm yếu hoặc một thiếu sót trong hệ thống bảo mật, mạng, ứng dụng, hoặc quy trình, tạo ra khả năng cho kẻ tấn công để xâm nhập, tận dụng, hoặc phá vỡ hệ thống một cách trái phép. Điều này có thể dẫn đến mất mát, tiết lộ, hoặc sửa đổi thông tin không mong muốn, và có thể ảnh hưởng đến tính toàn vẹn, sẵn sàng, và an ninh của hệ thống.
* Lỗ hổng bảo mật có thể xuất hiện từ nhiều nguồn khác nhau, bao gồm lỗi phần mềm, thiết kế hệ thống không an toàn, cấu hình không chính xác, hoặc yếu tố con người như việc sử dụng mật khẩu yếu. Việc tận dụng lỗ hổng bảo mật có thể dẫn đến các cuộc tấn công như xâm nhập mạng, truy cập trái phép vào dữ liệu, hoặc thậm chí kiểm soát toàn bộ hệ thống.
* Để đối phó với lỗ hổng bảo mật, các tổ chức thường xuyên thực hiện kiểm thử bảo mật, theo dõi các thông báo bảo mật, và triển khai biện pháp bảo mật như cập nhật phần mềm, mã hóa dữ liệu, và quản lý quyền truy cập.

### Tác động tiêu cực

* *Mất Dữ Liệu:*
* Mô Tả: Kẻ tấn công có thể thu thập, xóa, hoặc sửa đổi dữ liệu một cách trái phép, dẫn đến mất mát thông tin quan trọng của cá nhân hoặc doanh nghiệp.
* Hậu Quả: Mất dữ liệu quan trọng có thể gây tổn thất tài chính và ảnh hưởng đến uy tín của tổ chức.
* *Tiết Lộ Thông Tin Cá Nhân:*
* Mô Tả: Lỗ hổng bảo mật có thể dẫn đến việc tiết lộ thông tin cá nhân của người dùng, như tên, địa chỉ, thông tin tài khoản ngân hàng, hoặc thông tin thẻ tín dụng.
* Hậu Quả: Việc tiết lộ thông tin cá nhân có thể dẫn đến mức độ nguy cơ cao về lạm dụng thông tin và gây hậu quả nặng nề cho cá nhân.
* *Thiệt Hại Tài Chính:*
* Mô Tả: Các cuộc tấn công có thể dẫn đến mất mát tài chính, bao gồm chi phí phục hồi, mất doanh thu, và chi phí pháp lý.
* Hậu Quả: Thiệt hại tài chính có thể ảnh hưởng đến khả năng hoạt động của doanh nghiệp và có thể làm giảm giá trị thị trường.
* *Tình Toàn Vẹn Dữ Liệu:*
* Mô Tả: Lỗ hổng bảo mật có thể dẫn đến sự sửa đổi không được phép hoặc làm thay đổi thông tin trong hệ thống, làm giả mạo dữ liệu.
* Hậu Quả: Mất tính toàn vẹn của dữ liệu có thể tạo ra thông tin giả mạo, gây hậu quả nghiêm trọng cho quyết định và hoạt động của tổ chức.
* *Mất Kiểm Soát Hệ Thống:*
* Mô Tả: Kẻ tấn công có thể kiểm soát và thậm chí tận dụng hệ thống hoặc mạng, có thể dẫn đến tình trạng mất kiểm soát toàn bộ.
* Hậu Quả: Mất kiểm soát hệ thống có thể làm tăng nguy cơ mất dữ liệu, sự cô lập của hệ thống, và thậm chí làm ảnh hưởng đến an toàn của người dùng.
* *Mối Đe Dọa Cho An Ninh Toàn Cầu:*
* Mô Tả: Lỗ hổng bảo mật trong một hệ thống có thể trở thành một mối đe dọa cho an ninh toàn cầu khi nó có thể được tận dụng để tấn công hệ thống khác hoặc trở thành một phần của mạng botnet.
* Hậu Quả: Tình trạng lan truyền có thể làm tăng đáng kể nguy cơ cho cả hệ thống và mạng Internet nói chung.

### Mục tiêu, đối tượng và phạm vi nghiên cứu

- Mục tiêu nghiên cứu: Tìm hiểu, phân tích các lỗ hổng bảo mật thường gặp và đề xuất các giải pháp để bảo vệ web/app/os khỏi các nguy cơ và lỗ hổng bảo mật đó.

- Đối tượng nghiên cứu:

• Web: Nghiên cứu có thể tập trung vào các ứng dụng web, cơ sở dữ liệu, giao thức truyền thông, các thành phần phía máy chủ và phía người dùng, cũng như các hệ thống quản lý nội dung (CMS) và các nền tảng thương mại điện tử.

• App: Nghiên cứu có thể tập trung vào ứng dụng di động trên các nền tảng như iOS và Android, bao gồm cả ứng dụng di động và ứng dụng web di động.

• OS: Nghiên cứu có thể tập trung vào các hệ điều hành máy tính, hệ điều hành di động và các thành phần liên quan như quyền truy cập hệ thống, quản lý tài khoản và cơ chế bảo mật.

- Phạm vi nghiên cứu: Các lỗ hổng phổ biến XSS ,SQL injection,…

### Phương pháp nghiên cứu

- Phương pháp thu thập thông tin: Thu thập các tài liệu liên quan đến các lỗ hổng bảo mật thường gặp đồng thời quan sát cách thức hoạt động của các lỗ hổng bảo mật.

- Phương pháp phân tích thông tin: Dựa trên thông tin thu thập được, phân tích cách thức mà các tin tặc khai thác các lỗ hổng bảo mật đó.

### Cấu trúc báo cáo

Nội dung bản báo cáo sẽ bao gồm 3 phần sau:

* + - Cơ sở lý thuyết về an toàn thông tin trên nền tảng Web/App/Os, bao gồm các nội dung:

+ Khái niệm về Web/App/Os.

+ Mô tả cách thức hoạt động của Web/App/Os.

+ Các khái niệm HTTP, HTTPS, cookies và sessions.

+ Mã hóa và chữ ký số.

* + - TỔNG QUAN VỀ CÁC LỖ HỔNG BẢO MẬT Web,App,OS:

+ Định nghĩa về lỗ hổng bảo mật

+ Cách nhận biết

+ Tác động tiêu cực

+ Các giải pháp

* + - Thiết kế giao diện và các chức năng

+ Biểu đồ use case

+Mô tả chi tiết

+Giao diện

## CHƯƠNG 2: CƠ SỞ LÝ THUYẾT VỀ AN TOÀN THÔNG TIN TRÊN NỀN TẢNG WEB/APP/OS

### Website

#### 2.1.1.Khái niệm về Website

* Website là tập hợp những trang (page) con, mà mỗi trang con là một siêu văn bản (hypertext) bao gồm văn bản, hình ảnh, video và có thể liên kết từ trang này sang trang khác. Các ví dụ là các website wikipedia.org, google.com và amazon.com.
* Tất cả các trang web có thể truy cập công khai đều tạo thành World Wide Web. Cũng có những trang web riêng tư chỉ có thể được truy cập trên mạng riêng, chẳng hạn như trang web nội bộ của công ty dành cho nhân viên của công ty.
* Các trang web thường dành riêng cho một chủ đề hoặc mục đích cụ thể, chẳng hạn như tin tức, giáo dục, thương mại, giải trí hoặc mạng xã hội. Siêu liên kết giữa các trang web hướng dẫn điều hướng của trang web, thường bắt đầu với trang chủ.
* Người dùng có thể truy cập các trang web trên nhiều loại thiết bị, bao gồm máy tính để bàn, máy tính xách tay, máy tính bảng và điện thoại thông minh. Ứng dụng được sử dụng trên các thiết bị này được gọi là trình duyệt web.
* Hiện nay, để một website có thể vận hành trên môi trường Word Wide Web, cần bắt buộc có 3 phần chính:

+ Tên miền (Domain): là tên riêng và duy nhất của website

+ Hosting (Hosting Share, máy chủ, vps...): là máy chủ chứa mã nguồn, và dữ liệu của trang web cũng như các tập tin liên quan

+ Mã nguồn web (Source code): là các tệp tin html, xhtml,.. hoặc một bộ code được lập trình bằng các ngôn ngữ asp.net, PHP, java, Ruby on Rails....

#### 2.1.2Mô tả hoạt động của một Website

* Từ máy client gõ địa chỉ URL của trang web vào thanh địa chỉ của web browser, khi đó web browser sẽ dựa vào tên domain để tìm kiếm IP của web server (bằng cách dò trên DNS Server)
* Khi có IP, web browser sẽ gửi yêu cầu đến web server để lấy nội dung trang web
* Web server sẽ xử lý yêu cầu của client: lấy nội dung của website mà client yêu cầu để gửi client.
* Khi nhận được nội dung trang web thì web browser sẽ hiển thị lên cho người dùng.
* Trình duyệt web sẽ tách địa chỉ website ra làm ba phần:
* Giao thức (http)
* Máy chủ tên miền [(www.webname.com)](http://www.webname.com/)
* Tên tệp (file.htm)
* Trình duyệt web liên hệ với máy chủ tên miền (DNS server) để phân giải tên miền “[www.webname.com](http://www.webname.com/)” thành địa chỉ IP của web server
* Trình duyệt sẽ gửi tiếp 1 kết nối tới máy chủ địa chỉ IP tương ứng cổng 80.
* Dựa trên giao thức HTTP, trình duyệt gửi yêu cầu đến máy chủ, yêu cầu tệp: <http://webname.com/file.htm>
* Máy chủ sẽ gửi đoạn text dạng html đến trình duyệt của web client. Trình duyệt đọc các thẻ html, định dạng trang web và kết xuất ra màn hình.

### 2.2.APP

#### 2.2.1 Khái niệm về App

- An toàn thông tin trên nền tảng ứng dụng (app) là việc đảm bảo rằng dữ liệu và hoạt động của ứng dụng được bảo vệ khỏi các mối đe dọa an ninh mạng và các cuộc tấn công. Điều này bao gồm các biện pháp để ngăn chặn truy cập trái phép, bảo vệ thông tin cá nhân của người dùng, và đảm bảo tính toàn vẹn của dữ liệu.

#### 2.2.2 Mô tả hoạt động của một App

- Phát hiện lỗ hổng:

+ Kiểm thử bảo mật

+Kiểm tra mã nguồn

+Sử dụng công cụ quét lỗ hổng

+Phân tích bản vá và cập nhật

+Sử dụng hệ thống giám sát và phát hiện xâm nhập

- Kiểm tra mã nguồn:

+Xem xét các lỗ hổng phổ biến

+Sử dụng công cụ tự động hóa

+Kiểm tra mã nguồn bằng tay

+Phân tích mã nguồn mở

+Phân tích các phần mềm và thư viện bên thứ ba

- Sử dụng công cụ quét:

+Chạy công cụ quét lỗ hổng

+Theo dõi tiến trình quét

+Phân tích kết quả

+Thực hiện biện pháp sửa chữa

- Phân tích bản vá và cập nhật:

+Tìm hiểu về bản vá

+Kiểm tra tính khả thi và tương thích

+Triển khai bản vá

+Kiểm tra lại và kiểm tra đánh giá

+Cập nhật tài liệu và thông báo

+Kiểm tra lại và kiểm tra đánh giá lỗ hổng

-Sử dụng hệ thống giám sát và phát hiện xâm nhập:

+Triển khai giám sát liên tục

+Cấu hình quy tắc và cảnh báo

+Theo dõi lưu lượng mạng và hoạt động hệ thống

+Phân tích cảnh báo và hành động phản ứng

+Nâng cấp và tinh chỉnh

+Kiểm tra và đánh giá hiệu suất

### 2.3. OS

#### 2.3.1 Khái niệm về Os

- An toàn thông tin trên hệ điều hành (Operating System - OS) là việc đảm bảo rằng hệ điều hành và các ứng dụng chạy trên nó được bảo vệ khỏi các mối đe dọa an ninh mạng và các cuộc tấn công. Điều này bao gồm các biện pháp để ngăn chặn truy cập trái phép, bảo vệ dữ liệu và thông tin cá nhân của người dùng, và đảm bảo tính toàn vẹn của hệ thống.

#### 2.3.2 Mô tả hoạt động của một Os

-Xác thực người dùng:

* Phương tiện xác thực
* Tính an toàn của mật khẩu
* Quản lý danh sách truy cập

-Kiểm soát truy cập:

* Kiểm soát dựa trên vai trò
* Kiểm soát Dựa trên Điều Kiện
* Quản lý Quyền Cụ Thể

-Mã hóa:

* Mã hóa dữ liệu đang dữ trữ
* Mã hóa dữ liệu đang truyền
* Mã hóa tích hợp

-Tường lửa:

* Kiểm soát lưu lượng
* Phân tích gói tin

-Hệ thống phát hiện xâm nhập

* Kiểm soát và giám sát các hoạt động
* Phân tích hành vi và mẫu
* Phản ứng và báo cáo
* Hệ thống phát hiện xâm nhập mang(NIDS)
* Hệ thống phát hiện xâm nhập máy chủ(HIDS)

### 2.4 Các khái niệm cơ bản như HTTP, HTTPS, cookies, và sessions.

#### 2.4.1 Giao thức HTTP

* + - * HTTP (Hypertext Transfer Protocol) là giao thức truyền tải siêu văn bản. Đây là giao thức tiêu chuẩn cho World Wide Web (www) để truyền tải dữ liệu dưới dạng văn bản, âm thanh, hình ảnh, video từ Web Server tới trình duyệt web của người dùng và ngược lại.
      * HTTP là một giao thức ứng dụng của bộ giao thức TCP/IP (các giao thức nền tảng cho Internet). Bộ giao thức TCP/IP là một bộ các giao thức truyền thông cài đặt chồng giao thức mà Internet và hầu hết các mạng máy tính thương mại đang chạy trên đó. Bộ giao thức này được đặt theo tên hai giao thức chính là TCP (Transmission Control Protocol – Giao thức điều khiển truyền vận) và IP (Internet Protocol – Giao thức Internet).
      * HTTP hoạt động theo mô hình Client (máy khách) – Server (máy chủ). Việc truy cập website được tiến hành dựa trên các giao tiếp giữa 2 đối tượng trên. Khi bạn truy cập một trang web qua giao thức HTTP, trình duyệt sẽ thực hiện các phiên kết nối đến server của trang web đó thông qua địa chỉ IP do hệ thống phân giải tên miền DNS cung cấp. Máy chủ sau khi nhận lệnh, sẽ trả về lệnh tương ứng giúp hiển thị website, bao gồm các nội dung như: văn bản, ảnh, video, âm thanh,…
      * Trong quá trình kết nối và trao đổi thông tin, trình duyệt của bạn sẽ mặc nhiên thừa nhận địa chỉ IP đó đến từ server của chính website mà bạn muốn truy cập mà không hề có biện pháp xác thực nào. Các thông tin được gửi đi qua giao thức HTTP (bao gồm địa chỉ IP, các thông tin mà bạn nhập vào website…) cũng không hề được mã hóa và bảo mật. Đây chính là kẽ hở mà nhiều hacker đã lợi dụng để đánh cắp thông tin người dùng, thường được gọi là tấn công sniffing.

#### 2.4.2Giao thức HTTPS

* + - * HTTPS (Hypertext Transfer Protocol Secure) là giao thức truyền tải siêu văn bản an toàn. Thực chất, đây chính là giao thức HTTP nhưng tích hợp thêm Chứng chỉ bảo mật SSL nhằm mã hóa các thông điệp giao tiếp để tăng tính bảo mật. Có thể hiểu, HTTPS là phiên bản HTTP an toàn, bảo mật hơn.
      * HTTPS hoạt động tương tự như HTTP, tuy nhiên được bổ sung thêm chứng chỉ SSL (Secure Sockets Layer – tầng ổ bảo mật) hoặc TLS (Transport Layer Security – bảo mật tầng truyền tải). Hiện tại, đây là các tiêu chuẩn bảo mật hàng đầu cho hàng triệu website trên toàn thế giới.
      * Cả SSL và TLS đều sử dụng hệ thống PKI (Public Key Infrastructure -hạ tầng khóa công khai) không đối xứng. Hệ thống này sử dụng hai “khóa” để mã hóa thông tin liên lạc, “khóa công khai” (public key) và “khóa riêng” (private key). Bất cứ thứ gì được mã hóa bằng khóa công khai chỉ có thể được giải mã bởi khóa riêng và ngược lại. Các tiêu chuẩn này đảm bảo các nội dung sẽ được mã hóa trước khi truyền đi, và giải mã khi nhận. Điều này khiến hacker dù có chen ngang lấy được thông tin cũng không thể “hiểu” được thông tin đó.

#### 2.4.3 Cookie

* + - * Cookie được dùng để lưu những thông tin tạm thời. Những tập tin cookie sẽ được truyền từ server tới browser và được lưu trữ trên máy tính của bạn khi bạn truy cập vào ứng dụng.
      * Cookie thường được tạo ra khi người dùng truy cập một website, cookie sẽ ghi nhớ những thông tin như tên đăng nhập, mật khẩu, các tùy chọn do người dùng lựa chọn đi kèm. Các thông tin này được lưu trong máy tính để nhận biết người dùng khi truy cập vào một trang web.
      * Khi người dùng truy cập đến một trang web có sử dụng cookie, web server của trang đó sẽ tự động gửi cookie đến máy tính của người dùng. Khi truy cập đến các trang web sử dụng được cookie đã lưu, những cookie này tự động gửi thông tin của người dùng về cho chủ của nó (người tạo ra cookie). Tuy nhiên những thông tin do cookie ghi nhận không được tiết lộ rộng rãi, chỉ có website chứa cookie mới có thể xem được những thông tin này.
      * Mỗi cookie thường có khoảng thời gian timeout nhất định do lập trình viên xác định trước. Những thông tin được lưu vào cookie ví dụ như thông tin đăng nhập, thao tác người dùng, tần suất ghé thăm website, thời gian truy cập… Tất cả chúng đều là những thông tin mang tính tạm thời và được lưu trong 1 khoảng thời gian.

#### 2.4.4.Session

* + - * Session là một phiên làm việc là một khái niệm phổ biến được dùng trong lập trình web có kết nối với database. Đặc biệt các chức năng như đăng nhập, đăng xuất người dùng sẽ khó có thể thực hiện được nếu không sử dụng session.
      * Một session bắt đầu khi client gửi request đến server, nó tồn tại xuyên suốt từ trang này đến trang khác trong ứng dụng web và chỉ kết thúc khi hết thời gian timeout hoặc khi bạn đóng ứng dụng. Giá trị của session sẽ được lưu trong một file trên server. Ví dụ khi bạn đăng nhập vào một trang web và đăng nhập với tài khoản đã đăng ký trước đó. Server sau khi xác thực được thông tin bạn cung cấp là đúng thì nó sẽ sinh ra một tập tin chứa dữ liệu cần lưu trữ của người dùng.

### 2.5 Mã hóa và chữ ký số

#### 2.5.1 Mã hóa

* + - * Mã hóa là quá trình chuyển đổi thông tin từ dạng ban đầu thành dạng không đọc được (ciphertext) bằng cách sử dụng thuật toán và một khóa (key). Mục tiêu của việc mã hóa là bảo vệ thông tin trước khi nó được truyền tải hoặc lưu trữ trên các thiết bị.
      * Có hai loại mã hóa chính:

+ Mã hóa đối xứng (Symmetric Encryption)

+ Mã hóa không đối xứng (Asymmetric Encryption)

* + - * Mã hóa đảm bảo rằng dữ liệu truyền tải qua mạng hoặc được lưu trữ trên thiết bị sẽ không bị đánh cắp hoặc sử dụng sai mục đích. Khi dữ liệu được mã hóa, người nhìn thứ ba không thể đọc được nó mà không có khóa phù hợp.
      * Mã hóa đóng vai trò quan trọng trong bảo vệ thông tin cá nhân, tài khoản ngân hàng, giao dịch trực tuyến và nhiều loại dữ liệu quan trọng khác trên internet và trong hệ thống thông tin.

#### 2.5.2. Chữ ký số

* + - * Chữ ký số (Digital Signature) là một phương pháp xác thực tính xác thực của thông tin hoặc tài liệu trên internet. Nó giống như chữ ký truyền thống trên giấy tờ nhưng được tạo ra điện tử, cho phép xác định xem liệu một tài liệu hay thông tin đã được xác minh và không bị thay đổi sau khi được ký.
      * Quá trình tạo chữ ký số bao gồm các bước sau:

+ Tạo mã băm

+ Mã hóa chữ ký

+ Xác minh chữ ký

* + - * Chữ ký số đảm bảo tính xác thực và toàn vẹn của thông tin trên internet, giúp ngăn chặn sửa đổi trái phép, giả mạo, và xác minh nguồn gốc của thông tin. Nó thường được sử dụng trong giao dịch tài chính trực tuyến, email, các tài liệu hợp đồng kỹ thuật số, và trong các quy trình xác thực trên internet.

## CHƯƠNG 3: TỔNG QUAN VỀ CÁC LỖ HỔNG BẢO MẬT WEB/APPOS

### Các lỗ hổng bảo mật Web

#### Sensitive data exposure (Rò rỉ dữ liệu nhạy cảm)

* + - 1. **Định nghĩa**
* Lỗ hổng này thuộc về khía cạnh crypto và tài nguyên. Dữ liệu nhạy cảm phải được mã hóa mọi lúc, bao gồm cả khi gửi đi và khi lưu trữ – không được phép có ngoại lệ. Thông tin thẻ tín dụng và mật khẩu người dùng không bao giờ được gửi đi hoặc được lưu trữ không được mã hóa. Rõ ràng thuật toán mã hóa và hashing không phải là một cách bảo mật yếu. Ngoài ra, các tiêu chuẩn an ninh web đề nghị sử dụng AES (256 bit trở lên) và RSA (2048 bit trở lên).

**3.1.1.2. Cách nhận biết**

* + - Kiểm tra kỹ thuật mã hóa dữ liệu: Kiểm tra xem liệu dữ liệu nhạy cảm như thông tin cá nhân, mật khẩu, thông tin tài chính có được mã hóa không. Nếu dữ liệu được lưu trữ hoặc truyền đi mà không được mã hóa, đó có thể là một dấu hiệu của lỗ hổng rò rỉ dữ liệu nhạy cảm.
    - Kiểm tra các cấu hình bảo mật: Kiểm tra các cấu hình bảo mật của ứng dụng hoặc hệ thống để đảm bảo rằng chỉ có các người dùng được ủy quyền mới có thể truy cập vào dữ liệu nhạy cảm. Nếu cấu hình không đúng, có thể dẫn đến việc rò rỉ dữ liệu.
    - Xem xét quản lý phiên và xác thực: Đảm bảo rằng quản lý phiên và các quá trình xác thực được thực hiện một cách an toàn và hiệu quả. Rò rỉ dữ liệu nhạy cảm có thể xảy ra nếu có lỗ hổng trong cách quản lý phiên hoặc xác thực người dùng.
    - Kiểm tra log hệ thống: Phân tích log hệ thống để xem có bất kỳ hoạt động không bình thường nào như truy cập không được phép vào dữ liệu nhạy cảm. Có thể sử dụng các công cụ giám sát log để tự động phát hiện các hoạt động đáng ngờ.
    - Thực hiện kiểm thử bảo mật: Sử dụng các kỹ thuật kiểm thử bảo mật như kiểm thử rò rỉ thông tin (data leakage testing) để kiểm tra xem liệu dữ liệu nhạy cảm có thể bị rò rỉ không. Các kỹ thuật này có thể bao gồm kiểm tra đầu ra, kiểm tra giao diện người dùng, và kiểm tra các giao thức mạng.
    - Xem xét mã nguồn ứng dụng: Kiểm tra mã nguồn của ứng dụng để tìm kiếm các lỗ hổng bảo mật như việc lưu trữ dữ liệu nhạy cảm mà không được mã hóa hoặc việc truyền dữ liệu qua kênh không an toàn.
    - Kiểm tra bảo mật mạng: Kiểm tra các kết nối mạng để đảm bảo rằng dữ liệu nhạy cảm không được truyền đi qua các kênh không an toàn như kết nối không mã hóa hoặc kết nối không đáng tin cậy.

**3.1.1.3 Cách ngăn chặn**

* + - Mã hóa dữ liệu: Mã hóa dữ liệu nhạy cảm khi lưu trữ và truyền đi. Sử dụng mã hóa mạnh như SSL/TLS để bảo vệ dữ liệu khi truyền đi trên mạng, và mã hóa dữ liệu khi lưu trữ trên cơ sở dữ liệu hoặc hệ thống tệp tin.
    - Xác thực mạnh mẽ: Sử dụng các phương pháp xác thực mạnh mẽ để đảm bảo chỉ có người dùng được ủy quyền mới có thể truy cập vào dữ liệu nhạy cảm. Sử dụng cơ chế xác thực hai yếu tố hoặc xác thực đa bước để tăng cường bảo mật.
    - Quản lý phiên an toàn: Đảm bảo rằng quản lý phiên được thực hiện một cách an toàn và hiệu quả. Sử dụng các phương pháp như token phiên và giới hạn thời gian phiên để ngăn chặn việc truy cập không cần thiết vào dữ liệu.
    - Chặn truy cập không cần thiết: Đảm bảo rằng dữ liệu nhạy cảm chỉ được truy cập bởi người dùng và quy trình cần thiết. Áp dụng các cơ chế kiểm soát truy cập để ngăn chặn truy cập không cần thiết từ người dùng không ủy quyền hoặc từ bên ngoài.
    - Kiểm tra cấu hình bảo mật: Kiểm tra và đảm bảo rằng cấu hình bảo mật của ứng dụng và hệ thống đang hoạt động đúng cách. Tắt các tính năng không cần thiết, thiết lập quyền truy cập đúng cho người dùng và quy trình, và thực hiện các biện pháp bảo mật khác như CSP (Content Security Policy).
    - Giảm thiểu dữ liệu nhạy cảm: Tránh lưu trữ dữ liệu nhạy cảm mà không cần thiết. Chỉ lưu trữ dữ liệu nhạy cảm khi thực sự cần thiết và xóa bỏ dữ liệu cũ khi không cần thiết nữa.
    - Kiểm tra bảo mật định kỳ: Thực hiện kiểm tra bảo mật định kỳ để xác định và khắc phục các lỗ hổng bảo mật liên quan đến rò rỉ dữ liệu nhạy cảm. Sử dụng các công cụ kiểm thử bảo mật để kiểm tra ứng dụng và hệ thống của bạn từ các góc độ khác nhau.
    - Đào tạo nhân viên về bảo mật: Đào tạo nhân viên của bạn về các nguy cơ rò rỉ dữ liệu nhạy cảm và cách ngăn chặn chúng. Đảm bảo rằng nhân viên có hiểu biết về các quy tắc bảo mật và thực hành an toàn khi làm việc với dữ liệu nhạy cảm.

#### Missing function level access control (lỗi phân quyền)

**3.1.2.1. Định nghĩa**

-Đây chỉ là sai sót trong vấn đề phân quyền. Nó có nghĩa là khi một hàm được gọi trên máy chủ, quá trình phân quyền không chính xác. Các nhà phát triển dựa vào thực tế là phía máy chủ tạo ra giao diện người dùng và họ nghĩ rằng khách hàng không thể truy cập các chức năng nếu không được cung cấp bởi máy chủ.

Tuy nhiên, kẻ tấn công luôn có thể yêu cầu các chức năng “ẩn” và sẽ không bị cản trở bởi việc giao diện người dùng không cho phép thực hiện các chức năng này. Hãy tưởng tượng trong giao diện người dùng chỉ có bảng điều khiển/admin và nút nếu người dùng thực sự là quản trị viên. Không có gì ngăn cản kẻ tấn công phát hiện ra những tính năng này và lạm dụng nó nếu không phân quyền.

* + - 1. **Cách nhận biết**
    - **Thực hiện kiểm thử tính năng không phân quyền:** Kiểm tra các tính năng của ứng dụng mà không đăng nhập hoặc sử dụng một tài khoản có quyền truy cập thấp. Nếu bạn có thể truy cập vào các tính năng quản trị hoặc tính năng cần đặc quyền mà không cần phải xác thực, có thể ứng dụng của bạn đang mắc lỗi phân quyền.
    - **Sử dụng các công cụ kiểm thử tự động:** Có các công cụ kiểm thử tự động có thể tự động phát hiện lỗ hổng "Missing Function Level Access Control" bằng cách kiểm tra các yêu cầu và phản hồi của ứng dụng. Các công cụ như OWASP ZAP hoặc Burp Suite có thể hỗ trợ bạn trong việc phát hiện lỗ hổng này.
    - **Phân tích log hoạt động của người dùng:** Xem xét các log hoạt động của người dùng để tìm kiếm các hoạt động không bình thường hoặc có dấu hiệu của việc truy cập vào các tính năng hoặc chức năng không được phép.
    - **Kiểm tra mã nguồn ứng dụng:** Kiểm tra mã nguồn của ứng dụng để tìm kiếm bất kỳ điều kiện logic nào có thể cho phép truy cập không đáng phải được phép vào các tính năng hoặc chức năng.
    - **Kiểm tra các yêu cầu HTTP:** Kiểm tra các yêu cầu HTTP để xem xét liệu các thông tin đi kèm có đủ để xác định quyền truy cập của người dùng không. Nếu các yêu cầu không chứa thông tin về quyền truy cập, có thể có lỗi phân quyền.
    - **Kiểm tra phản hồi HTTP:** Kiểm tra các phản hồi HTTP từ ứng dụng để xem xét liệu chúng có chứa thông tin nhạy cảm hoặc các chức năng không được phép không. Nếu người dùng có thể thấy thông tin hoặc truy cập vào các chức năng không đáng phải được phép, có thể ứng dụng đang mắc lỗi phân quyền.

**3.1.2.3 .Cách ngăn chặn**

* Xác thực mạnh mẽ: Thực hiện xác thực mạnh mẽ để đảm bảo rằng người dùng chỉ có thể truy cập vào các tính năng hoặc chức năng mà họ được phép. Sử dụng các biện pháp như xác thực hai yếu tố hoặc xác thực đa bước để tăng cường bảo mật.
* Quản lý phiên an toàn: Đảm bảo rằng quản lý phiên được thực hiện một cách an toàn và hiệu quả. Sử dụng các token phiên và giới hạn thời gian phiên để ngăn chặn việc truy cập không cần thiết vào các tính năng hoặc chức năng.
* Kiểm tra quyền truy cập: Thực hiện kiểm tra quyền truy cập trước khi cho phép người dùng truy cập vào các tính năng hoặc chức năng. Đảm bảo rằng các quyền được xác định chính xác và chỉ cho phép truy cập nếu người dùng có quyền tương ứng.
* Kiểm tra cấu hình bảo mật: Kiểm tra và đảm bảo rằng cấu hình bảo mật của ứng dụng và hệ thống đang hoạt động đúng cách. Tắt các tính năng không cần thiết, thiết lập quyền truy cập đúng cho người dùng và quy trình, và thực hiện các biện pháp bảo mật khác như CSP (Content Security Policy).
* Kiểm tra mã nguồn ứng dụng: Kiểm tra mã nguồn của ứng dụng để tìm kiếm bất kỳ điều kiện logic nào có thể cho phép truy cập không đáng phải được phép vào các tính năng hoặc chức năng. Sửa lỗi logic và thực hiện kiểm tra tự động để đảm bảo tính liên tục của quá trình kiểm tra.
* Kiểm tra thủ công và tự động: Thực hiện kiểm tra thủ công và tự động để đảm bảo tính đầy đủ và hiệu quả của việc kiểm tra quyền truy cập. Sử dụng các công cụ kiểm thử bảo mật để tự động phát hiện lỗ hổng và thực hiện kiểm tra thủ công để kiểm tra các tình huống phức tạp hơn.
* Đào tạo nhân viên về bảo mật: Đào tạo nhân viên của bạn về các nguy cơ của lỗ hổng "Missing Function Level Access Control" và cách ngăn chặn chúng. Đảm bảo rằng nhân viên hiểu biết về các quy tắc bảo mật và thực hành an toàn khi phát triển và triển khai ứng dụng.
  + 1. **SQL Injection (SQLi)**

**3.1.3.1 Định nghĩa**

SQL Injection là một kỹ thuật tấn công mà kẻ tấn công cố gắng chèn các đoạn mã SQL độc hại vào các truy vấn SQL của ứng dụng web. Khi thực thi, các đoạn mã này có thể thay đổi cấu trúc của câu lệnh SQL, dẫn đến việc truy cập, thay đổi hoặc xóa dữ liệu từ cơ sở dữ liệu. Các lỗ hổng SQLi thường xuất hiện khi ứng dụng không kiểm soát đầu vào người dùng đúng cách.

**3.1.3.2. Cách nhận biết**

* Ký tự đặc biệt trong đầu vào: Sự xuất hiện của các ký tự như ', ;, --, UNION, có thể là dấu hiệu của SQL Injection.
* Sự thay đổi trong dữ liệu: Các biến đổi bất thường trong dữ liệu cơ sở dữ liệu, chẳng hạn như sự xuất hiện của thông tin không mong muốn hoặc thay đổi không rõ nguồn gốc.
* Lỗi SQL xuất hiện: Nếu ứng dụng không xử lý lỗi SQL một cách an toàn, lỗi có thể xuất hiện trực tiếp trên giao diện người dùng.

**3.1.3.3.Cách ngăn chặn**

* Sử dụng Prepared Statements hoặc Parametrized Queries: Thay vì tạo câu lệnh SQL bằng cách ghép chuỗi, sử dụng cơ chế Prepared Statements để truyền tham số an toàn vào câu lệnh SQL.
* Kiểm tra và xử lý đầu vào người dùng: Loại bỏ hoặc mã hóa các ký tự đặc biệt trong dữ liệu đầu vào để ngăn chặn việc chèn mã độc hại.
* Nguyên tắc của Least Privilege: Đảm bảo rằng tài khoản cơ sở dữ liệu mà ứng dụng sử dụng có đặc quyền tối thiểu cần thiết để thực hiện các nhiệm vụ, giảm thiểu ảnh hưởng của một cuộc tấn công SQLi.

**3.1.4 Cross-Site Scripting (XSS)**

**3.1.4.1 Định nghĩa**

Cross-Site Scripting (XSS) là một kỹ thuật tấn công mà kẻ tấn công chèn mã JavaScript độc hại vào trang web hoặc ứng dụng web, thường thông qua các trường nhập liệu của người dùng. Khi người dùng khác truy cập trang web, mã JavaScript này được thực thi trong trình duyệt của họ.

**3.1.4.2 Cách nhận biết**

* Chèn mã HTML/JavaScript: Sự xuất hiện của các đoạn mã HTML hoặc JavaScript không an toàn trong dữ liệu đầu vào người dùng.
* Sự thay đổi trong giao diện người dùng: Các biến đổi bất thường trong nội dung của trang web, thường là sự xuất hiện của thông điệp không mong muốn hoặc các hành động không thể giải thích được.
* Kiểm tra trình duyệt: Sử dụng các công cụ kiểm thử bảo mật để phát hiện các lỗ hổng XSS trong mã nguồn.

**3.1.4.3 Cách ngăn chặn**

* Escape/Encode dữ liệu người dùng: Trước khi hiển thị dữ liệu người dùng trên trang web, hãy sử dụng các hàm escape/encode HTML để ngăn chặn việc thực thi mã JavaScript.
* HTTP Only Cookies: Sử dụng HTTP Only flag cho cookies để ngăn chặn truy cập từ mã JavaScript và giảm nguy cơ chiếm đoạt phiên làm việc.
* Content Security Policy (CSP): Thiết lập và thực thi CSP để hạn chế nguồn của các tài nguyên có thể được tải và thực thi trên trang web.
* Kiểm tra và xử lý đầu vào người dùng: Kiểm tra và lọc dữ liệu đầu vào để ngăn chặn các ký tự đặc biệt và mã độc hại.

**3.1.5. Lỗ Hổng Unvalidated Redirects and Forwards**

**3.1.5.1 Định nghĩa**

Lỗ hổng Unvalidated Redirects and Forwards xảy ra khi ứng dụng chấp nhận đầu vào không an toàn để chuyển hướng người dùng đến các địa chỉ mà không kiểm tra tính hợp lệ của đích đến, mở cửa cho nguy cơ chuyển hướng đến trang web độc hại.

**3.1.5.2 Cách nhận biết**

* Kiểm tra Mã Nguồn Ứng Dụng:

+ Xem cách ứng dụng xử lý chuyển hướng và cách nó xây dựng các đường dẫn chuyển hướng.

* Kiểm Thử Đầu Vào
* Nhập giá trị đầu vào không hợp lệ hoặc URL chuyển hướng độc hại để kiểm tra xem ứng dụng có kiểm tra và từ chối chúng hay không.
* Theo Dõi Yêu Cầu HTTP
* Sử dụng công cụ theo dõi yêu cầu HTTP để xác định liệu địa chỉ chuyển hướng được kiểm tra hay không.

**3.1.5.3 Cách ngăn chặn**

* + Kiểm Tra và Xử Lý Đầu Vào:

+ Kiểm tra tính hợp lệ của URL trước khi chấp nhận và sử dụng nó để chuyển hướng.

* + Sử Dụng Whitelist:

+ Sử dụng danh sách trắng của các URL hợp lệ để chỉ chấp nhận chuyển hướng đến các địa chỉ đã xác định.

* + Thông Báo Người Dùng:

+ Cảnh báo người dùng về chuyển hướng và cung cấp thông tin chi tiết để họ có thể kiểm tra và xác nhận tính hợp lệ.

* + Sử Dụng Phương Pháp Post/Token:

+ Sử dụng phương pháp POST kèm theo token thay vì chuyển hướng ngay lập tức từ URL.

* + Kiểm Tra Phản Hồi Ứng Dụng:

+ Kiểm tra cách ứng dụng phản hồi khi nhận đầu vào không hợp lệ. Nếu không có thông báo lỗi hoặc xử lý đầu vào không đúng cách, có thể là dấu hiệu của lỗ hổng.

* + Sử Dụng Công Cụ Kiểm Thử Tự Động:

+ Sử dụng công cụ kiểm thử tự động để tự động phát hiện các lỗ hổng bảo mật trong ứng dụng, bao gồm Unvalidated Redirects and Forwards.

**3.1.6. Kernel Exploits**

**3.1.6.1 Định nghĩa**

Lỗ hổng Kernel Exploits xuất hiện khi kẻ tấn công khai thác các lỗ hổng trong lõi (kernel) của hệ điều hành. Lõi là một thành phần quan trọng, quản lý tài nguyên và cung cấp giao diện giữa phần cứng và phần mềm.

**3.1.6.2 Cách nhận biết**

* Kiểm Tra Logs Hệ Thống:

+ Xem xét các logs hệ thống để phát hiện các sự kiện không bình thường, lỗi kernel, hay cố gắng tấn công vào lõi.

* Kiểm Thử Bảo Mật:

+ Sử dụng các công cụ như kernel fuzzer để tìm kiếm lỗ hổng trong mã nguồn lõi.

* Giám Sát Hệ Thống:

+ Thiết lập giám sát hệ thống để theo dõi sự thay đổi bất thường trong bộ nhớ, lưu lượng mạng, và các hoạt động kernel.

**3.1.6.3 Cách ngăn chặn**

* Cập Nhật Kernel:

+ Thường xuyên cập nhật kernel với các bản vá bảo mật mới nhất để giảm thiểu rủi ro từ lỗ hổng đã biết.

* Hạn Chế Quyền Truy Cập:

+ Hạn chế quyền truy cập vào kernel bằng cách thiết lập nguyên tắc least privilege. Giảm thiểu quyền truy cập giúp giảm thiểu tác động khi có lỗ hổng.

* Sử Dụng Công Cụ Kiểm Thử Kernel:

+ Sử dụng các công cụ như KASAN (KernelAddressSANitizer) để phát hiện lỗ hổng và lỗi trong quá trình phát triển và kiểm thử kernel.

**3.1.7. Security misconfiguration**

**3.1.7.1. Định nghĩa**

Security misconfiguration là một lỗi phổ biến trong việc triển khai ứng dụng web hoặc hệ thống thông tin, xuất phát từ việc cấu hình hệ thống mạng, máy chủ, ứng dụng hoặc framework không đúng cách, để lại các lỗ hổng bảo mật có thể bị tấn công.

**3.1.7.2. Cách nhận biết**

* **Quét bảo mật tự động:** Sử dụng công cụ quét bảo mật tự động như OWASP ZAP, Nessus, hoặc OpenVAS để kiểm tra các cấu hình không an toàn trong hệ thống.
* **Kiểm tra hệ thống cấu hình:** Kiểm tra các tập tin cấu hình của ứng dụng, máy chủ và hệ thống mạng để đảm bảo rằng chúng không bị để mặc định hoặc cấu hình không an toàn.
* **Quét mã nguồn mở:** Kiểm tra mã nguồn mở của ứng dụng hoặc framework để tìm kiếm các cấu hình không an toàn, như các giá trị cứng cố định, không được mã hóa, hoặc không được bảo vệ đúng cách.

**3.1.7.3 Cách ngăn chặn**

* **Thực hiện kiểm tra cấu hình định kỳ:** Kiểm tra và cập nhật các cấu hình hệ thống, máy chủ và ứng dụng định kỳ để đảm bảo rằng chúng không bị để mặc định hoặc cấu hình không an toàn.
* **Sử dụng các hướng dẫn bảo mật:** Sử dụng các hướng dẫn bảo mật được cung cấp bởi các tổ chức như OWASP để áp dụng các cấu hình an toàn cho hệ thống của bạn.
* **Giảm thiểu bề dày cuộc tấn công:** Loại bỏ hoặc giảm thiểu sự tồn tại của các thành phần không cần thiết trong hệ thống, đồng thời áp dụng các biện pháp bảo mật như tường lửa và bộ lọc gói tin để giảm thiểu khả năng tấn công từ bên ngoài.
* **Kiểm tra mã nguồn:** Thực hiện kiểm tra mã nguồn định kỳ để phát hiện và loại bỏ các cấu hình không an toàn từ mã nguồn mở.
* **Hạn chế quyền truy cập:** Áp dụng nguyên tắc nguyên lý lưỡng năng và hạn chế quyền truy cập của người dùng và các thành phần hệ thống đến những gì cần thiết cho nhiệm vụ của họ.

**3.1.8. Cross-Site Request Forgery**

**3.1.8.1.Định nghĩa**

Cross-Site Request Forgery (CSRF) là một loại tấn công mạng mà kẻ tấn công tạo ra và gửi các yêu cầu từ một nguồn website mà nạn nhân đã được xác thực để thực hiện các hành động không mong muốn trên một website khác mà nạn nhân đang được truy cập, thường là để lợi dụng các phiên đăng nhập hiện tại của nạn nhân.

**3.1.8.2. Cách nhận biết**

* **Kiểm tra lịch sử hoạt động của người dùng:** Kiểm tra lịch sử hoạt động của người dùng để xác định xem có bất kỳ hành động nào không mong muốn hoặc không được phê duyệt đã được thực hiện.
* **Phân tích yêu cầu HTTP**: Phân tích các yêu cầu HTTP để xem xét xem có bất kỳ hoạt động nào không bình thường nào có thể gây ra CSRF.
* **Sử dụng công cụ phát hiện tấn công:** Sử dụng các công cụ phát hiện tấn công để theo dõi và phát hiện các yêu cầu không mong muốn hoặc không được phê duyệt.

**3.1.8.3. Cách ngăn chặn**

* **Sử dụng token CSRF (CSRF token):** Đảm bảo rằng mỗi yêu cầu của người dùng đi kèm với một token CSRF không dự đoán được và độc lập để xác thực nguồn gốc của yêu cầu. Khi nhận được yêu cầu, máy chủ kiểm tra xem token này có khớp với token đã tạo cho phiên đăng nhập hiện tại hay không.
* **Thực hiện SameSite cookie attribute:** Đặt thuộc tính SameSite cho các cookie trong cấu hình của máy chủ web để giảm thiểu nguy cơ của CSRF. SameSite attribute có thể được đặt thành 'Strict' hoặc 'Lax' để hạn chế các yêu cầu CSRF từ các trang web khác.
* **Sử dụng Captcha:** Sử dụng Captcha để xác minh xem người dùng có phải là con người hay không, giảm nguy cơ của tấn công CSRF từ các bot hoặc các kịch bản tự động.
* **Giảm thiểu sự tin tưởng đối với cookie:** Tránh sử dụng các thông tin nhạy cảm hoặc quyền truy cập trong cookie và luôn xác thực lại người dùng trước khi thực hiện bất kỳ hành động quan trọng nào.

**3.1.9. Unvalidated Redirects and Forwards**

**3.1.9.1. Định nghĩa**

Unvalidated Redirects and Forwards là một loại lỗ hổng bảo mật trong các ứng dụng web, khi các trang web chấp nhận thông tin đầu vào từ người dùng và sử dụng thông tin này để chuyển hướng người dùng đến một địa chỉ URL khác mà không kiểm tra xác thực đối với URL đó. Điều này có thể được lợi dụng bởi kẻ tấn công để điều hướng người dùng đến các trang web độc hại hoặc phishing.

**3.1.9.2. Cách nhận biết**

* **Kiểm tra URL chuyển hướng:** Kiểm tra các yêu cầu chuyển hướng từ các trang web và xác minh xem chúng có kiểm tra hoặc xác thực URL đích không.
* **Phân tích mã nguồn:** Phân tích mã nguồn của trang web để tìm kiếm các đoạn mã hoặc chức năng liên quan đến chuyển hướng và xác minh liệu chúng có thực hiện xác thực URL không.
* **Sử dụng công cụ kiểm thử bảo mật:** Sử dụng các công cụ kiểm thử bảo mật tự động để kiểm tra các lỗ hổng bảo mật trong ứng dụng, bao gồm Unvalidated Redirects and Forwards.

**3.1.9.3. Cách ngăn chặn**

* **Xác thực và kiểm tra URL**: Luôn kiểm tra và xác thực URL đích trước khi thực hiện chuyển hướng hoặc chuyển tiếp. Đảm bảo rằng URL chỉ được phép chuyển hướng đến các địa chỉ URL đã được xác thực và được quản lý.
* **Sử dụng danh sách trắng:** Sử dụng danh sách trắng (whitelist) để chỉ cho phép chuyển hướng đến các địa chỉ URL được xác thực và được cho phép trước.
* **Thông báo cho người dùng:** Thông báo cho người dùng về việc họ sẽ được chuyển hướng và cung cấp thông tin chi tiết về địa chỉ URL đích, giúp họ nhận ra các chuyển hướng không mong muốn.
* **Giảm thiểu việc sử dụng chuyển hướng và chuyển tiếp:** Tránh sử dụng chuyển hướng và chuyển tiếp nếu không cần thiết. Nếu có thể, hãy tạo ra một trang thông báo hoặc cung cấp các liên kết cụ thể cho người dùng thay vì chuyển hướng tự động.

### Lỗ hổng bảo mật App

#### 3.2.1. Lỗ hổng XSRF/CSRF (Cross-Site Request Forgery):

**3.2.1.1.Định nghĩa**

Lỗ hổng này cho phép kẻ tấn công thực hiện các hành động không được ủy quyền trên ứng dụng bằng cách lừa đảo người dùng thực hiện các thao tác không mong muốn.

* + - 1. ***Cách nhận biết***
* **Kiểm tra các yêu cầu không mong muốn:** Kiểm tra các yêu cầu HTTP được gửi từ trình duyệt của người dùng mà họ không thực sự thực hiện hoặc không có ý định thực hiện. Điều này có thể bao gồm việc thực hiện các hành động như thay đổi mật khẩu, thực hiện các giao dịch tài chính, hoặc thay đổi cài đặt của tài khoản người dùng mà không cần xác nhận từ người dùng.
* **Kiểm tra bất thường trong lưu lượng mạng:** Theo dõi và phân tích lưu lượng mạng để phát hiện các hoạt động không thường xuyên hoặc không mong muốn, như việc gửi các yêu cầu không xác định hoặc không được kiểm soát từ các nguồn không đáng tin cậy.
* **Kiểm tra các yêu cầu không chứa token CSRF:** Nếu ứng dụng của bạn sử dụng cơ chế bảo vệ bằng token CSRF, bạn có thể kiểm tra các yêu cầu HTTP để xem liệu chúng có chứa token CSRF không. Nếu một yêu cầu không chứa token CSRF hoặc token không hợp lệ, đó có thể là dấu hiệu của lỗ hổng CSRF.
* **Kiểm tra header Referer:** Một cách phổ biến để ngăn chặn CSRF là kiểm tra header Referer của yêu cầu HTTP. Trong một số trường hợp, một yêu cầu được gửi từ một trang web khác nên không có header Referer hoặc header Referer không chính xác.
* **Sử dụng công cụ kiểm thử bảo mật:** Có các công cụ kiểm thử bảo mật có thể tự động phát hiện lỗ hổng CSRF trong ứng dụng của bạn bằng cách gửi các yêu cầu giả mạo và kiểm tra cách ứng dụng xử lý chúng.
* **Kiểm tra mã nguồn ứng dụng:** Kiểm tra mã nguồn của ứng dụng để xác định xem có bất kỳ điểm nào mà các yêu cầu HTTP có thể được chấp nhận mà không cần xác thực hoặc chứa token CSRF.

**3.2.1.3. Cách ngăn chặn**

* **Sử dụng CSRF Token**: Cung cấp một CSRF token cho mỗi yêu cầu từ phía người dùng và yêu cầu token này được bao gồm trong mỗi yêu cầu đến ứng dụng của bạn. Khi nhận được yêu cầu, ứng dụng sẽ kiểm tra xem token này có hợp lệ không, nếu không, yêu cầu sẽ bị từ chối.
* **Sử dụng SameSite Cookie Attribute:** Thiết lập thuộc tính SameSite cho các cookie, có thể giúp ngăn chặn CSRF bằng cách chỉ cho phép các yêu cầu từ cùng một trang web gốc (same-site requests) sử dụng cookie.
* **Kiểm tra Referer Header**: Kiểm tra header Referer trong các yêu cầu đến ứng dụng của bạn để đảm bảo rằng yêu cầu được gửi từ cùng một trang web gốc.
* **Xác thực Origin:** Xác thực nguồn (Origin) của yêu cầu, đảm bảo rằng yêu cầu được gửi từ một nguồn đáng tin cậy.
* **Sử dụng CSRF Double Submit Cookie:** Trong phương pháp này, một cookie chứa một mã CSRF token được tạo mỗi khi một người dùng đăng nhập và một trường ẩn trong mỗi biểu mẫu yêu cầu cũng chứa giá trị token tương tự. Khi biểu mẫu được gửi đi, ứng dụng sẽ so sánh giá trị của cookie và giá trị trong trường ẩn để xác định tính hợp lệ của yêu cầu.
* **Sử dụng Captcha:** Sử dụng Captcha để xác thực người dùng, đặc biệt đối với các hoạt động nhạy cảm như thay đổi mật khẩu hoặc thực hiện các giao dịch tài chính.
* **Thực hiện bảo mật đối với phương thức HTTP:** Sử dụng phương thức POST thay vì GET để thực hiện các hoạt động có thể gây ảnh hưởng lớn.
* **Cập nhật trình duyệt và framework:** Luôn cập nhật trình duyệt web và các framework, thư viện được sử dụng trong ứng dụng của bạn để đảm bảo rằng họ đã áp dụng các biện pháp bảo mật mới nhất chống lại CSRF.

#### Lỗ hổng SQL Injection:

* + - 1. **Định nghĩa**

Lỗ hổng SQL Injection là một loại lỗ hổng bảo mật phổ biến trong các ứng dụng web. Kẻ tấn công sử dụng lỗ hổng này để chèn các câu lệnh SQL độc hại vào các trường dữ liệu nhập vào của ứng dụng, thông qua các trường như biểu mẫu, trường tìm kiếm hoặc thậm chí các tham số trong URL. Khi ứng dụng không kiểm tra hoặc xử lý đầu vào người dùng một cách an toàn, tin tặc có thể thực thi các câu lệnh SQL không mong muốn, gây ra các tác động bảo mật nghiêm trọng như truy cập trái phép vào cơ sở dữ liệu, thay đổi hoặc xóa dữ liệu, hoặc thậm chí kiểm soát toàn bộ hệ thống.

* + - 1. **Cách nhận biết**
* Các lỗi SQL xuất hiện trên giao diện người dùng, thường là các thông báo lỗi hoặc dữ liệu không xử lý đúng cách.
* Các yêu cầu HTTP không bình thường hoặc không được dự kiến gửi đến ứng dụng, chẳng hạn như các tham số URL hoặc trường dữ liệu biểu mẫu không hợp lệ.
* Sự thay đổi bất thường trong dữ liệu hoặc cơ sở dữ liệu, như việc hiển thị thông tin không chính xác hoặc mất dữ liệu.
  + - 1. **Cách ngăn chặn**
* Sử dụng các câu lệnh SQL tham số hóa hoặc câu lệnh Prepared Statement trong mã của bạn, thay vì tạo câu lệnh SQL bằng cách nối chuỗi với dữ liệu nhập từ người dùng.
* Thực hiện kiểm tra và xử lý đầu vào người dùng một cách an toàn, bao gồm việc loại bỏ hoặc mã hóa các ký tự đặc biệt có thể được sử dụng để phá vỡ câu lệnh SQL.
* Áp dụng các chính sách bảo mật và phân quyền đầy đủ trên cơ sở dữ liệu, giảm thiểu rủi ro khi lỗ hổng xảy ra.
* Thực hiện kiểm thử bảo mật định kỳ để phát hiện và sửa chữa các lỗ hổng SQL Injection có thể tồn tại trong ứng dụng của bạn.

#### Lỗ hổng Insecure Deserialization:

* + - 1. **Định nghĩa**

Lỗ hổng Insecure Deserialization là một loại lỗ hổng bảo mật phổ biến trong ứng dụng web và các ứng dụng khác mà sử dụng quá trình deserialization để chuyển đổi dữ liệu từ dạng đã mã hóa (như JSON hoặc XML) sang các đối tượng trong mã nguồn. Khi ứng dụng không kiểm tra hoặc xác minh dữ liệu deserialized một cách an toàn, tin tặc có thể chèn dữ liệu độc hại hoặc thực thi mã từ xa, dẫn đến các cuộc tấn công như kiểm soát toàn bộ hệ thống, thực hiện các hành động không mong muốn, hoặc thậm chí làm suy yếu toàn bộ ứng dụng.

* + - 1. **Cách nhận biết**
* Sự thay đổi bất thường trong hành vi của ứng dụng hoặc dữ liệu, chẳng hạn như việc xuất hiện đối tượng không mong muốn hoặc các giá trị dữ liệu không chính xác.
* Các lỗi hoặc thông báo lỗi xuất hiện trên giao diện người dùng hoặc trong các bản ghi log của ứng dụng, liên quan đến việc deserialization dữ liệu.
* Sự xuất hiện của các đối tượng hoặc dữ liệu độc hại trong cơ sở dữ liệu hoặc hệ thống tập tin.

* + - 1. **Cách ngăn chặn**
* Sử dụng các thư viện deserialization an toàn và đáng tin cậy từ các nguồn đáng tin cậy.
* Xác minh và kiểm tra dữ liệu deserialized một cách cẩn thận, đảm bảo rằng nó đúng định dạng và không chứa bất kỳ dữ liệu độc hại nào.
* Giới hạn quyền truy cập của người dùng hoặc quy trình thực thi để hạn chế khả năng thực thi mã độc hại từ dữ liệu deserialized.
* Sử dụng các kỹ thuật mã hóa hoặc ký tự hóa để bảo vệ dữ liệu deserialized khỏi các cuộc tấn công như thay đổi dữ liệu hay thực thi mã độc hại.

### 3.3.Lỗ hổng bảo mật OS

#### 3.3.1. Lỗ hổng Kernel

**3.3.1.1. Định nghĩa**

Đây là lỗ hổng xảy ra trong kernel của hệ điều hành, là phần trung tâm của hệ thống điều hành quản lý tài nguyên và tương tác với phần cứng. Một lỗ hổng kernel có thể cho phép kẻ tấn công thực hiện các hành động không được phép hoặc thậm chí kiểm soát toàn bộ hệ thống.

**3.3.1.2. Cách nhận biết**

* **Thông báo lỗi hệ thống (System Error Messages):** Kernel thường ghi lại các thông báo lỗi trong các tập tin như /var/log/messages, /var/log/syslog, hoặc /var/log/kern.log trên hệ thống Linux. Các thông báo lỗi này có thể cho thấy các vấn đề như lỗi bộ nhớ, lỗi kernel panic, hoặc các lỗi hệ thống khác.
* **Sự không ổn định hoặc tự động khởi động lại (Instability or Automatic Reboots):** Nếu hệ thống thường xuyên gặp sự cố không lường trước hoặc tự động khởi động lại mà không có lý do rõ ràng, có thể đó là dấu hiệu của một lỗ hổng kernel.
* **Hiệu suất kém (Poor Performance):** Một lỗ hổng kernel có thể dẫn đến hiệu suất hệ thống kém đi hoặc tăng cường sự chậm trễ trong các hoạt động hệ thống cơ bản.
* **Sự khai thác bất thường (Unusual Exploitation):** Sự xuất hiện của các kỹ thuật tấn công mới hoặc các cuộc tấn công bất thường có thể là dấu hiệu của việc khai thác lỗ hổng kernel.
* **Phát hiện thông qua công cụ giám sát (Detection via Monitoring Tools):** Sử dụng các công cụ giám sát hệ thống như Nagios, Zabbix, hoặc Prometheus để theo dõi các chỉ số hệ thống như tải CPU, tình trạng bộ nhớ, hoạt động mạng, và các chỉ số khác có thể giúp phát hiện các vấn đề liên quan đến kernel.
* **Phân tích log và hành vi không bình thường (Analysis of Logs and Abnormal Behavior):** Kiểm tra các log hệ thống và phân tích hành vi không bình thường của hệ thống, chẳng hạn như các hoạt động không được phép hoặc không mong muốn, có thể giúp phát hiện các dấu hiệu của lỗ hổng kernel.
* **Thực hiện kiểm tra bảo mật (Security Audits):** Sử dụng các công cụ kiểm tra bảo mật như Lynis, OpenVAS, hoặc Nessus để thực hiện kiểm tra bảo mật hệ thống, đóng vai trò quan trọng trong việc phát hiện các lỗ hổng kernel.

**3.3.1.3. Cách ngăn chặn**

* **Cập nhật hệ thống đến phiên bản mới nhất:** Cung cấp bản vá bảo mật cho kernel là một phần quan trọng của việc duy trì bảo mật hệ thống. Cập nhật hệ thống đến phiên bản mới nhất giúp giảm thiểu nguy cơ từ các lỗ hổng đã được phát hiện.
* **Kiểm tra và giảm thiểu số lượng module kernel không cần thiết:** Module kernel không cần thiết có thể tạo ra các điểm yếu tiềm ẩn trong hệ thống. Kiểm tra và tắt các module không cần thiết để giảm thiểu bề dày và tiềm năng của hệ thống.
* **Thực hiện cấu hình an toàn cho kernel:** Đảm bảo rằng kernel được cấu hình với các tùy chọn an toàn nhất cho môi trường cụ thể của bạn. Điều này bao gồm việc tắt các tính năng không cần thiết, thiết lập các giới hạn bảo mật và quản lý các cơ chế bảo mật như SELinux hoặc AppArmor.
* **Sử dụng giải pháp bảo mật thứ cấp:** Sử dụng các giải pháp bảo mật thứ cấp như intrusion detection systems (IDS) hoặc intrusion prevention systems (IPS) để giám sát và phòng ngừa các cuộc tấn công vào kernel.
* **Giám sát và kiểm tra lỗ hổng định kỳ:** Thực hiện kiểm tra bảo mật định kỳ và theo dõi các thông báo bảo mật mới từ nhà cung cấp kernel để đảm bảo rằng hệ thống của bạn luôn được bảo vệ khỏi các lỗ hổng mới phát hiện.
* **Thực hiện các biện pháp bảo mật mạng:** Sử dụng tường lửa và các giải pháp bảo mật mạng để giảm thiểu các nguy cơ từ các cuộc tấn công từ xa vào hệ thống.
* **Thực hiện kiểm tra bảo mật chặt chẽ:** Thực hiện kiểm tra bảo mật định kỳ và audit log để phát hiện sớm các vấn đề bảo mật trong hệ thống kernel.
* **Đào tạo và nhận thức về bảo mật:** Đảm bảo rằng nhân viên của bạn được đào tạo về các biện pháp bảo mật cơ bản và nhận thức về các nguy cơ tiềm ẩn từ các lỗ hổng kernel.

#### 3.3.2.Lỗ hổng Local File Inclusion (LFI)

**3.3.2.1. Định nghĩa**

Lỗ hổng Local File Inclusion (LFI) là một loại lỗ hổng bảo mật thường xuất hiện trong các ứng dụng web. Khi một ứng dụng web chấp nhận dữ liệu người dùng không được kiểm tra đúng cách, tin tặc có thể chèn đường dẫn của các tập tin cục bộ vào trong yêu cầu web, dẫn đến việc ứng dụng chấp nhận và hiển thị nội dung của các tập tin đó, thậm chí là các tập tin nhạy cảm như các tập tin cấu hình hệ thống hoặc mã nguồn.

**3.3.2.2.Cách nhận biết**

* Hiển thị đường dẫn hoặc nội dung của các tập tin không mong muốn trên giao diện người dùng hoặc trong các bản ghi log của ứng dụng.
* Các yêu cầu HTTP không bình thường hoặc không được dự kiến chứa đường dẫn tập tin cục bộ hoặc các tham số tìm kiếm tập tin.
* Sự xuất hiện của các lỗi hoặc thông báo lỗi liên quan đến việc mở hoặc xử lý các tập tin trong ứng dụng.

**3.3.2.3. Cách ngăn chặn**

* **Kiểm tra và xử lý đầu vào người dùng đúng cách:** Đảm bảo rằng ứng dụng của bạn kiểm tra và xử lý các đường dẫn tập tin người dùng nhập vào một cách an toàn, bao gồm kiểm tra đường dẫn tập tin có tồn tại và có nằm trong phạm vi cho phép không.
* **Giới hạn quyền truy cập tập tin:** Đảm bảo rằng hệ thống chỉ có thể truy cập và hiển thị các tập tin cần thiết, hạn chế quyền truy cập đến các tập tin nhạy cảm hoặc không cần thiết.
* **Sử dụng các phương thức an toàn để lấy và hiển thị nội dung tập tin:** Sử dụng các phương thức an toàn và đáng tin cậy để lấy và hiển thị nội dung của các tập tin, bao gồm việc sử dụng hệ thống tệp đính kèm hoặc phương pháp đọc tệp an toàn từ cơ sở dữ liệu.

## CHƯƠNG 4: THIẾT KẾ GIAO DIỆN VÀ CÁC CHỨC NĂNG

### Mô hình hóa chức năng

#### Biểu đồ use case

#### 

*Hình 4.1.1. Use case tìm kiếm*

#### 

*Hình 4.1.2. Use case chi tiết*

#### Mô tả use case

1. Tên Use Case:

* Tìm Kiếm

1. Mô tả vắn tắt:

* Use case cho phép người dùng tra cứu thông tin các lỗ hổng bảo mật Web/App/Os

1. Luồng sự kiện
2. Luồng cơ bản

* Use case bắt đầu khi người dùng kích vào thanh tìm kiếm ở đầu trang, hệ thống sẽ lấy thông tin chi tiết các lỗ hổng bảo mật Web/App/Os và hiển thị lên màn hình.

1. Luồng rẽ nhánh

* Tại luồng bước 1 luồng cơ bản nếu người dùng nhập thông tin không tồn tại trong cơ sở dữ liệu, hệ thống sẽ thông báo lên màn hình và yêu cầu người dùng nhập lại
* Tại bất kỳ bước nào trong luồng cơ bản, nếu không kết nối được với cơ sở dữ liệu thì hệ thống sẽ hiển thị một thông báo lỗi và use case kết thúc.

1. Các yêu cầu đặc biệt

* Không có

1. Tiền điều kiện

* Không có

1. Hậu điều kiện

* Không có

1. Điểm mở rộng

* Không có

### Giao diện người dung

#### Giao diện

- Giao diện thân thiện, dễ sử dụng, phù hợp với người dùng không chuyên về lĩnh vực công nghệ.

- Thiết kế đơn giản và trực quan: Giao diện nên được thiết kế đơn giản, dễ sử dụng và trực quan để người dùng có thể dễ dàng tìm kiếm thông tin lỗ hỏng mà họ quan tâm.

A screenshot of a computer

Description automatically generated

*Hình 4.2.1. Trang chủ*

#### Công cụ tìm kiếm

* + - * Cung cấp một công cụ tìm kiếm dễ sử dụng để người dùng có thể nhập từ khóa hoặc các tiêu chí tìm kiếm để tìm kiếm thông tin lỗ hỏng của các ứng dụng Web/App/OS.

A close up of a text

Description automatically generated

*Hình 4.2.2. Tìm kiếm*

#### 4.2.3. Danh sách lỗ hổng

* Hiển thị danh sách các lỗ hỏng ứng dụng Web/App/OS theo thứ tự và các thông tin quan trọng như tên lỗ hỏng, mô tả, ngày phát hiện, mức độ nghiêm trọng, và thông tin về phiên bản hệ điều hành.
* A screenshot of a computer

  Description automatically generated

*Hình 4.2.3. Giao diện về thông tin các lỗ hổng*

### Tổng kết lại các biện pháp bảo mật

- Trách nhiệm trong lĩnh vực bảo mật thông tin được phân chia rõ ràng giữa các bên liên quan, bao gồm nhà quản trị hệ thống, nhà phát triển, và người sử dụng. Dưới đây là trách nhiệm cụ thể của từng người:

#### Trách nhiệm đối với nhà quản trị hệ thống

* + - * Bảo dưỡng và cập nhật hệ thống: nhà quản trị hệ thống có trách nhiệm đảo bảo rằng tất cả hệ thống và phần mềm đều được cập nhật đều đặn với các bản vá bảo mật mới nhất.
      * Quản lý quyền truy cập: thiết lập và quản lý các quyền truy cập để đảm bảo rằng chỉ những người cần thiết mới có quyền truy cập vào tài nguyên quan trọng
      * Cấu hình Firewall và mô-đun bảo mật: thiết lập và duy trì cấu hình của firewall và các mô-đun bảo mật để lọc và giám sát luồng dữ liệu, ngăn chặn truy cập không ủy quyền.
      * Theo dõi và ghi log: thiết lập hệ thống theo dõi và ghi log để theo dõi hoạt động mạng và phát hiện nguy cơ bảo mật

#### Trách nhiệm với người sử dụng

* + - * Chọn mật khẩu mạnh: sử dụng cả chữ cái, số, ký tự đặc biệt. Tránh sử dụng mật khẩu dễ đoán.
      * Đăng nhập an toàn: luôn đăng nhập từ trang web chính thức và tránh đăng nhập thông qua liên kết gửi email hoặc tin nhắn.
      * Không sử dụng máy tính công cộng: tránh sử dụng máy tính công cộng hoặc máy tính của người khác để truy cập thông tin nhạy cảm.
      * Theo dõi tài khoản: kiểm tra định kỳ các giao dịch tài khoản và báo cáo ngay lập tức nếu có bất kỳ hoạt động đáng ngờ nào.
      * Tắt cookies khi không cần thiết để giảm nguy cơ theo dõi và tấn công XSS.
    1. ***Tổng kết:***

Mỗi bên đều đóng vai trò quan trọng trong việc bảo vệ hệ thống. Các nhà quản trị hệ thống chịu trách nhiệm về triển khai và duy trì hạ tầng bảo mật, những nhà phát triển cung cấp ứng dụng an toàn và bảo mật, còn người sử dụng có nhiệm vụ duy trì thái độ an toàn khi sử dụng dịch vụ trực tuyến. Sự hợp tác giữa tất cả các bên này là chìa khóa để xây dựng một hệ thống an toàn và bảo mật trong môi trường trực tuyến ngày nay.

## KẾT LUẬN

* + - * Trong quá trình nghiên cứu về bảo mật web/app/os, nhóm chúng em đã tập trung khám phá, phân tích và đề xuất các giải pháp an toàn cho các vấn đề quan trọng nhất liên quan đến an ninh thông tin trên nền trang mạng. Dưới đây là những điểm chính được đề cập trong phần kết luận của đề tài "Xây dựng phần mềm cơ sở dữ liệu về lỗ hổng bảo mật".
      * Kết luận tổng quan:
* Phần mềm cơ sở dữ liệu về lỗ hổng bảo mật đã đạt được mục tiêu chính của việc quản lý thông tin lỗ hổng.
* Giao diện người dùng thân thiện và dễ triển khai, phù hợp cho người dùng không chuyên sâu về công nghệ.
* Nhận thức rõ về những hạn chế và thách thức, cùng với hướng phát triển để nâng cao tính toàn diện và hiệu suất của hệ thống.

## TÀI LIỆU THAM KHẢO

### Tiếng Việt

* Dương, S. (2021). “Bảo mật ứng dụng web trên môi trường Internet”. Thư viện Trường Đại học Công nghiệp Hà Nội
* Lê, V. (2021, Tháng 6 10). “Session là gì, Cookie là gì?” TopDev. Truy cập từ <https://topdev.vn/blog/session-la-gi-cookie-la-gi/>
* Nguyễn, T. (2022, Tháng 2 5). “Nguyên tắc hoạt động của Website.” Marketing Toàn Cầu. Truy cập từ [https://marketingtoancau.com/tin-](https://marketingtoancau.com/tin-tuc/nguyen-tac-hoat-dong-website-303) [tuc/nguyen-tac-hoat-dong-website-303](https://marketingtoancau.com/tin-tuc/nguyen-tac-hoat-dong-website-303)
* Trịnh, N. (2020, Tháng 9 3). “HTTP và HTTPS là gì?” CyStack. Truy cập từ <https://cystack.net/blog/http-va-https-la-gi>
* Wikipedia. (n.d.). Website. Truy cập từ <https://vi.wikipedia.org/wiki/Website>

### Tiếng Anh

* ADVANCED XSS AND CSRF EXPLOITATION. Truy cập từ

<https://academy.hackthebox.com/module/details/235>

* Juice-shop developments and issues. Truy cập từ <https://github.com/juice-shop/juice-shop>
* Tác giả. (Năm). “What is a Digital Signature.” Binance Academy. Truy cập từ <https://academy.binance.com/articles/what-is-a-digital-signature>
* Top 10 OWASP. Truy cập từ <https://owasp.org/www-project-top-ten/>
* Web Application Security. Truy cập từ <https://tryhackme.com/room/introwebapplicationsecurity>