TRƯỜNG ĐẠI HỌC KHOA HỌC

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO MÔN HỌC

AN NINH MẠNG

Cross-Site Scripting và biện pháp phòng chống (Stored and Reflected)

Giáo viên hướng dẫn:
ThS. Võ Việt Dũng

Sinh viên thực hiện :
1. Trần Tấn Toàn **(Nhóm trưởng)**
2. Phạm Thị Uyển Nhi
3. Nguyễn Thanh Hải
4. Hồ Giang Sơn

Nhóm: 16
Lớp: TIN3163.002
Năm học: 2025-2026

Huế, 12/2025

Mục lục

Lời cảm ơn

Lời đầu tiên, nhóm chúng em xin gửi lời cảm ơn chân thành đến Khoa Công nghệ thông tin – Trường Đại học Khoa học thành phố Huế đã đưa bộ môn An ninh mạng vào chương trình giảng dạy. Môn học đã giúp chúng em có cơ hội tiếp cận với những kiến thức chuyên sâu, hiểu rõ hơn về các thách thức bảo mật và định hướng nghề nghiệp trong tương lai của mình. Đồng thời, chúng em cũng xin cảm ơn Nhà trường đã tạo điều kiện về cơ sở vật chất cùng hệ thống thư viện hiện đại, cung cấp nguồn tài liệu phong phú giúp chúng em có thêm tư liệu quý giá để hoàn thành bài tiểu luận này. Đặc biệt, trong suốt quá trình học tập và nghiên cứu, chúng em đã nhận được sự hướng dẫn tận tình, tâm huyết từ ThS. Võ Việt Dũng. Những kiến thức và góp ý quý báu của thầy không chỉ giúp chúng em hiểu rõ hơn về đề tài mà còn rèn luyện tư duy của một người làm an ninh mạng. Nhóm chúng em xin được bày tỏ lòng biết ơn sâu sắc nhất tới thầy! Trong quá trình thực hiện đề tài "**Kỹ thuật tấn công XSS (Cross Site Scripting) và cách ngăn chặn**", dù đã cố gắng nhưng do kiến thức và kinh nghiệm thực tiễn còn hạn chế, bài làm khó tránh khỏi những thiếu sót. Nhóm chúng em rất kính mong nhận được những ý kiến đóng góp của thầy để bài tiểu luận được hoàn thiện hơn.

Chúng em xin chân thành cảm ơn!

Phần mở đầu

Trong thời đại bùng nổ công nghệ thông tin, an toàn thông tin đã trở thành vấn đề cấp thiết hàng đầu. Các hệ thống Website với lượng dữ liệu khổng lồ được cập nhật liên tục bởi người dùng trên toàn cầu đang trở thành mục tiêu hấp dẫn cho các cuộc tấn công mạng tinh vi. Trong số đó, Cross-Site Scripting (XSS) nổi lên như một trong những lỗ hổng phổ biến và nguy hiểm nhất, đe dọa trực tiếp đến tính bảo mật của người dùng và uy tín của các tổ chức.

XSS lợi dụng các khiếm khuyết trong việc kiểm soát dữ liệu đầu vào để chèn mã độc, từ đó đánh cắp thông tin nhạy cảm (như Cookie, Session), chiếm quyền điều khiển tài khoản, hoặc điều hướng người dùng đến các trang web độc hại. Nhận thức được mối nguy hiểm này, nhóm quyết định thực hiện đề tài: "Kỹ thuật tấn công XSS (Cross Site Scripting) và cách ngăn chặn". Bài tiểu luận này tập trung nghiên cứu chuyên sâu vào hai dạng tấn công chính là:

+ Stored XSS (XSS lưu trữ): Dạng tấn công nguy hiểm khi mã độc được lưu trữ trực tiếp trên cơ sở dữ liệu của máy chủ.

+ Reflected XSS (XSS phản xạ): Dạng tấn công phổ biến thông qua các đường dẫn (URL) chứa mã độc. Việc đi sâu vào phân tích bản chất, cơ chế hoạt động và các kịch bản tấn công thực tế của hai dạng XSS này không chỉ giúp chúng em nâng cao kiến thức chuyên môn về An ninh mạng mà còn là cơ sở để đề xuất các giải pháp phòng ngừa hiệu quả. Tiểu luận sẽ đưa ra các kỹ thuật ngăn chặn thực tiễn dành cho cả nhà phát triển phần mềm (Developer) và người sử dụng, góp phần xây dựng một môi trường Internet an toàn và bền vững hơn.

Dù đã dành nhiều tâm huyết nghiên cứu, song do hạn chế về kinh nghiệm thực tiễn, bài làm chắc chắn không tránh khỏi những thiếu sót. Nhóm chúng em rất mong nhận được những ý kiến đóng góp từ ThS. Võ Việt Dũng để đề tài được hoàn thiện và có tính ứng dụng cao hơn.

Chương I: Giới thiệu chung

1.1. Giới thiệu về kỹ thuật tấn công XSS

1.1.1. Cross Site Scripting (XSS) là gì ?

Cross Site Scripting (XSS) là một lỗi bảo mật cho phép kẻ tấn công chèn các đoạn mã độc nguy hiểm vào trong source code ứng dụng web. Để khai thác một lỗ hổng XSS, hacker sẽ chèn mã độc thông qua các đoạn script để thực thi chúng ở phía Client, thông thường, các cuộc tấn công XSS được sử dụng để vượt qua truy cập và mạo danh người dùng. Đây còn là một trong những kĩ thuật tấn công phổ biến nhất hiện nay, được liệt vào danh sách những kỹ thuật tấn công nguy hiểm nhất với ứng dụng web, mà tất cả các Tester có kinh nghiệm đều biết đến.

1.1.2. Mục tiêu của tấn công XSS Kẻ tấn công có thể sử dụng XSS để thực hiện nhiều mục tiêu khác nhau, bao gồm:

- Đánh cắp cookie: Cookie chứa thông tin đăng nhập và các thông tin cá nhân khác của người dùng.
- Chiếm quyền điều khiển tài khoản: Kẻ tấn công có thể chiếm quyền điều khiển tài khoản của người dùng để thực hiện các hành vi trái phép.
- Thay đổi nội dung website: Kẻ tấn công có thể thay đổi nội dung website để hiển thị thông tin giả mạo hoặc lừa đảo.
- Phát tán mã độc: Kẻ tấn công có thể sử dụng XSS để phát tán mã độc sang các máy tính khác.
- Tấn công từ chối dịch vụ (DoS): Kẻ tấn công có thể sử dụng XSS để làm quá tải website, khiến người dùng không thể truy cập.

1.1.3. Cách thức tấn công XSS Kẻ tấn công thường khai thác các lỗ hổng bảo mật trên website, chẳng hạn như:

- Thiếu kiểm tra đầu vào: Website không kiểm tra kỹ lưỡng dữ liệu đầu vào từ người dùng, cho phép kẻ tấn công chèn mã độc hại.
- Lỗ hổng trong mã nguồn: Lỗi trong mã nguồn của website có thể tạo điều kiện cho kẻ tấn công khai thác.
- Cấu hình bảo mật yếu: Website không được cấu hình bảo mật đúng cách, tạo cơ hội cho kẻ tấn công xâm nhập.

Chương II: KỸ THUẬT TẤN CÔNG XSS

2.1.Phân loại

Có 2 loại tấn công lỗ hổng XSS phổ biến, gồm:

2.1.1. Reflected XSS?

Reflected XSS (hay Non-persistent XSS) là loại tấn công mạng thông qua việc chèn Script độc vào các phản hồi mà trình duyệt web gửi cho máy chủ.

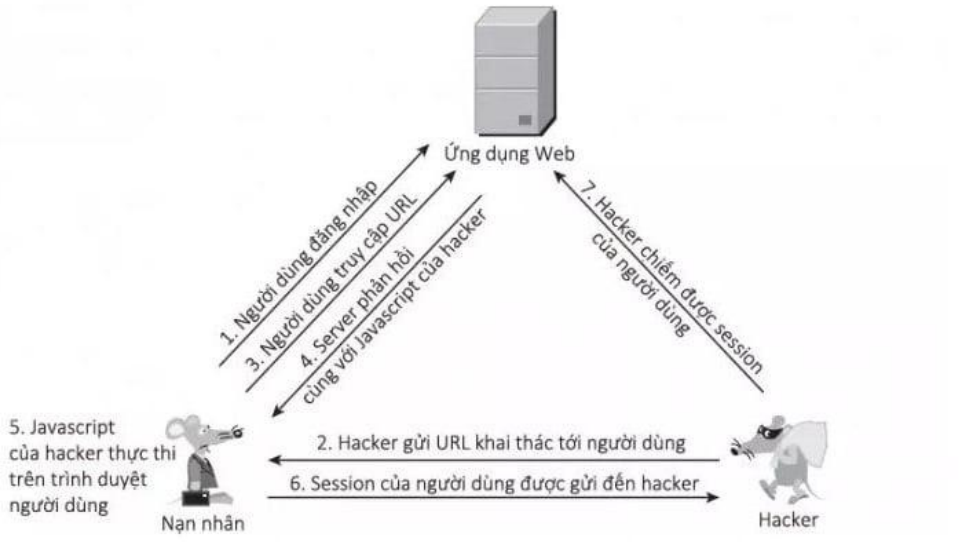
- Kẻ tấn công xâm nhập mã độc hại vào các tham số của URL hoặc các trường dữ liệu khác trên trang web mục tiêu

- Khi người dùng truy cập vào URL chứa các tham số này, trình duyệt của họ sẽ gửi yêu cầu tới máy chủ web và máy chủ web sẽ tạo một phản hồi chứa mã độc hại từ các tham số này

- Mã độc hại được thực thi trong môi trường trình duyệt của người dùng

dẫn đến việc đánh cắp thông tin cá nhân, thực hiện các hành động trái

phép hoặc thậm chí kiểm soát trang web đó



Hình 2.1.1.1. Mô hình tấn công Reflected XSS

Ví dụ:

Kẻ tấn công gửi email giả mạo với nội dung hấp dẫn, bên trong chứa đường link

độc hại. Khi nạn nhân click vào link thì trình duyệt sẽ chạy mã Javascript độc và

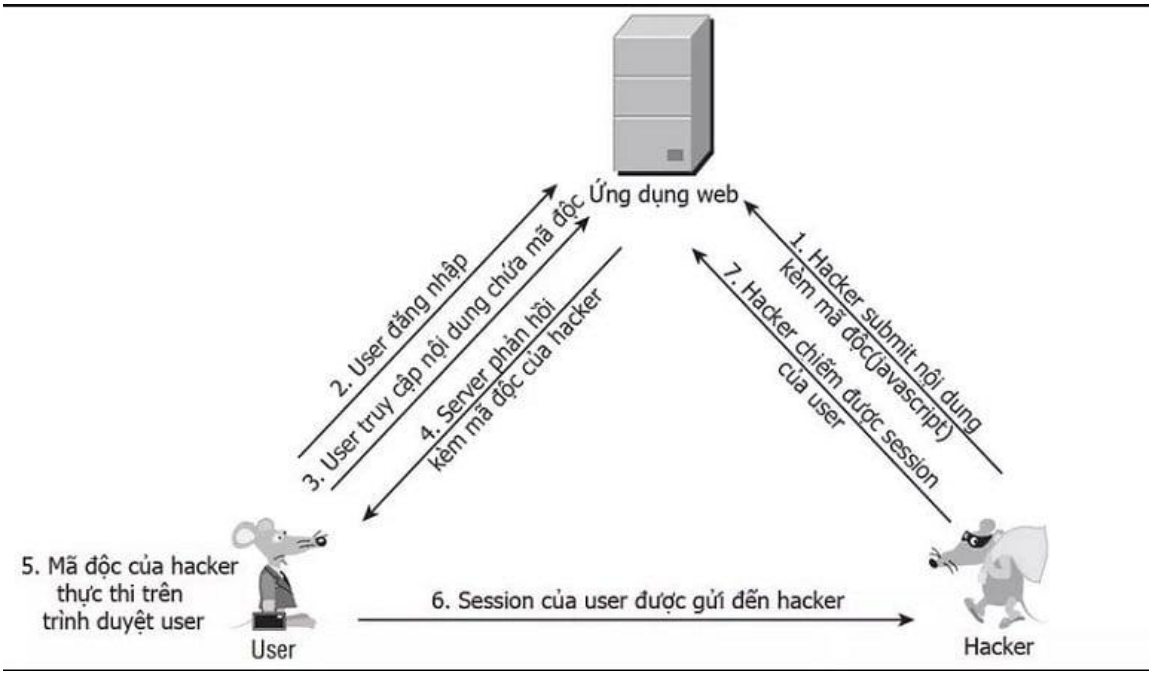
bị đánh cắp thông tin.

2.1.2. Stored XSS

Stored XSS (Persistent XSS hoặc Second-order XSS) hoạt động khi ứng dụng nhận dữ liệu từ một nguồn không đáng tin cậy và có phản hồi HTTP không an toàn. Theo đó, các mã độc sẽ xâm nhập vĩnh viễn vào cơ sở dữ liệu, tin nhắn, bình luận và cả hệ thống máy chủ. Khi nạn nhân truy xuất các mã độc đó từ máy chủ, Hacker sẽ có quyền kiểm soát mọi hoạt động của họ.

- Kẻ tấn công chèn mã JavaScript độc hại vào cơ sở dữ liệu của trang web

- Khi người dùng truy cập trang web, mã JavaScript độc hại sẽ được tải xuống và thực thi trên trình duyệt của họ



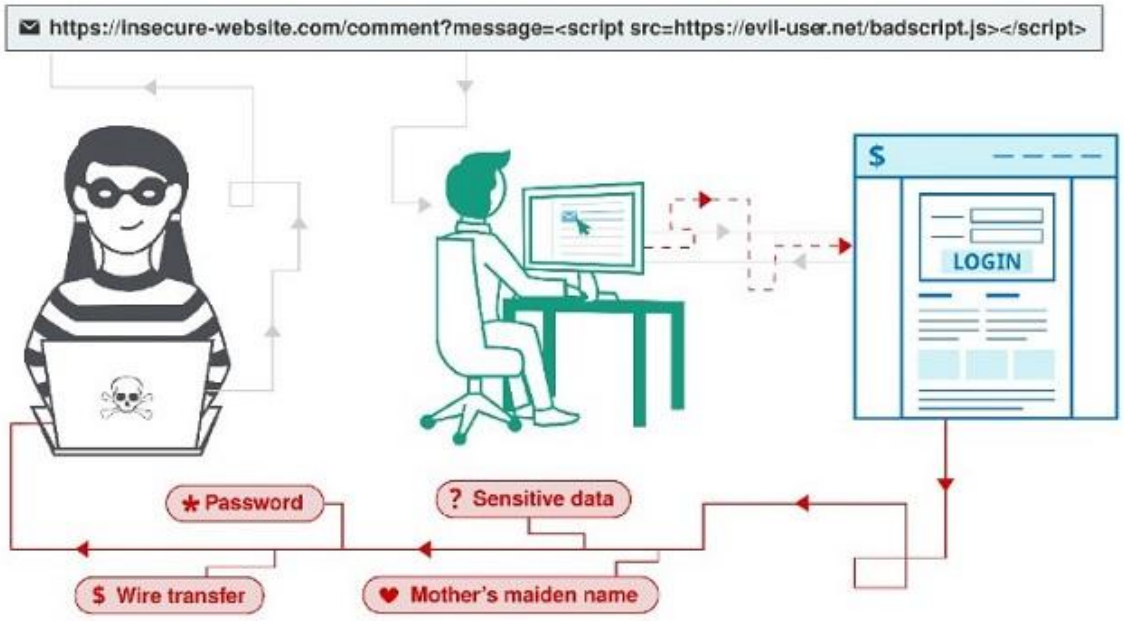
Hình 2.1.2.1. Mô hình tấn công Stored XSS

Ví dụ:

- Kẻ tấn công đăng nhập vào trang web và chèn mã JavaScript độc hại vào phần bình luận.
- Khi người dùng khác truy cập trang web và xem bình luận, mã JavaScript độc hại sẽ được thực thi trên trình duyệt của họ

2.1. Đặc tính cơ bản

2.2.1. Cách thức hiện



Hình 2.2.1.1. Cách thức tấn công của XSS

Tấn công Cross Site Scripting nghĩa là gửi và chèn lệnh, script độc hại, những mã độc này thường được viết với ngôn ngữ lập trình phía client như Javascript, HTML, VBScript, Flash… Tuy nhiên, cách tấn công này thông thường sử dụng Javascript và HTML. Cách tấn công này có thể được thực hiện theo nhiều cách khác nhau, phụ thuộc vào loại tấn công XSS, những mã độc có thể được phản chiếu trên trình duyệt của nạn nhân hoặc được lưu trữ trong cơ sở dữ liệu và được chạy mỗi khi người dùng gọi chức năng thích hợp.

Nguyên nhân chính của loại tấn công này là xác thực đầu vào dữ liệu người dùng không phù hợp, dữ liệu độc hại từ đầu vào có thể xâm nhập vào dữ liệu đầu ra. Mã độc có thể nhập một script và được chèn vào mã nguồn của website. Khi đó trình duyệt không thể biết mã thực thi có phải độc hại hay không. Do đó mã độc hại có thể đang được thực thi trên trình duyệt của nạn nhân hoặc bất kỳ hình thức giả nào đang được hiển thị cho người sử dụng.

2.2.2. Cách thức hoạt động

Giai đoạn 1: Kẻ tấn công tìm kiếm website dễ bị tấn công XSS.

Kẻ tấn công thường sử dụng các công cụ tự động hoặc thủ công để quét và phát hiện các website có lỗ hổng XSS.Các website có lỗ hổng này thường là những ứng dụng web chưa được bảo mật hoặc kiểm thử thất bại.

Giai đoạn 2: Chèn mã độc vào các form, URL,... của website.

Khi kẻ tấn công đã xác định được một website có lỗ hổng XSS, họ sẽ chèn mã độc vào các trường nhập liệu như form, tham số của URL, hộp bình luận, hoặc bất kỳ điểm nào mà ứng dụng web cho phép người dùng nhập dữ liệu.

Giai đoạn 3: Khi người dùng truy cập vào website đó, trình duyệt sẽ tự động chạy mã độc.

Khi người dùng truy cập vào website mà kẻ tấn công đã chèn mã độc, trình duyệt của họ sẽ tự động tải và thực thi đoạn mã đó. Điều này xảy ra vì trình duyệt web thường không phân biệt giữa mã JavaScript hợp lệ và không hợp lệ, và tự động thực thi tất cả mã JavaScript có trong trang.

Giai đoạn 4: Mã độc đánh cắp thông tin người dùng hoặc thực hiện các hành động độc hại khác.

Sau khi mã độc được thực thi, nó có thể thực hiện các hành động như đánh cắp thông tin người dùng (như tên đăng nhập, mật khẩu), chuyển hướng người dùng đến các trang web giả mạo, thực hiện các giao dịch không được phép, hoặc thậm chí kiểm soát toàn bộ trang web nếu có đủ quyền truy cập.

2.3. So sánh kỹ thuật Stored XSS và Reflected XSS

2.3.1. Khả năng gây hại

XSS không chỉ là việc hiển thị một hộp thoại thông báo đơn giản, mà nó là tiền đề cho các hành vi phá hoại nghiêm trọng.

Đối với người dùng:

- XSS không chỉ là việc hiển thị một hộp thoại thông báo đơn giản, mà nó là tiền đề cho các hành vi phá hoại nghiêm trọng:

- Đánh cắp thông tin định danh (Session Hijacking):

+ Hacker sử dụng các đoạn mã để lấy cắp Cookie và mã phiên (Session ID). Khi có được các thông tin này, kẻ tấn công có thể giả mạo nạn nhân để truy cập vào tài khoản mà không cần tên đăng nhập hay mật khẩu.

- Thu thập dữ liệu cá nhân (PII):

+ Mã độc có thể đọc và gửi về server của hacker các thông tin nhạy cảm có trên trang web như: Họ tên, địa chỉ, số điện thoại, thông tin thẻ tín dụng hoặc các bí mật kinh doanh được hiển thị trong trang quản trị.

- Chuyển hướng người dùng (Malicious Redirection):

+ XSS có thể tự động chuyển hướng người dùng từ một trang web tin cậy sang một trang web giả mạo (Phishing) hoặc trang web chứa mã độc nhằm lừa đảo chiếm đoạt tài sản.

- Phát tán mã độc (Malware Injection):

+ Kẻ tấn công có thể lợi dụng trình duyệt của nạn nhân để tải xuống và cài đặt các phần mềm độc hại, mã hóa dữ liệu (Ransomware) hoặc phần mềm gián điệp lên máy tính cá nhân.

- Lợi dụng tài nguyên máy tính (DDoS & Cryptojacking):

+ XSS có thể biến hàng ngàn trình duyệt của người dùng thành các "botnet" để thực hiện các cuộc tấn công từ chối dịch vụ (DDoS) vào các hệ thống khác hoặc lợi dụng phần cứng máy tính nạn nhân để đào tiền ảo.

2.3.2. Hạn chế của kỹ thuật

Dù là một kỹ thuật tấn công phổ biến, XSS vẫn tồn tại những hạn chế nhất định khiến hacker không phải lúc nào cũng thành công.

Đối với người tấn công:

- Yêu cầu sự tương tác của người dùng:

+ Đặc biệt với Reflected XSS, cuộc tấn công chỉ thành công khi lừa được nạn nhân click vào một đường link độc hại. Nếu người dùng có nhận thức an ninh mạng tốt và không click vào link lạ, kỹ thuật này hoàn toàn vô hiệu.

- Phạm vi tác động bị giới hạn:

+ Trong nhiều kịch bản, XSS chỉ ảnh hưởng đến một nhóm người dùng cụ thể truy cập vào vùng bị nhiễm độc. Khác với Virus hay Worm, XSS thường không có khả năng tự lây lan qua các hệ thống mạng độc lập.

- Thời gian tồn tại ngắn:

+ Đối với Reflected XSS, mã độc chỉ tồn tại trong một phiên truy vấn duy nhất. Khi người dùng tắt trình duyệt hoặc chuyển sang trang khác, cuộc tấn công kết thúc.

- Sự ngăn chặn từ các cơ chế bảo mật hiện đại:

+ XSS Filter: Các trình duyệt hiện đại (Chrome, Edge) tích hợp bộ lọc sẵn để chặn các script nghi vấn trong URL.

+ Content Security Policy (CSP): Nếu website cấu hình CSP chặt chẽ, trình duyệt sẽ từ chối thực thi bất kỳ script nào không nằm trong danh sách trắng (Whitelist), dù hacker có chèn được mã độc vào trang web.

- Phụ thuộc vào lỗ hổng mã nguồn:

+ XSS chỉ có thể thực hiện được trên các ứng dụng web không kiểm tra kỹ dữ liệu đầu vào. Với các framework hiện đại (như React, Angular), việc tự động mã hóa dữ liệu đầu ra đã khiến XSS trở nên khó khai thác hơn rất nhiều.

2.3.3. So sánh hai kỹ thuật

Tiêu chí so sánh	Reflected XSS	Stored XSS
Vị trí lưu trữ	Không lưu trữ trên Server. Mã độc nằm ngay trong URL hoặc Request.	**Được lưu trữ vĩnh viễn** trên cơ sở dữ liệu (Database) của máy chủ.
Vị trí lưu trữ	Phải lừa người dùng click vào đường link chứa mã độc.	Người dùng chỉ cần truy cập vào trang web bị nhiễm là mã độc thực thi.
Tính bền vững	**Tạm thời:** Chỉ tồn tại trong một phiên truy cập duy nhất.	**Lâu dài:** Tồn tại cho đến khi quản trị viên phát hiện và xóa mã độc.
Đối tượng mục tiêu	Thường nhắm vào **một cá nhân** cụ thể qua email hoặc tin nhắn.	Nhắm vào **tất cả người dùng** truy cập vào trang web/ứng dụng đó.
Độ khó thực hiện	Dễ, không cần quyền truy cập vào hệ thống của Server.	Khó hơn, yêu cầu tìm được vị trí nhận và lưu trữ dữ liệu (như comment, profile).
Mức độ nguy hiểm	Trung bình - Cao.	**Rất cao** (Do khả năng lây lan rộng và khó phát hiện).

Chương III: Thực Nghiệm và phân tích kỹ thuật

3.1. Xây dựng môi trường thực nghiệm (Lab)

Để tiến hành thực nghiệm mà không gây ảnh hưởng đến các hệ thống đang vận hành, nhóm đã thiết lập môi trường Lab ảo như sau:

- Hệ điều hành: Windows 10/11 hoặc Kali Linux.

- Phần mềm máy chủ: XAMPP (chứa Apache và MySQL).

- Ứng dụng mục tiêu: DVWA (Damn Vulnerable Web Application) - Một ứng dụng web được thiết kế sẵn các lỗ hổng bảo mật để học tập.

- Cấu hình: Nhóm đã thực hiện sửa đổi file config.inc.php, thiết lập db_user = 'root' và db_password = '' để kết nối với cơ sở dữ liệu MySQL trên XAMPP.

Hình 3.1.1. giao diện của DVWA

3.2. Thực hành khai thác lỗ hổng XSS

3.2.1. Kỹ thuật tạo Payload cơ bản với hàm alert()

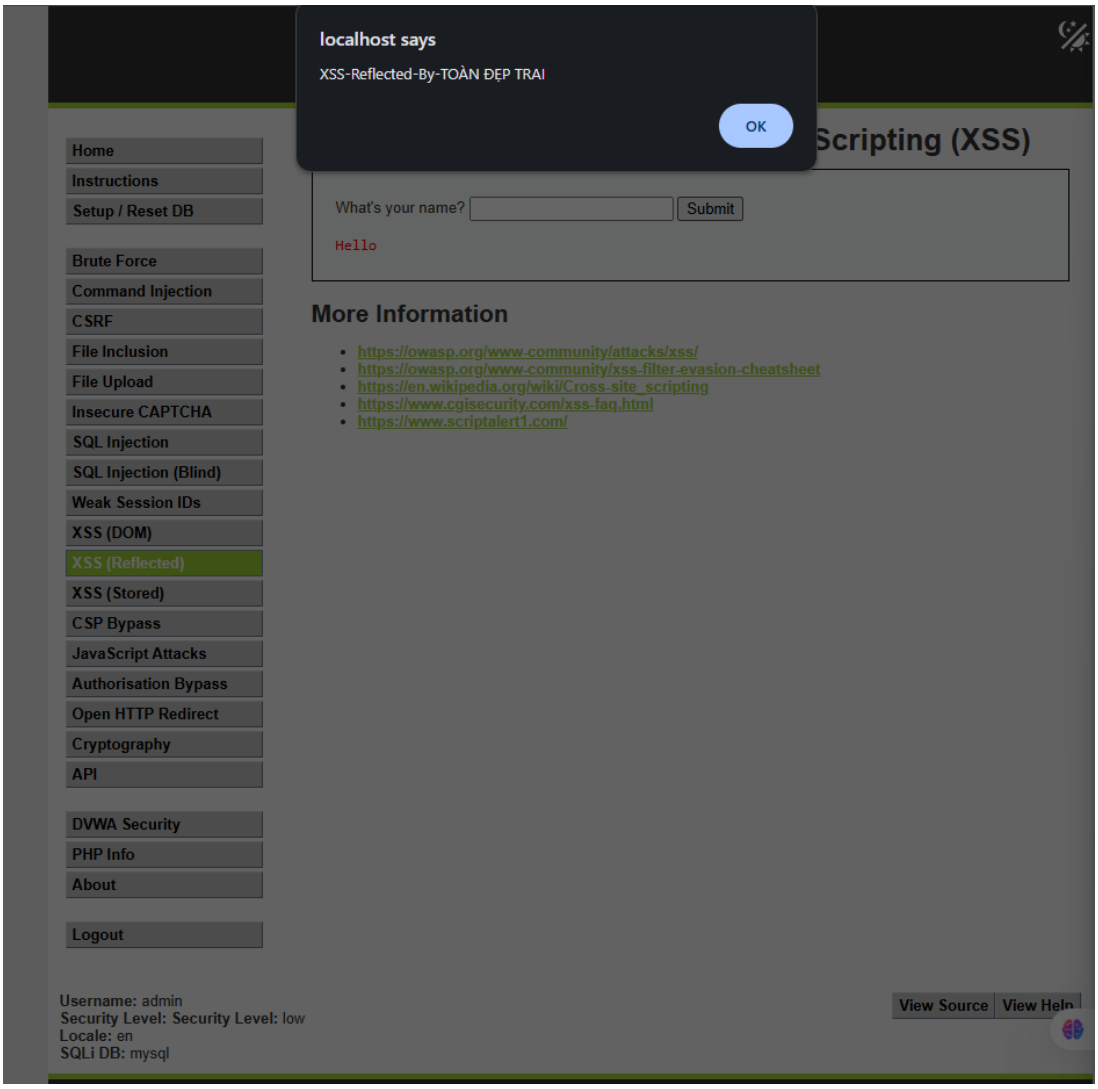
Đây là bước "Proof of Concept" (PoC) để kiểm tra xem website có bị lỗi XSS hay không.

- Kịch bản với Reflected XSS:

+ Vị trí: Ô tìm kiếm (Search box).

+ Payload: <script>alert('XSS-Reflected-By-TOÀN ĐẸP TRAI')</script>

+ Cách thức: Kẻ tấn công gửi một URL chứa payload trên cho nạn nhân. Khi nạn nhân click vào, trình duyệt sẽ thực thi đoạn mã và hiện lên hộp thoại thông báo.



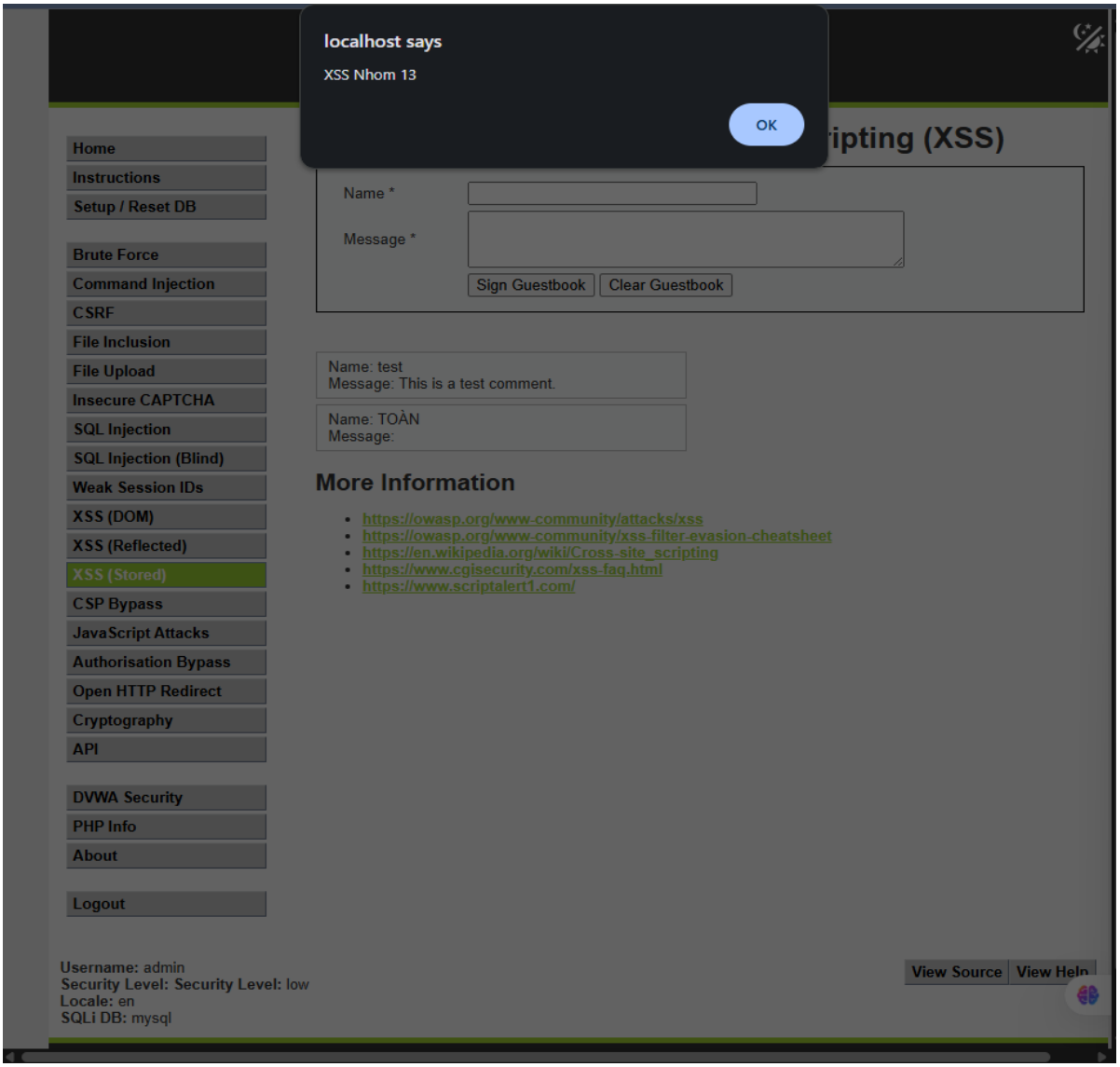
Hình 3.2.1.1. Thực hành Reflected XSS

- Kịch bản với Stored XSS:

+ Vị trí: Phần để lại bình luận (Guestbook) hoặc hồ sơ cá nhân.

+ Payload: <script>alert('XSS-Stored-By-Nhom')</script>

+ Cách thức: Hacker nhập mã vào ô bình luận. Mã này được lưu vào Database. Mọi người dùng sau đó truy cập vào trang bình luận đều sẽ bị hiện thông báo (mã tự động chạy).



Hình 3.2.1.2. Thực hành Stored XSS

3.2.2. Kỹ thuật tấn công đánh cắp phiên làm việc (Cookie Stealing)

Đây là kỹ thuật thực tế và nguy hiểm nhất trong XSS.

+ Mục tiêu**:** Lấy được document.cookie (chứa Session ID) của người dùng và gửi về máy chủ của hacker.

+ Các bước thực hiện:

1. **Chuẩn bị máy chủ nhận tin:** Sử dụng một công cụ như **Webhook.site** hoặc một file log.php đơn giản trên localhost để hứng dữ liệu.
2. **Tạo Payload đánh cắp:**

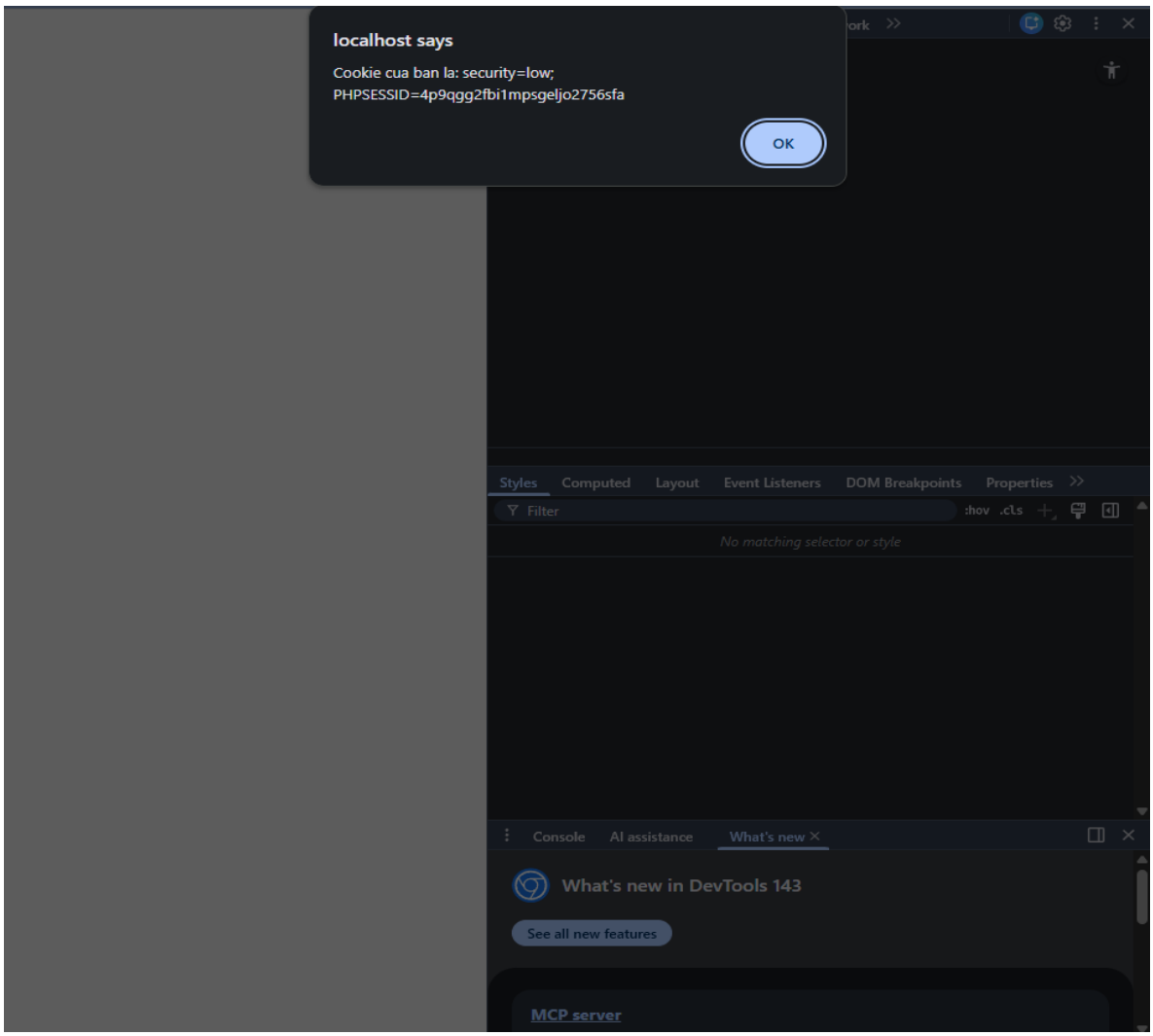
```
JavaScript
<script>
    alert("Cookie cua ban la: " + document.cookie)
</script>
```

3. **Thực thi:**

- Với **Stored XSS**, hacker chèn mã này vào phần bình luận của một bài viết hot.

+ Khi quản trị viên (Admin) vào duyệt bài, trình duyệt của Admin sẽ tự động gửi Cookie quản trị về máy chủ của hacker mà Admin không hề hay biết.

4. **Kết quả:** Hacker sử dụng Cookie vừa lấy được, dùng công cụ chỉnh sửa Cookie (như Cookie-Editor) để chèn vào trình duyệt của mình và truy cập thẳng vào trang quản trị mà không cần mật khẩu.



Hình 3.2.1.2. Thực hành tấn công đánh cắp phiên làm việc (Cookie Stealing)

3.3. Đánh giá tiềm năng tấn công và các hạn chế của kỹ thuật XSS

Sau khi thực nghiệm, nhóm đưa ra các đánh giá kỹ thuật sau:

- Tiềm năng: XSS có thể kết hợp với các kỹ thuật **Phishing** để tạo ra các cuộc tấn công cực kỳ tinh vi, đánh lừa cả những người dùng có kinh nghiệm.

- Hạn chế: * Hiện nay các trình duyệt như Chrome đã có cơ chế chặn script lạ trong URL (XSS Auditor).

Chương IV: Liên hệ các vụ tấn công thực tế ở Việt Nam

4.1. Các vụ tấn công XSS nổi bật tại Việt Nam

Tại Việt Nam, đã xảy ra nhiều vụ tấn công XSS (Cross-Site Scripting) đáng chú ý, trong đó có những sự cố nhằm vào các website lớn và hệ thống quan trọng. Một số trường hợp tiêu biểu gồm:

4.1.1. Tấn công vào các diễn đàn lớn

Một số diễn đàn lớn như Webtretho từng bị tấn công, làm gián đoạn hoạt động và gây khó khăn cho người dùng khi truy cập. Vụ việc này xuất phát từ việc không bảo mật chặt chẽ các phần nhập liệu, cho phép kẻ tấn công chèn mã độc để thực thi.

Diễn đàn cho mẹ và bé Webtretho là địa điểm đông đúc tập trung thành viên nữ giới. Một chủ đề được bàn tán sôi nổi tại diễn đàn này trong mùa hè năm vừa qua là thực trạng phức tạp tại khu du lịch Sầm Sơn. Nhiều thành viên diễn đàn này đã có ý kiến và phản hồi phê phán cách kinh doanh không tích cực tại Sầm Sơn khiến một thành viên không hài lòng và đe dọa đánh sập Webtretho. Diễn đàn này sau đó gặp trục trặc, việc truy cập và đăng nhập gặp nhiều khó khăn. Vụ tấn công khiến chủ đề về Sầm Sơn càng tăng độ nóng.

4.1.2. Lỗ hổng lưu trữ XSS trên các trang web nội địa

Các trang web thường xuyên đối mặt với lỗ hổng XSS lưu trữ (Stored XSS), nơi mã độc được chèn và lưu trữ trực tiếp trên hệ thống, khiến thông tin cá nhân của người dùng bị lộ hoặc bị lợi dụng để phát tán mã độc thêm.

4.1.3. Tấn công có chủ đích vào cơ sở trọng yếu

Một số vụ tấn công nhắm vào các cơ sở dữ liệu trọng yếu đã được ghi nhận trong năm 2023. Các hacker lợi dụng lỗ hổng XSS để chiếm quyền điều khiển, đánh cắp dữ liệu nhạy cảm, hoặc thực thi các cuộc tấn công phức tạp hơn.

4.1.4. Hệ quả và nguy cơ từ việc bảo mật yếu

Việc các tổ chức và doanh nghiệp không thường xuyên kiểm tra, khắc phục lỗ hổng trên website đã tạo điều kiện cho tấn công XSS xảy ra, làm gia tăng nguy cơ mất mát dữ liệu và ảnh hưởng nghiêm trọng đến người dùng. Những vụ việc này nhấn mạnh tầm quan trọng của việc kiểm tra và bảo mật các hệ thống web để giảm thiểu rủi ro từ tấn công XSS. Việc cập nhật các bản vá, sử dụng công cụ giám sát an ninh mạng, và đào tạo nhân sự chuyên trách là rất cần thiết để đảm bảo an toàn không gian mạng.

4.2. Nguyên nhân

Cross-Site Scripting (XSS) xảy ra khi một ứng dụng web cho phép kẻ tấn công chèn mã độc (thường là JavaScript) vào trang web mà người dùng khác truy cập. Những nguyên nhân chính dẫn đến các vụ tấn công XSS bao gồm:

4.2.1. Thiếu kiểm tra và lọc dữ liệu đầu vào

- Nguyên nhân: Ứng dụng không kiểm tra và lọc dữ liệu mà người dùng nhập vào trước khi hiển thị trên trang web.
- Ví dụ: Một trường tìm kiếm hoặc biểu mẫu nhập liệu không loại bỏ hoặc mã hóa các ký tự đặc biệt như <, >, ", '.

4.2.2. Không mã hóa dữ liệu đầu ra

- Nguyên nhân: Ứng dụng hiển thị dữ liệu từ người dùng mà không mã hóa các ký tự đặc biệt để tránh bị hiểu là mã HTML hoặc JavaScript.
- Ví dụ: Khi hiển thị dữ liệu từ cơ sở dữ liệu lên trang web, ứng dụng không mã hóa các ký tự như < thành < hoặc " thành ".

4.2.3. Quản lý kém về nội dung động

- Nguyên nhân: Ứng dụng chèn trực tiếp nội dung động (dynamic content) vào HTML hoặc JavaScript mà không xử lý đúng cách.
- Ví dụ: Một đoạn mã JavaScript trong ứng dụng sử dụng dữ liệu từ người dùng như: + javascript

+ document.write(userInput);

4.2.4. Sử dụng thư viện hoặc plugin không an toàn

- Nguyên nhân: Thư viện hoặc plugin bên thứ ba có thể chứa lỗ hổng cho phép thực hiện XSS.
- Ví dụ: Các công cụ xử lý dữ liệu hoặc giao diện không đảm bảo mã hóa đúng cách dữ liệu người dùng.

4.2.5. Không cấu hình đúng CSP (Content Security Policy)

- Nguyên nhân: Không thiết lập hoặc thiết lập sai chính sách CSP, cho phép mã JavaScript từ nguồn không đáng tin cậy được thực thi.
- Ví dụ: Một ứng dụng cho phép nhúng mã từ các miền bên ngoài mà không kiểm tra nguồn gốc.

4.2.6. Lỗ hổng trong cơ chế xác thực và phân quyền

- Nguyên nhân: Một kẻ tấn công có thể chèn mã độc vào tài khoản của mình và lợi dụng quyền hạn để phát tán mã độc.
- Ví dụ: Các bài viết hoặc bình luận từ người dùng không được kiểm duyệt hoặc lọc trước khi hiển thị.

4.2.7. Không cập nhật và vá lỗi kịp thời

- Nguyên nhân: Sử dụng phiên bản cũ của framework hoặc phần mềm với lỗ hổng XSS đã biết.
- Ví dụ: Một framework không xử lý đúng cách các chuỗi truy vấn hoặc thông tin đầu vào của người dùng.

4.2.8. Thiếu kiểm thử bảo mật

- Nguyên nhân: Không thực hiện kiểm thử bảo mật thường xuyên để phát hiện các điểm yếu XSS trong ứng dụng.
- Ví dụ: Không thực hiện kiểm tra hộp đen (black-box testing) hoặc hộp trắng (white-box testing) để xác định các lỗ hổng.

4.3. Tác động

- Khi một trang web bị tấn công bằng cross-site scripting, một loạt các vấn đề có thể nhanh chóng xuất hiện.

* Một trong số những vấn đề có thể gặp phải bao gồm:

+ Thông tin nhạy cảm của người dùng bị tiết lộ
+ Kẻ tấn công chiếm quyền truy cập vào tài khoản trực tuyến và giả mạo người dùng
+ Phá hoại nội dung trang web
+ Tải lên các chương trình mã độc như Trojan-horse
+ Chuyển hướng trang web, dẫn người dùng đến các trang web nguy hiểm

- Cross-site scripting có thể gây thiệt hại nghiêm trọng cho tổ chức nếu không được phát hiện và xử lý nhanh chóng. Cả doanh nghiệp và khách hàng đều có nguy cơ bị tấn công XSS. Uy tín và các mối quan hệ có thể bị ảnh hưởng tiêu cực sau khi xảy ra một cuộc tấn công malware thành công.

-Vào Mùa Giáng Sinh năm 2018, một cuộc tấn công XSS quy mô lớn đã xảy ra trên các trang web bán lẻ trực tuyến. Phần mềm độc hại đánh cắp thẻ tín dụng có tên" Magecart". Phần mềm này tận dụng lỗ hổng bằng cách tự chèn vào các trang thanh toán trực tuyến. Và đây là lần đầu tiên một cuộc tấn công như vậy xảy ra trên quy mô lớn. Thông tin thẻ tín dụng của người dùng có thể đã được tải lên một máy chủ do kẻ tấn công kiểm soát. Và có thể bị bán hoặc sử dụng cho các giao dịch gian lận.

4.4. Đề xuất giải pháp

- Với nhiều hình thức của các cuộc tấn công XSS. Các tổ chức cần biết cách tự bảo vệ mình một cách đầy đủ và ngăn chặn các vấn đề trong tương lai. Các trang web đang bắt đầu trở nên khó giám sát chặt chẽ hơn bao giờ hết. Do chúng ngày càng trở nên phức tạp. Tần suất các cuộc tấn công có thể sẽ tiếp tục gia tăng theo thời gian.

Những đề xuất sau đây có thể giúp bảo vệ người dùng của bạn chống lại các

cuộc tấn công XSS:

4.4.1. Sử dụng phương pháp kiểm tra và xử lý dữ liệu đầu vào

- Hãy đảm bảo rằng tất cả dữ liệu đầu vào từ người dùng được kiểm tra và xử lý đúng cách trước khi hiển thị trên trang web. Sử dụng các bộ lọc và thư viện bảo mật để loại bỏ hoặc mã hóa các ký tự đặc biệt.

Ví dụ: Giả sử trang web có tính năng cho phép người dùng thêm bình luận vào một bài viết. Nếu trang web không kiểm tra và xử lý đúng cách dữ liệu đầu vào, một kẻ tấn công có thể chèn mã độc vào bình luận.

Bình luận của người dùng: <script>alert('XSS Attack!');</script>

- Giải pháp phòng ngừa: Trước khi lưu trữ bình luận, ứng dụng web nên kiểm tra và làm sạch dữ liệu đầu vào từ người dùng. Sử dụng các bộ lọc như HTML escaping để chuyển đổi các ký tự đặc biệt như <, >, &, " thành các phiên bản HTML an toàn (<, >, &, "). Khi hiển thị bình luận lên trang, trình duyệt sẽ hiểu đó chỉ là dữ liệu văn bản, không thực thi mã độc.

4.4.2. Sử dụng cơ chế tạo mã độc thân thiện

Ví dụ: Trang web cho phép người dùng tạo một tên người dùng mới khi đăng ký tài khoản.

Tên người dùng mới: <script>malicious_code_here</script>

- Giải pháp phòng ngừa: Trong quá trình tạo tên người dùng mới, trang web nên sử dụng whitelist-based filtering để chỉ chấp nhận các ký tự và từ hợp lệ. Ví dụ, chỉ cho phép ký tự từ A-Z, a-z, số từ 0-9, và một số ký tự đặc biệt an toàn như dấu gạch dưới (_) hay dấu chấm (.), nhưng không cho phép các ký tự đặc biệt như <, >, ", v.v. Điều này đảm bảo rằng không có mã độc nào được chèn vào tên người dùng.

4.4.3. Thiết lập header HTTP hợp lệ

Ví dụ: Thiết lập Content Security Policy (CSP) để ngăn chặn việc thực thi mã độc từ các nguồn không tin cậy.

- Giải pháp phòng ngừa: Trang web nên thêm header CSP vào trong HTTP

response để chỉ định những nguồn nào được phép chạy mã độc. Ví dụ, có thể cấu hình CSP như sau:

- Content-Security-Policy: default-src ‘self’; script-src ‘self’ ‘trusted-cdn.com’;

Trong đó, ‘self’ cho phép chạy mã độc từ cùng một trang web, ‘trusted-cdn.com’ cho phép chạy mã độc từ một trang web tin cậy. Bất kỳ mã độc từ nguồn khác sẽ bị ngăn chặn và không được thực thi.

4.4.4. Đào tạo và nâng cao nhận thức bảo mật

- Giải pháp phòng ngừa: Đào tạo nhân viên, đặc biệt là nhà phát triển và quản trị viên hệ thống, về các nguy cơ bảo mật, cách tấn công XSS xảy ra và cách phòng ngừa. Nhân viên nên biết cách sử dụng các công cụ bảo mật và thư viện để đảm bảo mã nguồn của trang web được an toàn và không chứa lỗ hổng bảo mật.

4.4.5. Hạn chế sử dụng dữ liệu của người dùng

- Chỉ sử dụng dữ liệu của người dùng khi thực sự cần thiết. Giảm thiểu nguy cơ mã độc tiềm ẩn.

4.4.6. Sử dụng Chính sách bảo mật nội dung

- Cung cấp mức độ bảo vệ cao hơn và giảm nhẹ rủi ro của các cuộc tấn công XSS. Bằng cách sử dụng các công cụ chuyên dụng để tìm ra các điểm yếu trong phần mềm. Kịp thời vá lỗi trước khi bị tin tặc khai thác.

Thường xuyên sử dụng công cụ quét lỗ hổng ứng dụng web. Để xác định các lỗ hổng XSS trong phần mềm của bạn.

- Bằng cách thực hiện các biện pháp phòng ngừa như trên, trang web và ứng dụng của bạn sẽ có khả năng chống lại tấn công XSS hiệu quả và bảo vệ người dùng và dữ liệu trước các nguy cơ an ninh ứng dụng web.

KẾT LUẬN

Trong bối cảnh chuyển đổi số mạnh mẽ, kỹ thuật tấn công Cross-Site Scripting (XSS) đã và đang khẳng định vị thế là một trong những mối đe dọa hàng đầu đối với an toàn thông tin toàn cầu. Thông qua quá trình nghiên cứu lý thuyết và trực tiếp xây dựng môi trường thực nghiệm trên nền tảng DVWA và XAMPP, bài tiểu luận đã làm sáng tỏ cơ chế vận hành, các biến thể nguy hiểm cũng như những hệ lụy khôn lường mà lỗ hổng này gây ra.

Kết quả thực nghiệm cho thấy XSS không đơn thuần là lỗi kỹ thuật nhỏ mà là hệ quả của việc xem nhẹ quy trình kiểm soát dữ liệu. Việc kẻ tấn công có thể dễ dàng đánh cắp thông tin định danh (Cookie) hay thay đổi nội dung trang web ngay trên môi trường bảo mật thấp (Low Security) là lời cảnh báo đắt giá cho các nhà phát triển ứng dụng. Những sự cố thực tế tại Việt Nam hay các cuộc tấn công quy mô lớn như Magecart đã minh chứng rằng: chỉ một sơ suất trong việc lọc dữ liệu đầu vào hoặc mã hóa dữ liệu đầu ra cũng đủ để phá vỡ toàn bộ hàng rào bảo mật của một tổ chức.

Tuy nhiên, thông qua việc phân tích các cơ chế phòng vệ nâng cao như Content Security Policy (CSP) hay các kỹ thuật Input Sanitization (làm sạch dữ liệu), chúng ta có thể khẳng định rằng XSS hoàn toàn có thể được kiểm soát. Chìa khóa để đối phó với thách thức này không chỉ nằm ở công nghệ, mà còn ở tư duy "Security by Design" – đưa bảo mật vào mọi giai đoạn của vòng đời phát triển phần mềm.

Khép lại bài tiểu luận, nhóm hy vọng những phân tích kỹ thuật và bằng chứng thực nghiệm thu được sẽ góp phần nâng cao nhận thức bảo mật cho cộng đồng lập trình viên và người dùng cuối. Trong một thế giới mạng đầy biến động, việc thấu hiểu kỹ thuật tấn công chính là nền tảng vững chắc nhất để xây dựng những hệ thống web an toàn, tin cậy và bền vững cho tương lai.

DANH MỤC TÀI LIỆU THAM KHẢO

- CaféF. (21/12/2023). *13.900 vụ tấn công mạng vào các tổ chức tại Việt Nam trong năm 2023*. Truy cập ngày 05/01/2026, từ: https://cafef.vn/13900-vu-tan-cong-mang-vao-cac-to-chuc-tai-viet-nam-trong-nam-2023-188231212134606736.chn

- Phạm Lê. (23/06/2023). *6 tháng đầu năm 2023: 5.100 vụ tấn công an ninh mạng vào các hệ thống tại Việt Nam*. Truy cập ngày 05/01/2026, từ: https://genk.vn/6-thang-dau-nam-2023-5100-vu-tan-cong-an-ninh-mang-vao-cac-he-thong-tai-viet-nam-20230623150251914.chn

- MI2. (n.d.). *Vulnerability management là gì? Phương pháp quản lý lỗ hổng bảo mật*. Truy cập ngày 05/01/2026, từ: https://mi2.com.vn/vulnerability-management-la-gi-phuong-phap-quan-ly-lo-hong-bao-mat/

- Kaspersky. (20/11/2024). *Các mối đe dọa an ninh mạng tại Việt Nam gia tăng đáng kể trong quý III năm 2024*. Truy cập ngày 05/01/2026, từ: https://kaspersky.nts.com.vn/cac-moi-de-doa-an-ninh-mang-tai-viet-nam-gia-tang-dang-ke-trong-quy-3-2024

- Nhân Hòa. (03/01/2012). *Những vụ tấn công mạng nổi bật tại Việt Nam trong năm qua*. Truy cập ngày 05/01/2026, từ: https://nhanhoa.com/tintuc/nhung-vu-tan-cong-mang-noi-bat-tai-viet-nam-trong-nam-qua.html

- VIBLO. (2018). *Kỹ thuật tấn công XSS và cách ngăn chặn*. Truy cập ngày 05/01/2026, từ: https://viblo.asia/p/ky-thuat-tan-cong-xss-va-cach-ngan-chan-YWOZr0Py5Q0

- FPT Shop. (2024). *XSS là gì? Tất tần tật thông tin về kỹ thuật tấn công XSS và giải pháp ngăn chặn hiệu quả*. Truy cập ngày 05/01/2026, từ: https://fptshop.com.vn/tin-tuc/danh-gia/xss-la-gi-172513

- VIBLO. (2018). *Kỹ thuật tấn công XSS*. Truy cập ngày 05/01/2026, từ: https://viblo.asia/p/ky-thuat-tan-cong-xss-924lJDRzKPM

- PortSwigger. (2024). *Cross-site scripting*. Truy cập ngày 05/01/2026, từ: https://portswigger.net/web-security/cross-site-scripting

- Viettel IDC. (2024). *XSS là gì? Cách kiểm tra và ngăn chặn tấn công hiệu quả*. Truy cập ngày 05/01/2026, từ: https://viettelidc.com.vn/tin-tuc/xss-la-gi-cach-kiem-tra-va-ngan-chan