**PHÂN BIỆT XSS VÀ SQL INJECTION**

**Thành viên nhóm 6:**

1. **Dương Văn Sơn**
2. **Trần Bình Minh**
3. **Nguyễn Đăng Hải Nam**
4. **Hoàng Gia Minh**

**GV: ThS. Dương Thúy Hường**

**MỤC LỤC**

[CHƯƠNG 1: TÌM HIỂU VỀ XSS 3](#_Toc51759323)

[1.1 Khái niệm 3](#_Toc51759324)

[1.2 Mục đích 3](#_Toc51759325)

[1.3 Các kiểu tấn công XSS 4](#_Toc51759326)

[1.3.1 Stored-XSS 4](#_Toc51759327)

[1.3.2 Reflected XSS 4](#_Toc51759328)

[1.3.3 DOM Based XSS 6](#_Toc51759329)

[1.4 Phương pháp phòng chống, ngăn chặn 7](#_Toc51759330)

[CHƯƠNG 2: Tìm hiểu về SQL - Injection 9](#_Toc51759331)

[2.1 Khái niệm: 9](#_Toc51759332)

[2.2 Mục đích 9](#_Toc51759333)

[2.3 Các kiểu tấn công sql injection 9](#_Toc51759334)

[2.3.1 In-band SQLi 10](#_Toc51759335)

[2.3.2 Inferential SQLi (Blind SQLi) 11](#_Toc51759336)

[2.3.3 Out-of-band SQLi 12](#_Toc51759337)

[2.4 Phương pháp phòng chống, ngăn chặn 13](#_Toc51759338)

[CHƯƠNG 3: Phân biệt XSS và SQL-injection 15](#_Toc51759339)

[TÀI LIỆU THAM KHẢO 17](#_Toc51759340)

# TÌM HIỂU VỀ XSS

## Khái niệm

* Cross-Site Scripting (XSS) là một trong những kĩ thuật tấn công phổ biến nhất hiện nay. Được mệnh danh là Godfather of Attack và trong nhiều năm liền được liệt vào danh sách những kỹ thuật tấn công nguy hiểm nhất với các ứng dụng web.
* Cross-Site Scripting không gọi tắt là CSS để tránh nhầm lẫn với khái niệm Cascading Style Sheet của HTML.
* Tấn công XSS là một đoạn mã độc, 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.
* Tấn công Cross Site Scripting nghĩa là gửi và chèn lệnh và 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.

## Mục đích

Mục đích chính của tấn công XSS là ăn cắp dữ liệu nhận dạng của người dùng như: cookies, session tokens và các thông tin khác. Trong hầu hết các trường hợp, cuộc tấn công này đang được sử dụng để ăn cắp cookie của người khác. Như chúng ta biết, cookie giúp chúng ta đăng nhập tự động. Do đó với cookie bị đánh cắp, chúng tai có thể đăng nhập bằng các thông tin nhận dạng khác. Và đây là một trong những lý do, tại sao tấn công XSS được coi là một trong những cuộc tấn công nguy hiểm nhất.

Thay đổi cấu trúc của trang web.

## Các kiểu tấn công XSS

Có 3 loại XSS là: Reflected XSS, Stored XSS và DOM-based XSS

### Stored-XSS

Stored XSS là dạng tấn công mà hacker chèn trực tiếp các mã độc vào cơ sở dữ liệu của website. Dạng tấn công này xảy ra khi các dữ liệu được gửi lên server không được kiểm tra kỹ lưỡng mà lưu trực tiếp vào cơ sở dữ liệu. Khi người dùng truy cập vào trang web này thì những đoạn script độc hại sẽ được thực thi chung với quá trình load trang web.

Ví dụ như chức năng comment trong blog, website, message. Khi người dùng post 1 comment chứa 1 đoạn script độc hại mà server cho phép nó lưu vào database thì khi trang web load các comment của người dùng cũng sẽ load ra cả đoạn script độc hại, đoạn script đó sẽ chạy trên các máy client và thực thi.

**Mở rộng:**

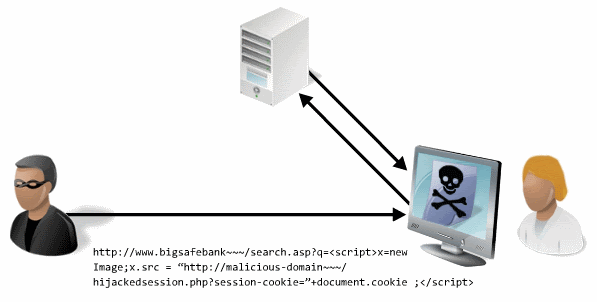
Trường hợp nếu một trang đăng nhập bị dính lỗi bảo mật XSS sẽ nguy hiểm như nào? Hacker/Attacker có thể thay đổi action sang một tệp PHP khác xử lý các giá trị nhận được bao gồm tài khoản, mật khẩu thậm chí là các thông tin nhạy cảm khác tùy vào nơi dính lỗ hổng XSS là khu vực đăng nhập, đăng ký, thanh toán,… Hacker/Attacker hoàn toàn có thể lưu lại (log) sau đó giả mạo việc đăng nhập bằng cách gửi trả (POST) dữ liệu tới tệp tin xử lý thật của Form ban đầu.

### Reflected XSS

Reflected XSS là dạng tấn công thường gặp nhất trong các loại hình XSS. Với Reflected XSS, hacker không gửi dữ liệu độc hại lên server nạn nhân, mà gửi trực tiếp link có chứa mã độc cho người dùng, khi người dùng click vào link này thì trang web sẽ được load chung với các đoạn script độc hại. Reflected XSS thường dùng để ăn cắp cookie, chiếm session,… của nạn nhân hoăc cài keylogger, trojan … vào máy tính nạn nhân.

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.

Dạng tấn công bằng Reflected XSS được mô tả như sau:



Trước tiên, hacker sẽ gửi cho nạn nhân một đường link có chứa mã độc hại đi kèm, ví dụ:

*http://victim.com/index.php?id=<script>alert(document.cookie) script>*

Để cho người dùng khó phát hiện nên khi gửi đường link trên cho nạn nhân, hacker có thể sẽ mã hoá nó thành những ký tự lạ khó đọc, ví dụ:

Cross-Site Scripting (XSS) là gì? - Ảnh 11.

Như vậy, nạn nhân sẽ không nghi ngờ đường link lạ, và click vào link.

Khi nạn nhân click vào đường link được hacker gửi, trình duyệt sẽ load trang web và thực thi các đoạn script kèm theo, sau đó gửi về cho hacker những thông tin của nạn nhân.

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.

**Reflected XSS và Stored XSS có 2 sự khác biệt lớn trong quá trình tấn công.**

* Thứ nhất, để khai thác Reflected XSS, hacker phải lừa được nạn nhân truy cập vào URL của mình. Còn Stored XSS không cần phải thực hiện việc này, sau khi chèn được mã nguy hiểm vào CSDL của ứng dụng, hacker chỉ việc ngồi chờ nạn nhân tự động truy cập vào. Với nạn nhân, việc này là hoàn toàn bình thường vì họ không hề hay biết dữ liệu mình truy cập đã bị nhiễm độc.
* Thứ 2, mục tiêu của hacker sẽ dễ dàng đạt được hơn nếu tại thời điểm tấn công nạn nhân vẫn trong phiên làm việc(session) của ứng dụng web. Với Reflected XSS, hacker có thể thuyết phục hay lừa nạn nhân đăng nhập rồi truy cập đến URL mà hắn ta cung cấp để thực thi mã độc. Nhưng Stored XSS thì khác, vì mã độc đã được lưu trong CSDL Web nên bất cứ khi nào người dùng truy cập các chức năng liên quan thì mã độc sẽ được thực thi, và nhiều khả năng là những chức năng này yêu cầu phải xác thực(đăng nhập) trước nên hiển nhiên trong thời gian này người dùng vẫn đang trong phiên làm việc.

Từ những điều này có thể thấy Stored XSS nguy hiểm hơn Reflected XSS rất nhiều, đối tượng bị ảnh hưởng có thể là tất cả nhưng người sử dụng ứng dụng web đó. Và nếu nạn nhân có vai trò quản trị thì còn có nguy cơ bị chiếm quyền điều khiển web.

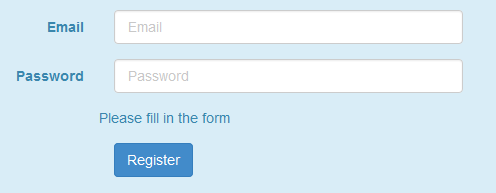
### DOM Based XSS

DOM Based XSS là 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*](http://example.com/register.php?message=Please)fill in the form

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



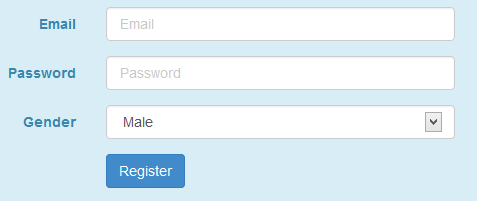
Thay vì truyền message=Please fill in the form thì truyền

*message=<label>Gender</label>*

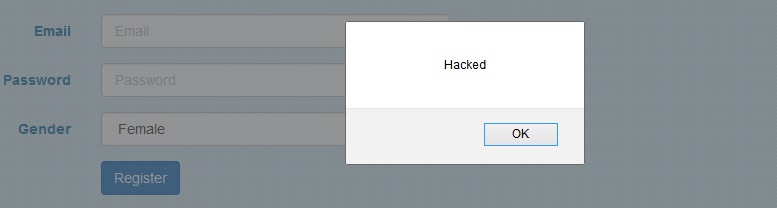
*<select class = "form-control" onchange="java\_script\_:show()"><option value="Male">Male</option><option value="Female">Female</option></select>*

*<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



## Phương pháp phòng chống, ngăn chặn

Sau đây là những phương pháp phổ biến nhất để ngăn chặn Cross-site scripting:

* Escape đầu vào của người dùng là một cách để ngăn các lỗ hổng XSS xuất hiện trong các ứng dụng. Điều này có nghĩa là lấy dữ liệu mà ứng dụng đã nhận được và đảm bảo an toàn trước khi hiển thị nó cho người dùng. Escape input từ người dùng sẽ khiến các ký tự quan trọng trong dữ liệu mà trang web nhận được không bị hiểu là mã thực thi. Nhờ đó ngăn trình duyệt diễn giải các ký tự được sử dụng để báo hiệu bắt đầu hoặc kết thúc mã thực thi và dịch chúng thành "escaped". Ví dụ, các ký tự như dấu ngoặc kép, dấu ngoặc đơn, dấu ngoặc vuông và một số dấu chấm câu khác đôi khi được sử dụng để làm nổi bật mã thực thi. Escape các ký tự này có nghĩa là chuyển đổi chúng từ các ký tự đơn thành các chuỗi khi trình duyệt biên dịch.
* Sàng lọc đầu vào của người dùng là một cách khác để ngăn chặn các cuộc tấn công Cross-site scripting, đặc biệt hữu ích trên các trang web cho phép HTML markup. Phương pháp ngăn chặn này sẽ xóa sạch dữ liệu chứa các ký tự có khả năng thực thi, thay đổi đầu vào người dùng thành định dạng có thể chấp nhận và đảm bảo dữ liệu nhận được không thể được hiểu là mã thực thi.
* Xác thực đầu vào đảm bảo việc ứng dụng hiển thị dữ liệu chính xác và ngăn dữ liệu độc hại gây hại cho trang web, cơ sở dữ liệu và người dùng. Xác thực đầu vào giúp ngăn XSS trong các biểu mẫu vì nó ngăn người dùng thêm các ký tự đặc biệt vào các trường nhập dữ liệu trang web bằng cách từ chối yêu cầu. Xác thực đầu vào giúp giảm khả năng gây ảnh hưởng tiêu cực, nếu kẻ tấn công phát hiện ra một lỗ hổng XSS ở đâu đó.

# Tìm hiểu về SQL - Injection

## Khái niệm:

SQL Injection là một kỹ thuật lợi dụng những lỗ hổng về câu truy vấn của các ứng dụng. Được thực hiện bằng cách chèn thêm một đoạn [SQL](https://topdev.vn/blog/sql-la-gi/) để làm sai lệnh đi câu truy vấn ban đầu, từ đó có thể khai thác dữ liệu từ database.

SQL injection có thể cho phép những kẻ tấn công thực hiện các thao tác như một người quản trị web, trên cơ sở dữ liệu của ứng dụng, thậm chí là server mà ứng dụng đó đang chạy. SQL injection thường được biết đến như là một vật trung gian tấn công trên các ứng dụng web có dữ liệu được quản lý bằng các hệ quản trị cơ sở dữ liệu như SQL Server, MySQL, Oracle, DB2, Sysbase… Công cụ dùng để tấn công là một trình duyệt web bất kì, chẳng hạn như Internet Explorer, Netscape, Lynx, Firefox,…

## Mục đích

- Hack tài khoản cá nhân của người khác.

- Ăn cắp hoặc sao chép dữ liệu của trang web hoặc hệ thống.

- Thay đổi dữ liệu nhạy cảm của hệ thống.

- Xóa dữ liệu nhạy cảm và quan trọng của hệ thống.

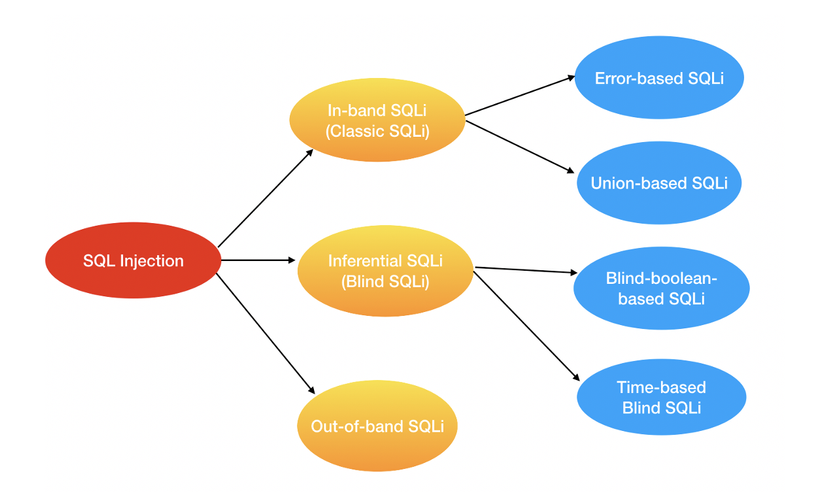
- Người dùng có thể đăng nhập vào ứng dụng với tư cách người dùng khác, ngay cả với tư cách quản trị viên.

- Người dùng có thể xem thông tin cá nhân thuộc về những người dùng khác, ví dụ chi tiết hồ sơ của người dùng khác, chi tiết giao dịch của họ,...

- Người dùng có thể sửa đổi cấu trúc của cơ sở dữ liệu, thậm chí xóa các bảng trong cơ sở dữ liệu ứng dụng.

- Người dùng có thể kiểm soát máy chủ cơ sở dữ liệu và thực thi lệnh theo ý muốn.

## Các kiểu tấn công sql injection



SQL Injection có thể chia nhỏ thành các dạng sau:

### In-band SQLi

Đây là dạng tấn công phổ biến nhất và cũng dễ để khai thác lỗ hổng SQL Injection nhất

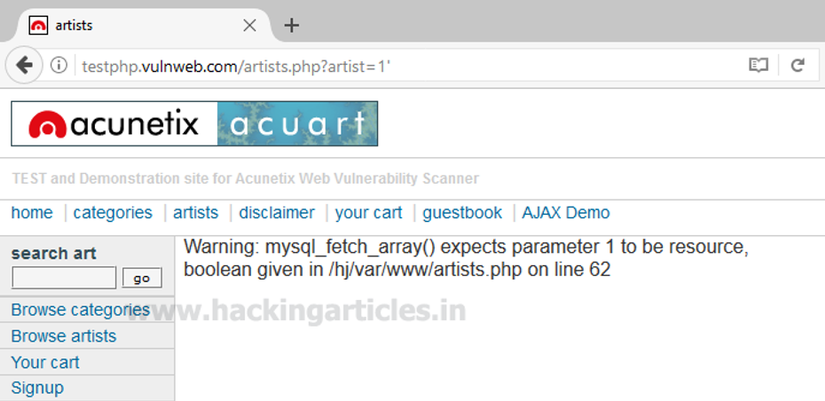
Xảy ra khi hacker có thể tổ chức tấn công và thu thập kết quả trực tiếp trên cùng một kênh liên lạc

In-Band SQLi chia làm 2 loại chính:

ERROR-BASED SQLI

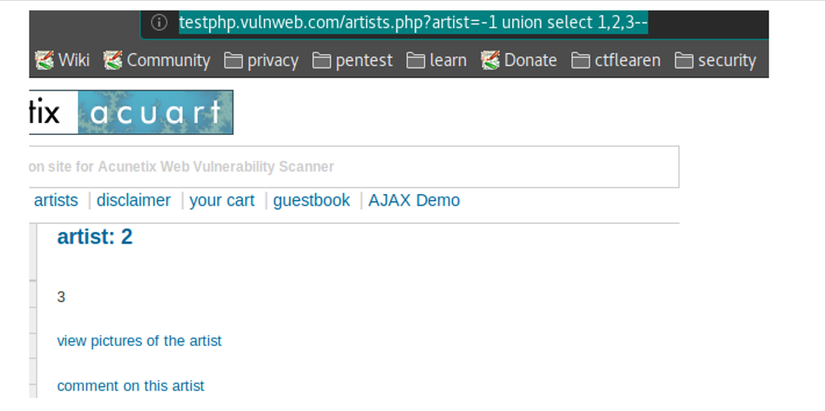
Là một kỹ thuật tấn công SQL Injection dựa vào thông báo lỗi được trả về từ Database Server có chứa thông tin về cấu trúc của cơ sở dữ liệu.

Trong một vài trường hợp, chỉ một mình Error-based là đủ cho hacker có thể liệt kê được các thuộc tính của cơ sở dữ liệu



UNION-BASED SQLI

Là một kỹ thuật tấn công SQL Injection dựa vào sức mạnh của toán tử UNION trong ngôn ngữ SQL cho phép tổng hợp kết quả của 2 hay nhiều câu truy vấn SELECTION trong cùng 1 kết quả và được trả về như một phần của HTTP response



### Inferential SQLi (Blind SQLi)

Không giống như In-band SQLi, Inferential SQL Injection tốn nhiều thời gian hơn cho việc tấn công do không có bất kì dữ liệu nào được thực sự trả về thông qua web application và hacker thì không thể theo dõi kết quả trực tiếp như kiểu tấn công In-band

Thay vào đó, kẻ tấn công sẽ cố gắng xây dựng lại cấu trúc cơ sở dữ liệu bằng việc gửi đi các payloads, dựa vào kết quả phản hồi của web application và kết quả hành vi của database server.

Có 2 dạng tấn công chính

Blind-boolean-based

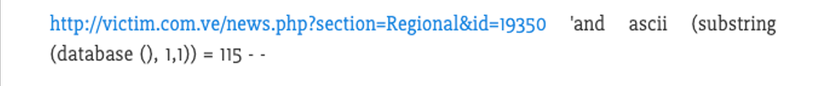
Blind-time-based SQLi

BLIND-BOOLEAN-BASED

Là kĩ thuật tấn công SQL Injection dựa vào việc gửi các truy vấn tới cơ sở dữ liệu bắt buộc ứng dụng trả về các kết quả khác nhau phụ thuộc vào câu truy vấn là True hay False.

Tuỳ thuộc kết quả trả về của câu truy vấn mà HTTP reponse có thể thay đổi, hoặc giữ nguyên

Kiểu tấn công này thường chậm (đặc biệt với cơ sở dữ liệu có kích thước lớn) do người tấn công cần phải liệt kê từng dữ liệu, hoặc mò từng kí tự



TIME-BASED BLIND SQLI

Time-base Blind SQLi là kĩ thuật tấn công dựa vào việc gửi những câu truy vấn tới cơ sở dữ liệu và buộc cơ sở dữ liệu phải chờ một khoảng thời gian (thường tính bằng giây) trước khi phản hồi.

Thời gian phản hồi (ngay lập tức hay trễ theo khoảng thời gian được set) cho phép kẻ tấn công suy đoán kết quả truy vấn là TRUE hay FALSE

Kiểu tấn công này cũng tốn nhiều thời gian tương tự như Boolean-based SQLi

### Out-of-band SQLi

Out-of-band SQLi không phải dạng tấn công phổ biến, chủ yếu bởi vì nó phụ thuộc vào các tính năng được bật trên Database Server được sở dụng bởi Web Application.

Kiểu tấn công này xảy ra khi hacker không thể trực tiếp tấn công và thu thập kết quả trực tiếp trên cùng một kênh (In-band SQLi), và đặc biệt là việc phản hồi từ server là không ổn định

Kiểu tấn công này phụ thuộc vào khả năng server thực hiện các request DNS hoặc HTTP để chuyển dữ liệu cho kẻ tấn công.

Ví dụ như câu lệnh xp\_dirtree trên Microsoft SQL Server có thể sử dụng để thực hiện DNS request tới một server khác do kẻ tấn công kiểm soát, hoặc Oracle Database’s UTL HTTP Package có thể sử dụng để gửi HTTP request từ SQL và PL/SQL tới server do kẻ tấn công làm chủ

**Các trường hợp thường bị tấn công Sql injection**

Bất cứ thao tác nào của ứng dụng có thực hiện truy vấn tới cơ sở dữ liệu đều có thể bị lợi dụng để tấn công Sql injection. Các thao tác cơ bản với CSDL là: select, insert, update đều có thể bị tấn công. Có thể kể ra vài thao tác phổ biến có thể tấn công như:

* Kiểm tra đăng nhập ứng dụng.
* Thao tác lưu comment của user xuống DB.
* Thao tác truy vấn thông tin user.
* …

## Phương pháp phòng chống, ngăn chặn

Mặc dù SQL rất nguy hại nhưng cũng dễ phòng chống. Gần đây, hầu như chúng ta ít viết SQL thuần mà toàn sử dụng ORM (Object-Relational Mapping) framework. Các framework web này sẽ tự tạo câu lệnh SQL nên hacker cũng khó tấn công hơn.

Tuy nhiên, có rất nhiều site vẫn sử dụng SQL thuần để truy cập dữ liệu. Đây chính là mồi ngon cho hacker. Để bảo vệ bản thân trước SQL Injection, ta có thể thực hiện các biện pháp sau.

Lọc dữ liệu từ người dùng: Cách phòng chống này tương tự như XSS. Ta sử dụng filter để lọc các kí tự đặc biệt (; ” ‘) hoặc các từ khoá (SELECT, UNION) do người dùng nhập vào. Nên sử dụng thư viện/function được cung cấp bởi framework. Viết lại từ đầu vừa tốn thời gian vừa dễ sơ sót.

Không cộng chuỗi để tạo SQL: Sử dụng parameter thay vì cộng chuỗi. Nếu dữ liệu truyền vào không hợp pháp, SQL Engine sẽ tự động báo lỗi, ta không cần dùng code để check.

Không hiển thị exception, message lỗi: Hacker dựa vào message lỗi để tìm ra cấu trúc database. Khi có lỗi, ta chỉ hiện thông báo lỗi chứ đừng hiển thị đầy đủ thông tin về lỗi, tránh hacker lợi dụng.

Phân quyền rõ ràng trong DB: Nếu chỉ truy cập dữ liệu từ một số bảng, hãy tạo một account trong DB, gán quyền truy cập cho account đó chứ đừng dùng account root hay sa. Lúc này, dù hacker có inject được sql cũng không thể đọc dữ liệu từ các bảng chính, sửa hay xoá dữ liệu.

Backup dữ liệu thường xuyên: Các cụ có câu “cẩn tắc vô áy náy”. Dữ liệu phải thường xuyên được backup để nếu có bị hacker xoá thì ta vẫn có thể khôi phục được. Còn nếu cả dữ liệu backup cũng bị xoá luôn thì … chúc mừng bạn, update CV rồi tìm cách chuyển công ty thôi!

# Phân biệt XSS và SQL-injection

Sự khác biệt chính giữa tấn công SQL và XSS injection là các cuộc tấn công SQL injection được sử dụng để lấy cắp thông tin từ cơ sở dữ liệu trong khi các cuộc tấn công XSS được sử dụng để chuyển hướng người dùng đến các trang web nơi kẻ tấn công có thể lấy cắp dữ liệu từ chúng. SQL injection là tập trung vào cơ sở dữ liệu trong khi XSS hướng tới việc tấn công người dùng cuối.

Một cuộc tấn công đưa vào SQL xảy ra khi mã ngôn ngữ truy vấn có cấu trúc (SQL) được đưa vào biểu mẫu, cookie hoặc tiêu đề http không sử dụng phương pháp làm sạch hoặc xác thực dữ liệu để xác minh rằng thông tin phù hợp với các tham số GET hoặc POST được chỉ định. Lỗ hổng này cho phép xâm nhập, thay đổi hoặc xóa dữ liệu khỏi cơ sở dữ liệu được kết nối với các trang web. Theo dự án bảo mật ứng dụng web mở (OWASP), năm kỹ thuật chính được sử dụng trong các cuộc tấn công SQL injection là union operator, boolean, error based, out of band và time delay. Dưới đây là một ví dụ về cú pháp SQL injection có thể được sử dụng để truy xuất tất cả số dư tài khoản khách hàng nếu không được giới hạn đúng cách trên trang web: số dư TỪ tài khoản WHERE account\_owner\_id = 0 HOẶC 1 = 1.

Ngược lại, một cuộc tấn công XSS sử dụng mã độc hại để chuyển hướng người dùng đến các trang web độc hại, đánh cắp cookie hoặc thông tin đăng nhập hoặc phá hoại trang web. Điều này thường được thực hiện bằng cách sử dụng các tập lệnh độc hại được thực thi trong trình duyệt máy khách do người dùng nhập, câu lệnh chức năng, yêu cầu máy khách hoặc các biểu thức khác. Ví dụ: những kẻ tấn công có thể tấn công các URL được tạo ra một cách độc hại thông qua các nỗ lực lừa đảo qua email, các tệp đính kèm email có liên kết nhúng, khung trên các trang web hợp pháp và các diễn đàn web mà người dùng mục tiêu thường xuyên truy cập. Trong khi các cuộc tấn công SQL injection nhắm mục tiêu thông tin trong cơ sở dữ liệu back end, các cuộc tấn công XSS tập trung vào việc đánh cắp dữ liệu từ giao diện người dùng của trang web. Theo OWASP, cú pháp sau có thể được sử dụng để thực hiện một cuộc tấn công XSS để lấy cắp dữ liệu cookie nếu không sử dụng xác thực đầu vào:

*<SCRIPT type = ”text / javascript”>*

*Var adr = ‘../evil.php?cakemonster=’ + Escape (document.cookie);*

*</SCRIPT>*

|  |  |  |
| --- | --- | --- |
|  | **XSS** | **SQL-Injection** |
| **Định nghĩa** | Một loại lỗ hổng bảo mật máy tính trong các ứng dụng web cho phép kẻ tấn công đưa các tập lệnh chạy phía máy khách vào các trang web được người dùng khác xem. | Một kỹ thuật chèn mã, tấn công các ứng dụng hướng tới dữ liệu, chèn các câu lệnh SQL vào các trường input để thực thi. |
| **Cách thức** | Lợi dụng lỗ hổng XSS để chèn mã thực thi vào trang web do Web server sinh ra. | Chèn các ký tự đặc biệt hoặc câu lệnh SQL vào giá trị tham số đầu vào để thay đổi ý nghĩa của câu truy vấn CSDL |
| **Ngôn ngữ** | JavaScript,HTML | Ngôn ngữ truy vấn có cấu trúc (SQL) |
| **Mục đích** | Mục đích chính của tấn công XSS là ăn cắp dữ liệu nhận dạng của người dùng như: cookies, session tokens và các thông tin khác | Đọc dữ liệu không được phép trong CSDL  Sửa đổi trái phép  Thực thi mã tấn công từ xa |

# TÀI LIỆU THAM KHẢO

[1] - <https://viblo.asia/p/tim-hieu-ve-ki-thuat-tan-cong-xss-ZK1ov15Lv5b9>

[2] - <https://nhungdongcodevui.com/2016/07/29/bai-1-tan-cong-sql-injection-xss/>

[3] - <https://viblo.asia/p/sql-injection-va-cach-phong-chong-OeVKB410lkW>

[4] - <https://medium.com/@nyomanpradipta120/sql-injection-union-attack-9c10de1a5635>