

## SQL Injection

### SLIDE 1

#### 1.Tổng quan về SQL Injection

**Định nghĩa:**

SQL (Structured Query Language): Là ngôn ngữ truy vấn cấu trúc, được dùng để giao tiếp với Cơ sở dữ liệu (Database).

SQL Injection: Là một kỹ thuật chèn mã (code injection) cho phép kẻ tấn công thực thi các câu lệnh SQL độc hại thông qua việc lợi dụng lỗ hổng của việc kiểm tra dữ liệu đầu vào trong các [Ứng dụng web](#), các thông báo lỗi của hệ quản trị cơ sở dữ liệu.



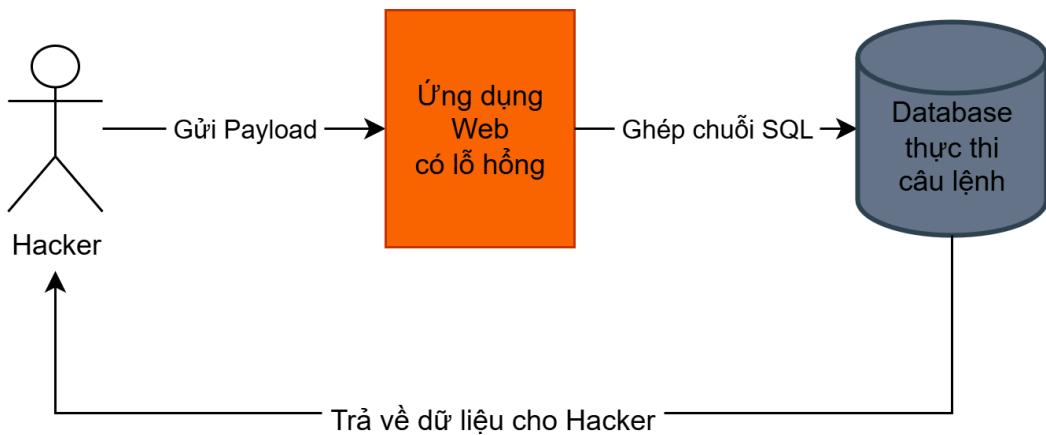
(Nguồn: [habtesoft.medium.com](https://habtesoft.medium.com))

### SLIDE 2

#### Cách thức hoạt động cơ bản:

- Kẻ tấn công tìm một điểm nhập liệu (input) trên ứng dụng (ví dụ: form đăng nhập, thanh tìm kiếm, tham số trên URL).
- Thay vì nhập dữ liệu hợp lệ, kẻ tấn công nhập một đoạn mã SQL độc hại (send payload).
- Ứng dụng web (phía server) ghép nối trực tiếp chuỗi input này vào câu lệnh SQL gốc mà không xử lý.

- Cơ sở dữ liệu thực thi câu lệnh đã bị "tiêm" mã dẫn đến hành vi ngoài ý muốn.

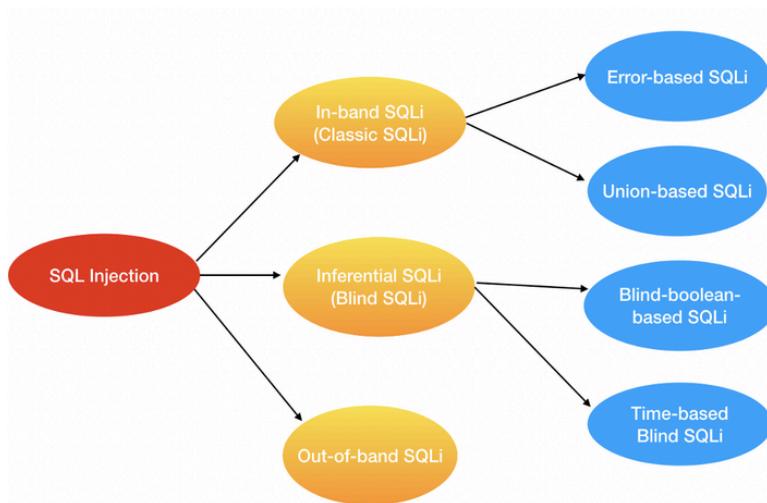


Sơ đồ minh họa luồng tấn công của Hacker

### SLIDE 3

## 2. Phân loại SQL Injection

Có 3 loại hình tấn công chính, dựa trên cách thức ứng dụng phản hồi lại payload



(Nguồn: viblo.asia)

### a. In-band SQLi (Classic SQLi):

Kẻ tấn công sử dụng cùng một kênh để tấn công và nhận kết quả. Đây là loại phổ biến và đơn giản nhất. Phân loại con:

- Error-Based SQLi: Kẻ tấn công khai thác DB báo lỗi. Thông báo lỗi này (nếu bị hiển thị ra ngoài) vô tình tiết lộ thông tin về cấu trúc DB (tên bảng, tên cột).
- UNION-Based SQLi: Kẻ tấn công dùng toán tử UNION để "ghép" kết quả từ một truy vấn độc hại (ví dụ: UNION SELECT username, password FROM users) vào kết quả của truy vấn hợp lệ ban đầu.

### b. Inferential SQLi (Blind SQLi - Tấn công mù):

Ứng dụng không trả về dữ liệu hoặc lỗi. Kẻ tấn công phải "hỏi" DB các câu hỏi True/False và quan sát sự thay đổi trong phản hồi của trang để "đoán" dữ liệu. Phân loại con:

- Boolean-Based SQLi: Dựa vào việc trang trả về nội dung khác nhau (ví dụ: "Sản phẩm tồn tại" / "Sản phẩm không tồn tại").
- Time-Based SQLi: Buộc DB thực hiện một hành động tồn thời gian (ví dụ: SLEEP(5)) nếu một điều kiện là đúng. Kẻ tấn công đo thời gian phản hồi của trang để biết điều kiện là True hay False.

### c. Out-of-band SQLi:

Mô tả: Kỹ thuật nâng cao và ít phổ biến. Sử dụng khi 2 cách trên không hiệu quả. Kẻ tấn công buộc DB gửi dữ liệu đến một máy chủ bên ngoài mà họ kiểm soát (ví dụ: qua DNS request hoặc HTTP request).

## SLIDE 4

### 3. Nguyên nhân

Lập trình viên quá tin tưởng vào dữ liệu đầu vào của người dùng và thiếu các biện pháp kiểm tra dữ liệu đầu vào. Cụ thể:

- String Concatenation:
  - Xây dựng câu lệnh SQL bằng cách ghép chuỗi. Đây là lỗi phổ biến và nghiêm trọng nhất.
  - Ví dụ (Code xấu): query = "SELECT \* FROM users WHERE username = " + user\_input + ";"
- Validate & Sanitize (Không xác thực và làm sạch dữ liệu):
  - Không kiểm tra xem input có chứa các ký tự đặc biệt (như ', --, ;, #) hay không.
  - Không lọc (filter) hoặc thoát (escape) các ký tự này trước khi đưa vào câu lệnh.
- Thiếu cơ chế phòng thủ an toàn:

- Không sử dụng các tính năng bảo mật có sẵn như Prepared Statements (Tham số hóa truy vấn) hoặc Stored Procedures.



(Nguồn: [www.maryville.edu](http://www.maryville.edu))

## SLIDE 5

### 4. Hậu quả

Việc cho phép kẻ tấn công thực thi SQL tùy ý có thể dẫn đến các hậu quả nghiêm trọng:

- Đánh cắp dữ liệu nhạy cảm như tên người dùng, mật khẩu, thông tin cá nhân (CCCD, SĐT), thông tin thẻ tín dụng.
- Authentication Bypass (Vượt qua xác thực): Đăng nhập vào hệ thống với tư cách admin hoặc người dùng bất kỳ mà không cần biết mật khẩu.
- Data Destruction (Phá hoại dữ liệu): Thay đổi (UPDATE), xóa (DELETE) dữ liệu hoặc thậm chí xóa toàn bộ bảng (DROP TABLE) hoặc cơ sở dữ liệu (DROP DATABASE).
- Server Compromise (Chiếm quyền điều khiển máy chủ): Trong một số trường hợp, kẻ tấn công có thể đọc/ghi file trên máy chủ hoặc thực thi các lệnh hệ thống từ đó chiếm toàn bộ quyền kiểm soát máy chủ.



(Nguồn: [forbes.com](http://forbes.com))

## SLIDE 6 + 7 (sửa chèn ảnh + nội dung theo demo)

### 5. Các ví dụ

#### Ví dụ 1 - Vượt qua đăng nhập (UNION-Based)

**Tình huống:** Một trang web có form đăng nhập (username, password).

##### Lệnh SQL phía Server (Lỗi):

```
SELECT * FROM users WHERE username = '[username_input]' AND password = '[password_input]';
```

##### Input của Hacker:

- Ô Username: ' OR '1'='1' --
- Ô Password: (để trống hoặc nhập bất cứ gì)

##### Câu lệnh SQL thực thi tại Database:

```
SELECT * FROM users WHERE username = " OR '1'='1' --' AND password = '...';
```

##### Phân tích:

- Đầu tiên dùng để đóng chuỗi username = ".
- OR '1'='1' được thêm vào. Vì 1=1 luôn ĐÚNG (TRUE), mệnh đề WHERE luôn đúng.
- -- (hoặc # trong MySQL) là ký tự comment. Nó "vô hiệu hóa" phần còn lại của câu lệnh (bao gồm cả việc kiểm tra mật khẩu).

**Kết quả:** Câu lệnh trả về tất cả các hàng (rows) trong bảng users. Ứng dụng thường sẽ đăng nhập cho kẻ tấn công bằng tài khoản đầu tiên trong bảng (thường là admin).

## Ví dụ 2 - Tân công Mù (Time-Based Blind SQLi)

**Tình huống:** Một trang web hiển thị bài viết, URL có dạng: <https://example.com/posts.php?id=1>. Trang này không hiển thị lỗi hay dữ liệu gì lừa, chỉ đơn giản là có bài viết (True) hoặc không có (False).

**Mục tiêu của Hacker:** Tìm hiểu xem mật khẩu của user admin ký tự đầu tiên có phải là 'a' không.

### **Payload của Hacker:**

posts.php?id=1' AND (SELECT 1 FROM users WHERE username='admin' AND password LIKE 'a%') AND (SELECT SLEEP(5))--

### **Phân tích:**

- Kẻ tấn công gửi payload này và đo thời gian phản hồi của máy chủ.
- **Trường hợp 1: Trang web phản hồi sau 5 giây (hoặc lâu hơn).**
  - Ý nghĩa: Điều kiện là TRUE. Mật khẩu của admin có bắt đầu bằng 'a'.
  - Câu lệnh đã thực thi: SLEEP(5) đã được gọi.
- **Trường hợp 2: Trang web phản hồi ngay lập tức.**
  - Ý nghĩa: Điều kiện là FALSE. Mật khẩu không bắt đầu bằng 'a'.

**Quá trình:** Kẻ tấn công dùng tool tự động (như sqlmap) lặp lại quá trình này (like 'b%', 'c%', 'ab%', 'ac%') để dò tìm toàn bộ mật khẩu, ký tự này đến ký tự khác.

### **Nguồn tham khảo:**

1. OWASP (Open Web Application Security Project). (2021). A03:2021-Injection, [https://owasp.org/Top10/A03\\_2021-Injection/](https://owasp.org/Top10/A03_2021-Injection/)
2. PortSwigger. (2024). *What is SQL Injection (SQLi)?*. Web Security Academy, <https://portswigger.net/web-security/sql-injection>
3. MDN Web Docs. (2024). *SQL injection*, [https://developer.mozilla.org/en-US/docs/Glossary/SQL\\_injection](https://developer.mozilla.org/en-US/docs/Glossary/SQL_injection)
4. SANS Institute. *SQL Injection*. Lấy từ: <https://www.sans.org/>

# Phần nội dung nói thêm (Không đưa lên slides)

## Giới thiệu đề tài (Đặt vấn đề)

**Bối cảnh:** Hầu hết các ứng dụng web hiện đại (từ mạng xã hội, ngân hàng đến các trang thương mại điện tử) đều sử dụng Cơ sở dữ liệu (Database) để lưu trữ thông tin (thông tin người dùng, sản phẩm, tin nhắn...).

**Vấn đề:** Điều gì xảy ra nếu kẻ xấu có thể "nói chuyện" trực tiếp với cơ sở dữ liệu đó thông qua các biểu mẫu (form) trên website?

## Giới thiệu SQL Injection:

- SQL Injection (SQLi) là một trong những kỹ thuật tấn công web **lâu đời, phổ biến và nguy hiểm nhất**.
- Trong nhiều năm liền, SQLi luôn đứng trong **Top 10 lỗ hổng bảo mật nghiêm trọng nhất** theo xếp hạng của **OWASP** (Open Web Application Security Project).

## Mục tiêu bài thuyết trình:

- Hiểu rõ SQL Injection là gì.
- Cách thức hoạt động và các dạng tấn công phổ biến.
- (Phần sau): các biện pháp phòng chống hiệu quả và Demo