|  |  |
| --- | --- |
| A blue and yellow logo  Description automatically generated with low confidence | BỘ GIÁO DỤC VÀ ĐÀO TẠO  **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM** |

**ĐỒ ÁN MÔN HỌC**

**Đề tài: SQL Injection**

Ngành: **Công nghệ thông tin**

Chuyên ngành: **Công nghệ phần mềm**

Giảng viên hướng dẫn : Ths. Tống Thanh Văn

Sinh viên thực hiện : Nguyễn Minh Nhựt\_1811061968

: Trần Minh Thuận \_1811061697

: Chiêm Hoàng Long\_1811062174

TP. Hồ Chí Minh, <2021>

**MỤC LỤC**

[CHƯƠNG 1. TỔNG QUAN 4](#_Toc70589430)

[1.1. Mục tiêu chọn đề tài 4](#_Toc70589431)

[1.2. Mục tiêu 4](#_Toc70589432)

[1.3. Cấu trúc đề tài 4](#_Toc70589433)

[CHƯƠNG 2. CƠ SỞ LÍ THUYẾT 6](#_Toc70589434)

[2.1. Tổng quan về ứng dụng Web 6](#_Toc70589435)

[2.1.1. Khái niệm ứng dụng Web 6](#_Toc70589436)

[2.1.2. Các vấn đề liên quan đến ứng dụng Wed 7](#_Toc70589437)

[2.1.2.1. Nguồn gốc phát triển ứng dụng Web 7](#_Toc70589438)

[2.1.2.2. Mô tả hoạt động của các ứng dụng web 8](#_Toc70589439)

[2.1.2.3 Các thành phần của một ứng dụng web 8](#_Toc70589440)

[2.1.3. Một số phương thức tấn công ứng dụng web 9](#_Toc70589441)

[2.1.3.1 Tấn công Bruteforce 9](#_Toc70589442)

[2.1.3.2. Lỗi chứng thực yếu (Insufficient Authentication) 9](#_Toc70589443)

[2.1.3.3 Dự đoán, chèn phiên (Credentical/Session Prediction) 9](#_Toc70589444)

[2.1.3.4. XSS ( Cross-Site Scripting ) 10](#_Toc70589445)

[2.1.3.5. SQL injection 10](#_Toc70589446)

[2.1.3.6. Path Traversal 10](#_Toc70589447)

[2.1.3.7. Từ chối dịch vụ (DoS) 11](#_Toc70589448)

[2.1.4. SQL(Structured Query Language) 11](#_Toc70589449)

[2.4.1. Khái niệm về SQL. 11](#_Toc70589450)

[2.4.2. Các câu lệnh SQL phổ biến. 12](#_Toc70589451)

[2.2. Giới hiệu về SQL injection 12](#_Toc70589452)

[2.2.1 Khái niệm về SQL Injection 12](#_Toc70589453)

[2.2.2 Các mối đe dạo chính từ SQL Injection 14](#_Toc70589454)

[2.2.3 Các phần dễ bị tấn công 16](#_Toc70589455)

[2.2.4 Kỹ thuật né tránh 17](#_Toc70589456)

[2.2.5. Các loại kỹ thuật trốn tránh chữ ký 17](#_Toc70589457)

[2.2.6 Biện pháp đối phó 18](#_Toc70589458)

[2.2.6.1 Phòng thủ trong ứng dụng 21](#_Toc70589459)

[2.2.7 Hậu quả của SQL Injection. 24](#_Toc70589460)

[CHƯƠNG 3. KẾT QUẢ THỰC NGHIỆM 25](#_Toc70589461)

# CHƯƠNG 1. TỔNG QUAN

## Mục tiêu chọn đề tài

Sự phát triển vượt bậc của công nghệ web đã đem lại rất nhiều thuận lợi cho người sử dụng cũng như các nhà phát triển. Nhưng cùng với sự phát triển này thì các ứng dụng web cũng trở thành mục tiêu ưu thích của những kể tấn công. Các hình thức tấn công rất đa dạng như thay đổi nội dung của trang web, tấn công từ chối dịch vụ khiến cho việc truy cập trang web không thể thực hiện hoặc rất khó thực hiện, chiếm quyền điều khiển trang web… Mục tiêu của các hacker cũng rất khác nhau, có thể tấn công xuất phát từ thiện chí, nhằm tìm ra những điểm yếu và thông báo cho nhà quản trị hệ thống. Nghiêm trọng hơn là tấn công để phục vụ cho các mục đích xấu như tống tiền trang web, lấy cắp những dữ liệu nhạy cảm như thông tin về thể tín dụng, mua hàng thông qua tài khoản của người khác… Trong các hình thức tấn công thì tấn công bằng cách chèn mã lệnh (injection) là phổ biến. Tấn công website bằng kỹ thuật SQL injection từ lâu đã là mối quan tâm bảo mật hàng đầu của các nhà phát triển web và chủ sở hữu website. Giờ đây, các cuộc tấn công này ngày càng trở nên khó phát hiện và ngăn chặn hơn.Số lượng các vụ tấn công nhằm vào cơ sở dữ liệu (CSDL) web đã lên tới một con số kỹ lục.

## Mục tiêu

Giúp chúng ta có thể hiểu hơn về các ứng dụng website, các mối đe dọa về vấn đề an toàn thông tin khi chúng ta làm việc trên ứng dụng web hàng ngày, hiểu rõ hơn về các kỹ thuật tấn công và bảo mật web. Xác định được nguyên nhân, nhận diện chính xác đối tượng động cơ, cách thức của kể tấn công xâm nhập vào cơ sở dữ liệu. Xác định mục tiêu, mối nguy hiểm thường trực về an ninh ứng dụng web của các tổ chức. Hiểu rõ khái niệm sql injection và phương thức hoạt động của các hacker thông qua lỗ hổng này. Biết sử dụng phương pháp và các công cụ cơ bản để kiểm tra an ninh bảo mật trên ứng dụng web nhằm có cách phòng chống hiệu quả.

## Cấu trúc đề tài

CHƯƠNG 1: TÔNG QUAN

-Tổng quan về vấn đề được nghiên cứu tóm tắt những lý thuyết, yêu cầu của đề tài

-Nhiệm vụ đồ án: lý do hình thành đề tài, mục tiêu nghiên cứu, đối tượng, phạm vi giới hạn

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

* Cơ sở lý thuyết, các lỗ hổng của sql, cách tấn công và phòng chống,…

CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM

- Kịch bản thực nghiệm

- Yêu cầu thực nghiệm

- Công cụ sử dụng

- Các bước thực hiện

- Kết luận

# CHƯƠNG 2. CƠ SỞ LÍ THUYẾT

## 2.1. Tổng quan về ứng dụng Web

### 2.1.1. Khái niệm ứng dụng Web

Ứng đụng Web là một ứng dụng chủ, khách sử dụng giao thức HTTP để tương tác với người dùng hay hệ thống khác.

Trình khách dành cho người sử dụng thường là một trình duyệt Web như Internet Explorer hay Google Chrome.Cũng có thể là một chương trình đóng vai trò đại lý người dùng hoạt động như một trình duyệt tự động. Người đùng gửi và nhận các thông tin từ trình chủ thông qua việc tác động vào các trang Web.Các chương trình có thế là các trang trao đối mua bán,các diễn đàn,gửi nhận e-mai,…

Tốc độ phát triển của các kỹ thuật xây dựng ứng dụng Wb cũng phát triển rất nhanh. Trước đây những ứng dụng Web thường được xây đựng bằng CGI (Common Gatway Interface) được chạy trên các trình chủ Web và có thể kết nối vào các cơ sở đữ liệu đơn giản trên cùng máy chủ Ngày nay,ứng dụng Web thường được viết bằng Java và chạy trên máy chủ phân tán, kết nói đến nhiều nguồn dữ liệu.

Dù có nhiều biến thế, một ứng dụng Web thông thường được cấu trúc như một ứng dụng 3 lớp :

* Lớp trình bày: Lớp này có nhiệm vụ hiến thị dữ liệu cho người dùng, ngoài ra còn có thế thêm các ứng dụng tạo bố cục cho trang Web.
* Lớp ứng dụng: Là nơi xử lý các ứng đụng Web. Nó sẽ xử lý thông tin người đùng yêu cầu,đưa ra quyết định,gửi kết quả đến “ Lớp trình bày “ lớp này thường được cài đặt bằng các kỹ thuật lập trình như CGI,Java,NET hay ColdFusion, được triển khai trên các trình chủ như IBM., WebSphere, WebLogic, Apache,…
* Lớp dữ liệu: Thường là các hệ quản trị cơ sở dữ liệu (DBMS) chịu trách nhiệm quản lý các file dữ liệu và quyền sử dụng.

Kiến trúc một ứng dụng Web,Trong đó:

* + Trình duyệt: Internet Explorer,Google Chrome.
  + Trình chủ: Apache,...
  + Hệ quản trị cơ sở dữ liệu: SQL,Server.MySQLL,...

**SQL injection** là một kỹ thuật cho phép những kẻ tấn công lợi dụng lỗ hổng của việc kiểm tra dữ liệu đầu vào trong các ứng dụng web và các thông báo lỗi của hệ quản trị cơ sở dữ liệu trả về để inject (tiêm vào) và thi hành các câu lệnh SQL bất hợp pháp. SQL injection có thể cho phép những kẻ tấn công thực hiện các thao tác, delete, insert, update, v.v. 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](https://vi.wikipedia.org/w/index.php?title=V%E1%BA%ADt_trung_gian_(m%C3%A3_%C4%91%E1%BB%99c)&action=edit&redlink=1) tấn công trên các ứng dụng web có dữ liệu được quản lý bằng các hệ quản trị cơ sở dữ liệu như [SQLServer](https://vi.wikipedia.org/w/index.php?title=SQL_Server&action=edit&redlink=1), [MySQL](https://vi.wikipedia.org/wiki/MySQL), [Oracle](https://vi.wikipedia.org/wiki/Oracle), [DB2](https://vi.wikipedia.org/wiki/DB2), [Sysbase](https://vi.wikipedia.org/w/index.php?title=Sysbase&action=edit&redlink=1)...



## 2.1.2. Các vấn đề liên quan đến ứng dụng Wed

### 2.1.2.1. Nguồn gốc phát triển ứng dụng Web

Các ứng dụng Web được phát triển từ nhiều nguồn khác nhau, nên các lỗ hồng, các lỗi bảo mật cũng rất đa dạng. Tuy vậy, các ứng dụng Web chủ yếu được phát triển từ các nguồn chính sau đây:

* Người lập trình tự phát triển ứng dụng Web (sử dụng các ngôn ngữ kịch bản đế tạo ứng dụng, phát triển rộng rãi mà ít quan tâm đến quá trình phát triển ứng dụng an toàn. Thiếu đội ngũ lập tình với kỹ năng nhận biết phát triến ứng dụng tránh các lỗi bảo mật).
* Sử dụng ứng dụng Web từ mã nguồn mở (Thường không theo đôi và cập nhật các bản và lỗi bảo mật).

Phát triển ứng dụng Web từ một ứng dụng mở khác (Trường hợp này thường không kiểm tra lỗi bảo mật ứng dụng cũ trước khí phát triển tiếp,nên vẫn tồn tại các lỗi bảo mật).

### 2.1.2.2. Mô tả hoạt động của các ứng dụng web

Đầu tiên trình duyệt sẽ gửi một yêu cầu (requesr) đến trình chủ Wb thông qua các lệnh cơ bản GET, POST,... của giao thức HTTP, trình chủ lúc này có thế cho thực thi một chương trình được xây dựng từ nhiều ngôn ngữ như: C,C++,Java,... hoặc trình chủ yêu cầu bộ diễn dịch thực thí các trang ASP, JSE,... theo yêu cầu của trình khách.

Tùy theo các tác vụ của chương trình được cài đặt mà nó xử lý tính toán kết nối đến cơ sở dữ liệu, lưu các thông tin do trình khách gửi đến... và từ đó trả về cho trình khách 1 luồng dữ liệu cố định dạng theo giao thức HTTP nó gồm 2 phần:

* + 1. Header mô tả các thông tin về gói dữ liệu và thuộc tính, trạng thất trao đổi giữa trình duyệt và Website.
    2. Body là phần nội dung dữ liệu mà Server gửi về Client,nó có thể là một file HTML, một hình ảnh, một đoạn phim hay một văn bản bất kỹ.

Theo mô hình ở hình với Firewal,luồng thông tín giữa trình chủ và trình khách là luồng thông tin hợp lệ.Vì thế nếu hacker tìm thấy vài lỗ hồng trong ứng dụng Web thì Firetall không còn hữu dụng trong việc ngăn chặn hacker này. Do đó, các kỹ thuật tấn công vào một hệ thống mạng ngày nay đang dần tập trung vào những sơ suất (hay lỗ hồng) trong quá trình tạo ứng dụng của những nhà phát triển Web hơn là tấn công trực tiếp vào hệ thống mạng, hệ điều hành. Tuy nhiên, hacker cũng có thế lợi dụng các lỗ hổng Web để mở rộng sự tấn công của mình vào các hệ thống không liên quan khác.

### 2.1.2.3 Các thành phần của một ứng dụng web

* 1. Thành phần server được viết bằng một trong số các ngôn ngữ lập trình hỗ trợ web như PHP, C#, Java, Python, JavaScript (Node.js). Thành phần server tự nó cũng không phải là một chương trình độc lập mà là thành phần mở rộng của một trong số các chương trình máy chủ web. Lựa chọn server và ngôn ngữ lập trình thường đi thành combo.
  2. Thành phần client của ứng dụng web là loại “chương trình” đặc biệt chạy bên trong trình duyệt. Thành phần client được tạo ra từ 3 thành phần riêng biệt: nội dung (HTML), hình thức (CSS), tương tác (JavaScript).
  3. Một thành phần không thể thiếu của ứng dụng web là cơ sở dữ liệu. Hầu như ứng dụng “đàng hoàng” nào cũng cần đến một cơ sở dữ liệu thực sự như MySQL, Oracle hay SQL Server. Ngôn ngữ sử dụng chính cho cơ sở dữ liệu là (các biến thể) SQL.

## 2.1.3. Một số phương thức tấn công ứng dụng web

### 2.1.3.1 ****Tấn công Bruteforce****

Bruteforce là cách thức thử tất cả các khả năng có thể có để đoán các thông tin cá nhân đăng nhập: tài khoản, mật khẩu, số thẻ tín dụng…Nhiều hệ thống cho phép sử dụng mật khẩu hoặc thuật toán mã hóa yếu sẽ tạo điều kiện cho tin tặc sử dụng phương pháp tấn công này để đoán tài khoản và mật khẩu đăng nhập. Sau đó sử dụng các thông tin này để đăng nhập truy cập vào tài nguyên hệ thống. Biện pháp đối phó:

* Tăng cường độ mạnh cho mật khẩu (Độ dài ít nhất 6 ký tự, không chứa chuỗi username, chứa ít nhất 1 ký tự số, chứa ít nhất 1 ký tự đặc biệt, không cho phép thay đổi mật khẩu trùng lặp đã sử dụng, quản lý, điều khiển thông báo lỗi)
* Sử dụng cơ chế chứng thực (Basic hoặc Digest Authentication)
* Hạn chế số lần đăng nhập hoặc khóa tài khoản đăng nhập sai
* Sử dụng module Mod\_Dosevasive để xác định dấu hiệu của kiểu tấn công này.

### 2.1.3.2. ****Lỗi chứng thực yếu (Insufficient Authentication)****

Lỗi chứng thực yếu xuất hiện khi một website cho phép truy cập các nội dung, tài nguyên nhạy cảm mà không có đủ quyền. Các trang quản trị là một ví dụ dễ thấy nhất. Nếu không có cơ chế phân quyền hợp lý thư mục cũng như tài khoản đăng nhập trang quản trị này. Tin tặc hoàn toàn có khả năng vượt qua được cơ chế đăng nhập để chiếm quyền điều khiển trang này.

### 2.1.3.3 ****Dự đoán, chèn phiên (Credentical/Session Prediction)****

Dự đoán, chèn phiênlà một phương thức chiếm phiên (hijacking). Thông thường, khi một tài khoản thực hiện quá trình chứng thực đối với server (tài khoản/mật khẩu). Dựa vào các thông tin này, server sẽ tạo một giá trị session ID duy nhất để cho phép và duy trì kết nối. Nếu đoán được session ID kế tiếp thì tin tặc có khả năng chiếm phiên đăng nhập của người dùng hợp lệ khác. Biện pháp đối phó:

* Sử dụng SSL (*mod\_ssl*) trong quá trình chứng thực để chống lại việc nghe lén dữ liệu quan trọng.
* Sử dụng cơ chế tạo session ID ngẫu nhiên, thuật toán mã hóa mạnh.
* Session ID phải đủ lớn để làm khó quá trình tấn công brute-fore.
* Giới hạn thời gian tồn tại của session ID.

### 2.1.3.4. ****XSS ( Cross-Site Scripting )****

XSS là một trong những kĩ thuật tấn công phổ biến nhất hiện nay, đồng thời nó cũng là một trong những vấn đề bảo mật quan trọng đối với các nhà phát triển web và cả những người sử dụng web. Bất kì một website nào cho phép người sử dụng đăng thông tin mà không có sự kiểm tra chặt chẽ các đoạn mã nguy hiểm thì đều có thể tiềm ẩn các lỗi XSS. Tin tặc tấn công bằng cách chèn vào các website động (ASP, PHP, CGI, JSP …) những thẻ HTML hay những đoạn mã script nguy hiểm có thể gây nguy hại cho những người sử dụng khác. Trong đó, những đoạn mã nguy hiểm đựơc chèn vào hầu hết được viết bằng các Client-Site Script như JavaScript, JScript, DHTML và cũng có thể là cả các thẻ HTML.

### 2.1.3.5. SQL injection

Tấn công SQL Injection được thực thi bằng cách chèn các câu truy vấn SQL vào dữ liệu tương tác giữa máy khách và trình ứng dụng. Quá trình khai thác lỗi SQL Injection thành công có thể giúp tin tặc lấy được các dữ liệu nhạy cảm trong cở sở dữ liệu, thay đổi cơ sở dữ liệu (Insert/Update/Delete), thực thi các hành động với quyền của người quản trị và cao hơn có thể điều khiển được hệ điều hành máy chủ.

### 2.1.3.6. ****Path Traversal****

Path Traversal hay còn được biết với một số tên khác như “dot-dot-slash”, “directory traversal”,”directory clumbing” và “backtracking” là hình thức tấn công truy cập đến những file và thư mục mà được lưu bên ngoài thư mục webroot. Hình thức tấn công này không cần sử dụng một công cụ nào mà chỉ đơn thuần thao tác các biến với ../ (dot-dot-slash) để truy cập đến file, thư mục, bao gồm cả source code, những file hệ thống, …

### 2.1.3.7. Từ chối dịch vụ (DoS)

DoS là kỹ thuật tấn công nhằm không cho phép các truy cập hợp lệ truy cập tới server. Kỹ thuật tấn công này thường xảy ra tại lớp mạngvà lớp ứng dụng. Các hệ thống đích có thể bị tấn công DoS:

* Người dùng riêng lẻ: quá trình đăng nhập lặp đi lặp lại với tài khoản hợp lệ nhưng mật khẩu không đúng. Sau nhiều lần đăng nhập sai, hệ thống sẽ khóa tài khoản hợp lệ này, dẫn đến người dùng hợp lệ sẽ không thể đăng nhập được.
* Máy chủ cơ sở dữ liệu: Sử dụng kỹ thuật chèn câu lệnh truy vấn SQL chỉnh sửa cơ sở dữ liệu, vì thế hệ thống sẽ không thể phục vụ các truy cập từ client.

Máy chủ phục vụ web: Sử dụng kỹ thuật tấn công tràn bộ đệm ( Buffer Overflow) để gới các gói truy vấn và làm đổ vỡ các tiến trình tại phía máy chủ phục vụ web, dẫn đến hệ thống máy chủ webserver sẽ không có khả năng phục vụ các truy cập hợp lý.

## 2.1.4. SQL(Structured Query Language)

### 2.4.1. Khái niệm về SQL.

SQL là loại ngôn ngữ máy tính, giúp cho thao tác lưu trữ và truy xuất dữ liệu được lưu trữ trong một cơ sở dữ liệu quan hệ. SQL là viết tắt của Structured Query Language là ngôn ngữ truy vấn có cấu trúc. Một trong những lý do khiến cho SQL được sử dụng phổ biến, chính là nó đã cho phép người dùng thực hiện đa dạng các chức năng sau:

* Cho phép người dùng truy cập dữ liệu trong các hệ thống quản lý cơ sở dữ liệu quan hệ.
* Cho phép người dùng mô tả dữ liệu.
* Cho phép người dùng xác định dữ liệu trong cơ sở dữ liệu và thao tác dữ liệu đó.
* Cho phép nhúng trong các ngôn ngữ khác sử dụng mô-đun SQL, thư viện và trình biên dịch trước.
* Cho phép người dùng tạo và thả các cơ sở dữ liệu và bảng.
* Cho phép người dùng tạo chế độ view, thủ tục lưu trữ, chức năng trong cơ sở dữ liệu.
* Cho phép người dùng thiết lập quyền trên các bảng, thủ tục và view.

### 2.4.2. Các câu lệnh SQL phổ biến.

Trong SQL có khá nhiều câu lệnh, mệnh đề khác nhau. Tuy nhiên dưới đây là một số những truy vấn thường gặp nhất các câu lệnh này có thể các hacker dùng xâm nhập và ứng dụng web cá nhân của mọi người và hacker có thể thanh đổi thông tin và có thể lấy được các thông tin quan trọng và sử dụng nó cho các mục đích cá nhân của họ.  Chúng ta sẽ cùng tìm hiểu chi tiết ngay bên dưới:

* Truy vấn Update
* Truy vấn Insert
* Truy vấn Select
* Mệnh đề Order By
* Mệnh đề Group By
* Từ khóa Distinct
* Mệnh đề Where
* Mệnh đề AND & OR

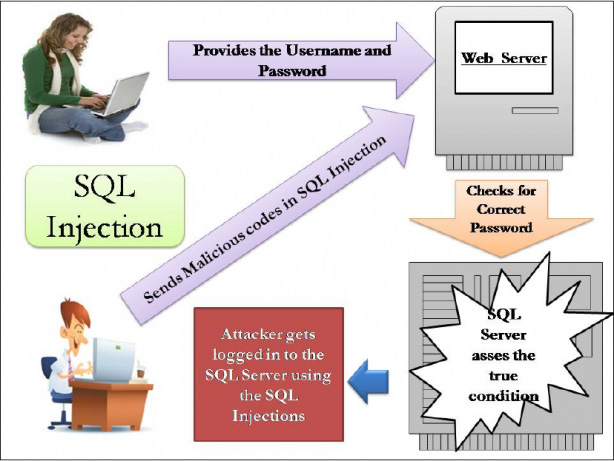
## 2.2. Giới hiệu về SQL injection

### 2.2.1 Khái niệm về SQL Injection

**SQL Injection** là một loại lỗ hỗng ứng dụng web mà kẻ tấn công có thể thao tác và thực hiện một lệnh truy vấn SQL để lấy các thông tin từ cơ sở đữ liệu. Đây là loại tấn công chủ yếu khi một ứng dụng web cho phép người dùng sử dụng truy cập và sử dụng dữ liệu mà không xét quyên truy cập hay mã hóa dữ liệu đó. Lỗ hổng này có thể dẫn đến việc lộ các thông tin nhạy cảm, số thẻ tín dụng, hoặc các đỡ liệu tài chính khác cho phêp kẻ tấn công có thê thêm, xóa, sửa, cập nhật, thay đôi các dữ liệu được lưu trong cơ sở dữ liệu. Đây là một lỗ hỗng ứng dụng web, không phải là một lỗi về cơ sở dữ liệu hay vấn đẻ vẻ máy chủ. Hầu hết các lập trình viên đều không nhận thức được mối đe đọa này.

Một vài lý do đã tạo nên tên tuổi lừng lẫy của **SQL Injection:**

* Bỏ qua xác thực(***Authentication Bypass***): Sử dụng cuộc tấn công này, kẻ tấn công đăng nhập vào một ứng dụng mà không cần cung cấp tên người dùng và mật khẩu hợp lệ, đồng thời có được các đặc quyền quản trị.
* Bỏ qua ủy quyền(***Authorization Bypass***): Sử dụng cuộc tấn công này, kẻ tấn công thay đổi thông tin ủy quyền được lưu trữ trong cơ sở dữ liệu bằng cách khai thác lỗ hổng SQL injection. Tiết lộ Thông tin: Sử dụng cuộc tấn công này, kẻ tấn công lấy được thông tin nhạy cảm được lưu trữ trong cơ sở dữ liệu.
* Tính toàn vẹn dữ liệu được thỏa hiệp(***Compromised Data Integrity***): Sử dụng cuộc tấn công này, kẻ tấn công phá hoại một trang web, chèn nội dung độc hại vào các trang web hoặc thay đổi nội dung của cơ sở dữ liệu.
* Tính sẵn có của dữ liệu bị xâm phạm(***Compromised Availability of Data***): Sử dụng cuộc tấn công này, kẻ tấn công sẽ xóa thông tin cơ sở dữ liệu, xóa nhật ký hoặc kiểm tra thông tin được lưu trữ trong cơ sở dữ liệu.
* Thực thi mã từ xa(***Remote Code Execution***): Sử dụng cuộc tấn công này, kẻ tấn công xâm phạm hệ điều hành máy chủ.



*Hình 2. Cơ bản về cơ chế SQL injection*

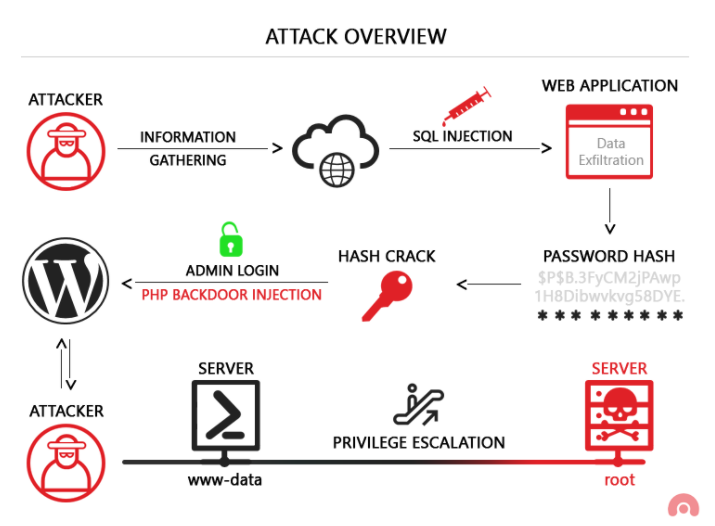
### 2.2.2 Các mối đe dạo chính từ SQL Injection

Các mối đe dọa chính của SQL Injection

* **Snoofing iđentity (Mạo danh):** Những kẻ tấn công sẽ mạo danh một email hoặc một trang web của một tổ chức để đánh lừa người dùng.
* **Changing prices (Thay đổi giá):** Một trong số những vấn để của SQL Injection là nó có thể thay đôi đữ liệu. Ở đây, những kẻ tấn công sẽ thay đổi giá cả một trang mua sắm trực tuyển để có thể mua sản phẩm đó với giá rẻ hơn.
* **Tamper with đatabase records (Xáo trộn các hỗ sơ cơ sỡ dữ liệu):** Cơ sở dữ liệu chính sẽ hoàn toàn bị hư hại; thậm chí còn có khả năng đã bị thay thể hoàn toàn hoặc thậm chí bị xóa hết đữ liệu.
* **Escalation of privileges (Leo thang đặc quyền):** Một khi hệ thống bị tấn công, kê tấn công sẽ tìm kiểm đặc quyên truy cập cao nhất của các thành viên quân trị để có thể chiếm quyền truy cập vào hệ thống cũng như vào mnạng nội bộ. -
* **Denial-of-service on the server (Từ chối các dịch vụ từ máy chữ):** Từ chối địch vụ từ máy chủ là một dạng tấn công mà người dùng không thể truy cập vào hệ thống. Ngày càng nhiêu yêu cầu được gửi đến rnây chủ mà không thể sử lý nó. Điều này dẫn đến tạm ngưng dịch vụ của máy chủ.
* **Complete đisclosure of all the đata on the system (Lộ tất cã thông tin dữ liệu của hệ thống):** Một khi một hệ thống bị tấn công các dữ liệu quan trọng và bí mật như: số thẻ tín sơ và tài chính,...v.v sẽ bị tiết lộ.
* **Destruction of data (Phá hủy dữ liệu)** : Những kẻ tấn công sau khi chiếm hoàn toàn quyền hệ thống sẽ phá hủy hoàn toàn dữ liệu, kết quả làm tôn thất rất lớn cho công ty.
* **Voiding system's critical transaction (Tránh các giao địch quan trọng cũa hệ thống) :** Những kẻ tấn công có thẻ vận hành hệ thống tránh tất cả các giao dịch.

#### Modifying the records (Thay đổi hỗ sơ): Kẻ tắn công có thê sửa đổi các đữ liệu t rong cơ sở đữ liệu của công ty, gây ra trở ngại lớn cho hệ thống quản lý cơ sở dữ liệu công ty.

### 2.2.3 Các phần dễ bị tấn công



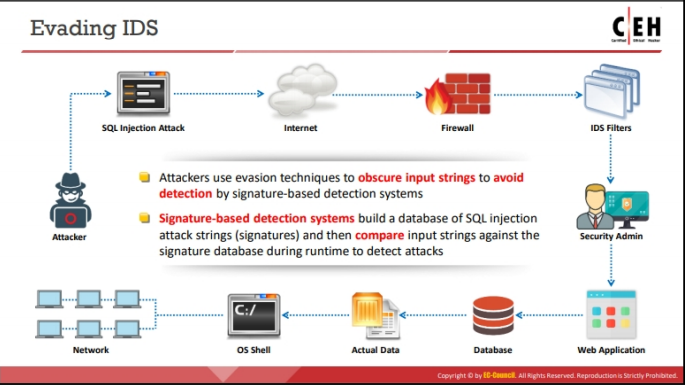
Các phần dễ bị tấn công bao gồm:

* Form đăng nhập
* Form tìm kiếm
* Form nhận xét
* Bất kì trường lưu hoặc trường đầu vào của dữ liệu
* Liên kết của website

Cần lưu ý là trong khi thử nghiệm chống lại tấn công này là không thể chỉ kiểm tra một hoặc một vài trường bởi vì một trường có thể được bảo vệ chống lại SQL Injection, nhưng một trường khác thì không. Do đó, điều quan trọng là đừng quên kiểm tra tất cả các trường của trang web

### 2.2.4 Kỹ thuật né tránh

Tường lửa và hệ thống phát hiện xâm nhập (IDS) có thể phát hiện các nỗ lực chèn SQL dựa trên các chữ ký được xác định trước. Ngay cả khi các mạng bao gồm các chu vi an ninh mạng này, những kẻ tấn công sử dụng các kỹ thuật trốn tránh để thực hiện việc chèn SQL mà không bị phát hiện. Các kỹ thuật trốn tránh như vậy bao gồm mã hóa hex, thao tác khoảng trắng, nhận xét trong dòng, kết hợp phức tạp, mã hóa ký tự, v.v. Phần này sẽ thảo luận chi tiết về các kỹ thuật này.



### 2.2.5. Các loại kỹ thuật trốn tránh chữ ký

Các loại kỹ thuật trốn tránh chữ ký khác nhau được liệt kê dưới đây:

• Chú thích trong dòng(*In-line comment*): Che khuất các chuỗi đầu vào bằng cách chèn chú thích trong dòng giữa các từ khóa SQL.

• Mã hóa Char(*Char encoding*): Sử dụng chức năng CHAR tích hợp để đại diện cho một ký tự.

• Nối chuỗi(*String Concatenation*): Nối văn bản để tạo một từ khóa SQL bằng cách sử dụng các hướng dẫn dành riêng cho DB.

• Mã bị xáo trộn(Obfuscated code): Mã bị xáo trộn là một câu lệnh SQL khó hiểu.

• Thao tác Khoảng trắng(*Manipulating White Spaces*): Che khuất các chuỗi nhập bằng cách chèn một khoảng trắng giữa các từ khóa SQL.

• Mã hóa Hex(*Hex encoding*): Sử dụng mã hóa thập lục phân để biểu diễn một chuỗi truy vấn SQL.

• Đối sánh phức tạp(*Sophisticed Matches*): Sử dụng biểu thức thay thế của "OR 1 = 1".

• Mã hóa URL(*URL encoding*): Che khuất một chuỗi đầu vào bằng cách thêm dấu phần trăm (%) trước mỗi điểm mã.

• Null Byte: Sử dụng ký tự byte rỗng (% 00) trước một chuỗi để bỏ qua cơ chế phát hiện.

• Biến thể(*Case Variation*): Làm xáo trộn câu lệnh SQL bằng cách trộn nó với các chữ cái viết hoa và viết thường.

• Khai báo các biến(*Declare variables*): Sử dụng các biến để chuyển một loạt các câu lệnh SQL được tạo thủ công đặc biệt và bỏ qua cơ chế phát hiện.

• Phân mảnh IP(*IP Fragmentation*): Sử dụng các phân mảnh gói để che khuất tải trọng tấn công, không bị phát hiện bởi cơ chế chữ ký.

• Các biến thể(*Varuation*): Sử dụng câu lệnh WHERE luôn được đánh giá là "đúng", để có thể sử dụng bất kỳ phép so sánh toán học hoặc chuỗi nào.

### 2.2.6 Biện pháp đối phó

Các phần trước đã thảo luận về mức độ nghiêm trọng của các cuộc tấn công SQL injection, các kỹ thuật khác nhau của chúng, các công cụ được sử dụng để thực hiện chèn sQl, các kỹ thuật được sử dụng để vượt qua chữ ký IDS / tường lửa, v.v. Các cuộc thảo luận này là về các kỹ thuật tấn công mà kẻ tấn công có thể áp dụng cho các cuộc tấn công SQL injection. Phần này thảo luận về các kỹ thuật phòng thủ chống lại các cuộc tấn công SQL injection và trình bày các biện pháp đối phó để bảo vệ các ứng dụng web.

Để chống lại SQL injection, nhà phát triển cần quan tâm đúng mức đến việc định cấu hình và phát triển một ứng dụng để tạo ra một ứng dụng mạnh mẽ và an toàn. Nhà phát triển nên sử dụng các phương pháp và biện pháp đối phó tốt nhất để ngăn các ứng dụng dễ bị tấn công SQL injection.

Một số biện pháp đối phó để bảo vệ chống lại các cuộc tấn công SQL injection được liệt kê dưới đây:

• Không đưa ra giả định về kích thước, kiểu hoặc nội dung của dữ liệu mà ứng dụng của bạn nhận được Kiểm tra kích thước và kiểu dữ liệu của đầu vào và thực thi các giới hạn thích hợp để ngăn vượt quá bộ đệm

• Kiểm tra nội dung của các biến chuỗi và chỉ chấp nhận các giá trị mong đợi

• Từ chối các mục nhập chứa dữ liệu nhị phân, chuỗi thoát và ký tự nhận xét

• Không bao giờ xây dựng các câu lệnh Transact-SQL trực tiếp từ đầu vào của người dùng và sử dụng các thủ tục được lưu trữ để xác thực thông tin nhập của người dùng

• Triển khai nhiều lớp xác thực và không bao giờ ghép đầu vào của người dùng chưa được xác thực

• Tránh xây dựng SQL động với các giá trị đầu vào được nối

• Đảm bảo rằng các tệp cấu hình web cho mỗi ứng dụng không chứa thông tin nhạy cảm

• Sử dụng các loại tài khoản SQL hạn chế nhất cho các ứng dụng

• Sử dụng hệ thống phát hiện xâm nhập mạng, máy chủ và ứng dụng để giám sát các cuộc tấn công xâm nhập

• Thực hiện kiểm tra chèn hộp đen tự động, phân tích mã nguồn tĩnh và kiểm tra thâm nhập thủ công để thăm dò lỗ hổng

• Giữ dữ liệu không đáng tin cậy tách biệt với các lệnh và truy vấn

• Trong trường hợp không có API được tham số hóa, hãy sử dụng cú pháp thoát cụ thể cho trình thông dịch để loại bỏ các ký tự đặc biệt

• Sử dụng thuật toán băm an toàn như SHA256 để lưu trữ mật khẩu người dùng thay vì văn bản rõ

• Sử dụng lớp trừu tượng quyền truy cập dữ liệu để thực thi quyền truy cập dữ liệu an toàn trên toàn bộ ứng dụng

• Đảm bảo rằng các thông báo dò tìm mã và gỡ lỗi được xóa trước khi triển khai ứng dụng

• Thiết kế mã sao cho mã này bẫy và xử lý các ngoại lệ một cách thích hợp

• Áp dụng các quy tắc đặc quyền ít nhất để chạy các ứng dụng truy cập vào DBMS

• Xác thực dữ liệu do người dùng cung cấp cũng như dữ liệu thu được từ các nguồn không đáng tin cậy ở phía máy chủ Tránh các số nhận dạng được trích dẫn / phân tách vì chúng làm phức tạp đáng kể tất cả các nỗ lực đưa vào danh sách trắng, danh sách đen và thoát

• Sử dụng một câu lệnh đã soạn sẵn để tạo một truy vấn được tham số hóa để chặn việc thực thi truy vấn

• Đảm bảo rằng tất cả các đầu vào của người dùng đã được khử trùng trước khi sử dụng chúng trong các câu lệnh SQL động

• Sử dụng các biểu thức chính quy và các thủ tục được lưu trữ để phát hiện mã có thể gây hại

• Tránh sử dụng bất kỳ ứng dụng web nào không được máy chủ web kiểm tra

• Cô lập máy chủ web bằng cách khóa nó trong các miền khác nhau

• Đảm bảo tất cả các bản vá lỗi phần mềm được cập nhật thường xuyên Thường xuyên theo dõi các câu lệnh SQL từ các ứng dụng được kết nối với cơ sở dữ liệu để xác định các câu lệnh SQL độc hại

• Việc sử dụng các khung nhìn là cần thiết để bảo vệ dữ liệu trong các bảng cơ sở bằng cách hạn chế quyền truy cập và thực hiện các phép biến đổi

• Vô hiệu hóa quyền truy cập shell vào cơ sở dữ liệu

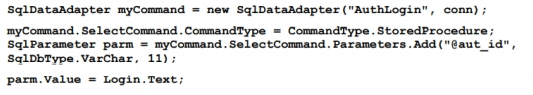
• Không tiết lộ thông tin lỗi cơ sở dữ liệu cho người dùng cuối

• Sử dụng một API an toàn cung cấp giao diện được tham số hóa hoặc tránh hoàn toàn việc sử dụng trình thông dịch

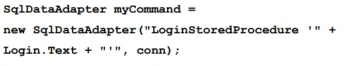
• Thuê ngoài quy trình xác thực của các ứng dụng, chẳng hạn như sử dụng OAUTH APIS, cho phép người dùng đăng nhập bằng tài khoản người dùng hiện có của họ và đảm bảo hơn nữa rằng chi tiết đăng nhập của họ được lưu trữ ở một vị trí

***\* Sử dụng các SQL Parameters an toàn***

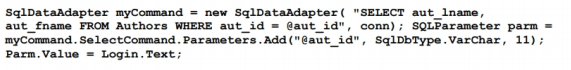
Thực thi kiểm tra kiểu và độ dài bằng cách sử dụng Parameters để đầu vào được coi là giá trị chữ thay vì mã thực thi



***\* Ví dụ về mã dễ bị tấn công***



***\* Ví dụ về mã bảo vệ***



Để bảo vệ chống lại các cuộc tấn công SQL injection, một hệ thống phải tuân theo các biện pháp đối phó được mô tả trong phần trước và sử dụng các tham số SQL an toàn về kiểu. Để bảo vệ máy chủ web, hãy sử dụng WAF / IDS và lọc các gói. Thường xuyên cập nhật phần mềm bằng cách sử dụng các bản vá để giữ cho máy chủ được cập nhật nhằm bảo vệ máy chủ khỏi những kẻ tấn công. Vệ sinh và lọc đầu vào của người dùng, phân tích mã nguồn cho SQL injection và giảm thiểu việc sử dụng các ứng dụng của bên thứ ba để bảo vệ các ứng dụng web. Sử dụng các thủ tục được lưu trữ và truy vấn tham số để truy xuất dữ liệu, vô hiệu hóa các thông báo lỗi dài dòng có thể hướng dẫn kẻ tấn công thông tin hữu ích và sử dụng các trang lỗi tùy chỉnh để bảo vệ các ứng dụng web. Để tránh việc đưa SQL vào cơ sở dữ liệu, hãy kết nối các tài khoản không có đặc quyền và cấp các đặc quyền ít nhất có thể cho cơ sở dữ liệu, bảng và cột. Tắt các lệnh như xp\_cmdshell, có thể ảnh hưởng đến HĐH của hệ thống.

### 2.2.6.1 Phòng thủ trong ứng dụng

* **Xác thực đầu vào**

Có một số cách mà thông qua đó đầu vào được cung cấp cho ứng dụng được làm sạch trước khi được cơ sở dữ liệu xử lý. Cách tiếp cận chính là xác nhận đầu vào do người dùng cung cấp bằng cách sử dụng các kỹ thuật như danh sách trắng và danh sách đen. Xác thực đầu vào giúp các nhà phát triển ngăn không cho dữ liệu do người dùng cung cấp ảnh hưởng đến logic của mã.

* **Xác thực danh sách trắng**

Xác thực danh sách trắng là phương pháp hay nhất, theo đó chỉ danh sách các thực thể (tức là kiểu dữ liệu, phạm vi, kích thước, giá trị, v.v.) đã được phê duyệt để truy cập bảo mật mới được chấp nhận.

Xác thực danh sách trắng cũng có thể được gọi là xác thực tích cực hoặc bao gồm. Xác thực này thường được triển khai bằng cách sử dụng biểu thức chính quy. Ví dụ: các ký tự được sử dụng để xác thực danh sách trắng bao gồm "A {) 0e? $". Việc thực hiện xác thực danh sách trắng có thể phức tạp trong một số trường hợp không thể dễ dàng xác định được đầu vào hoặc nếu đầu vào có bộ ký tự lớn.

* **Xác thực danh sách đen**

Xác thực danh sách đen từ chối tất cả các đầu vào độc hại đã bị từ chối để truy cập được bảo vệ. Việc xác nhận danh sách đen có thể là một thách thức vì mọi nội dung và đặc điểm của cuộc tấn công cũng phải được giải thích, hiểu và dự đoán cho các cuộc tấn công trong tương lai. Xác thực danh sách đen cũng có thể được gọi là xác nhận hoặc loại trừ phủ định.

Việc xác thực này thường được triển khai bằng cách sử dụng các biểu thức chính quy có chứa danh sách các ký tự hoặc chuỗi cần bị cấm. Ví dụ: các ký tự được sử dụng để xác thực danh sách đen bao gồm "181--1: 1 / \ • \\\ • \_I \ EI @ l xp\_".

Nói chung, danh sách đen không được thực hiện một cách cô lập; nó được thực hiện cùng với danh sách trắng. Phương pháp tốt nhất để ngăn chặn các cuộc tấn công SQL injection là sử dụng danh sách đen cùng với kỹ thuật mã hóa đầu ra để đầu vào có thể được mã hóa và kiểm tra trước khi chuyển đến cơ sở dữ liệu.

* **Mã hóa đầu ra**

Mã hóa đầu ra là một kỹ thuật xác nhận có thể được sử dụng sau khi xác nhận đầu vào. Kỹ thuật này được sử dụng để mã hóa đầu vào để đảm bảo rằng nó được làm sạch đúng cách trước khi chuyển đến cơ sở dữ liệu. Trong một số trường hợp, khi SQL động được sử dụng, việc xác thực danh sách trắng không hoạt động. Ví dụ: khi kiểm tra trường xác thực tên, O'Henry là tên hợp lệ, nhưng việc đưa vào danh sách trắng không cho phép tên đó do ký tự đặc biệt “ ‘ ” và điều này có thể tạo ra sự cố khi truy vấn SQL được tạo động như được hiển thị bên dưới:

Trong trường hợp trên, kẻ tấn công có thể đưa đầu vào độc hại vào trường first\_name như hình dưới đây:



Truy vấn kết quả thực thi như sau:



Trong Máy chủ MYSQL, một dấu nháy đơn (‘) được sử dụng để kết thúc chuỗi; do đó, mã hóa dấu nháy đơn là bắt buộc khi nó được bao gồm trong các câu lệnh SQL động. Điều này cũng có thể được thực hiện theo hai cách; dấu nháy đơn có thể được thay thế bằng hai dấu nháy đơn hoặc dấu gạch chéo ngược theo sau bằng một dấu nháy đơn. Hai phương pháp này coi câu trích dẫn đơn lẻ như một phần của chuỗi ký tự, ngăn chặn bất kỳ nỗ lực chèn SQL nào.

Ví dụ: chúng ta có thể sử dụng mã hóa đầu ra sau trong Java:

**myQuery = myQuery.replace(“ ‘ ”, “ \’ ”);**

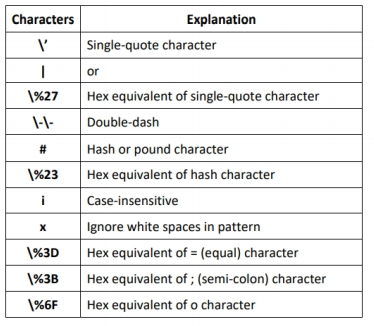
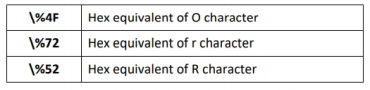
Một nhược điểm lớn của mã hóa đầu ra là đầu vào cần được mã hóa mọi lúc trước khi nó được cung cấp cho truy vấn cơ sở dữ liệu; nếu không, ứng dụng có thể trở thành nạn nhân của các cuộc tấn công SQL injection.

* **Thực thi các đặc quyền ít nhất**

Thực thi đặc quyền ít nhất là một phương pháp bảo mật tốt nhất, theo đó mức đặc quyền thấp nhất được gán cho mọi tài khoản truy cập cơ sở dữ liệu. Không nên gán quyền truy cập cấp DBA và cấp quản trị viên cho ứng dụng. Trong một số tình huống quan trọng, một số ứng dụng có thể yêu cầu quyền truy cập nâng cao; do đó, các chuyên gia bảo mật phải thực hiện các công việc cơ bản phù hợp và họ cũng phải tìm ra các yêu cầu chính xác của ứng dụng.

Ví dụ: khi chỉ cần quyền truy cập đọc cho ứng dụng, thì chỉ các đặc quyền truy cập đọc mới được cấp. Các đặc quyền tối thiểu phải được gán cho hệ điều hành nơi DBMS chạy và nó không bao giờ được chạy DBMS dưới dạng root. Do đó, bằng cách giảm thiểu các quyền truy cập, người ta có thể giảm thiểu khả năng bị truy cập trái phép và bảo vệ khỏi các cuộc tấn công SQL injection cũng như các cuộc tấn công khác.

* **Phát hiện các cuộc tấn công SQL Injection**

Các chuyên gia bảo mật phải phát triển và triển khai các quy tắc trong IDS để phát hiện các cụm từ thông dụng được sử dụng trong các cuộc tấn công đưa vào SQL trên máy chủ web. Với mục đích này, họ phải sử dụng các biểu thức chính quy để phát hiện các ký tự meta chèn SQL như dấu nháy đơn (') và dấu gạch ngang kép (-). Các biểu thức chính quy để phát hiện các ký tự cụ thể của SQL injection và ý nghĩa của chúng được liệt kê bên dưới:

### 2.2.7 ****Hậu quả của SQL Injection.****

* Hậu quả lớn nhất mà SQL Injection gây ra là: Làm lộ dữ liệu trong database. Tuỳ vào tầm quan trọng của dữ liệu mà hậu quả dao động ở mức nhẹ cho đến vô cùng nghiêm trọng.
* Lộ dữ liệu khách hàng có thể ảnh hưởng rất nghiêm trọng đến công ty. Hình ảnh công ty có thể bị ảnh hưởng, khách hàng chuyển qua sử dụng dịch vụ khác, dẫn đến phá sản v…v...
* Lỗ hỗng này cũng ảnh hưởng lớn đến khách hàng. Do họ thường dùng chung một mật khẩu cho nhiều tài khoản, chỉ cần lộ mật khẩu một tài khoản thì các tài khoản khác cũng lộ theo.
* Đây cũng là lý do chính phải mã hoá mật khẩu, nếu database có bị tấn công thì người dùng cũng không bị mất mật khẩu. (Đây là lý do vietnamwork bị phê bình vì không mã hoá mật khẩu).
* Trong nhiều trường hợp, hacker không chỉ đọc được dữ liệu mà còn có thể chỉnh sửa dữ liệu. Lúc này hacker có thể đăng nhập dưới vai trò admin, lợi dụng hệ thống, hoặc xoá toàn bộ dữ liệu để hệ thống ngừng hoạt động.

# 

# CHƯƠNG 3. KẾT QUẢ THỰC NGHIỆM

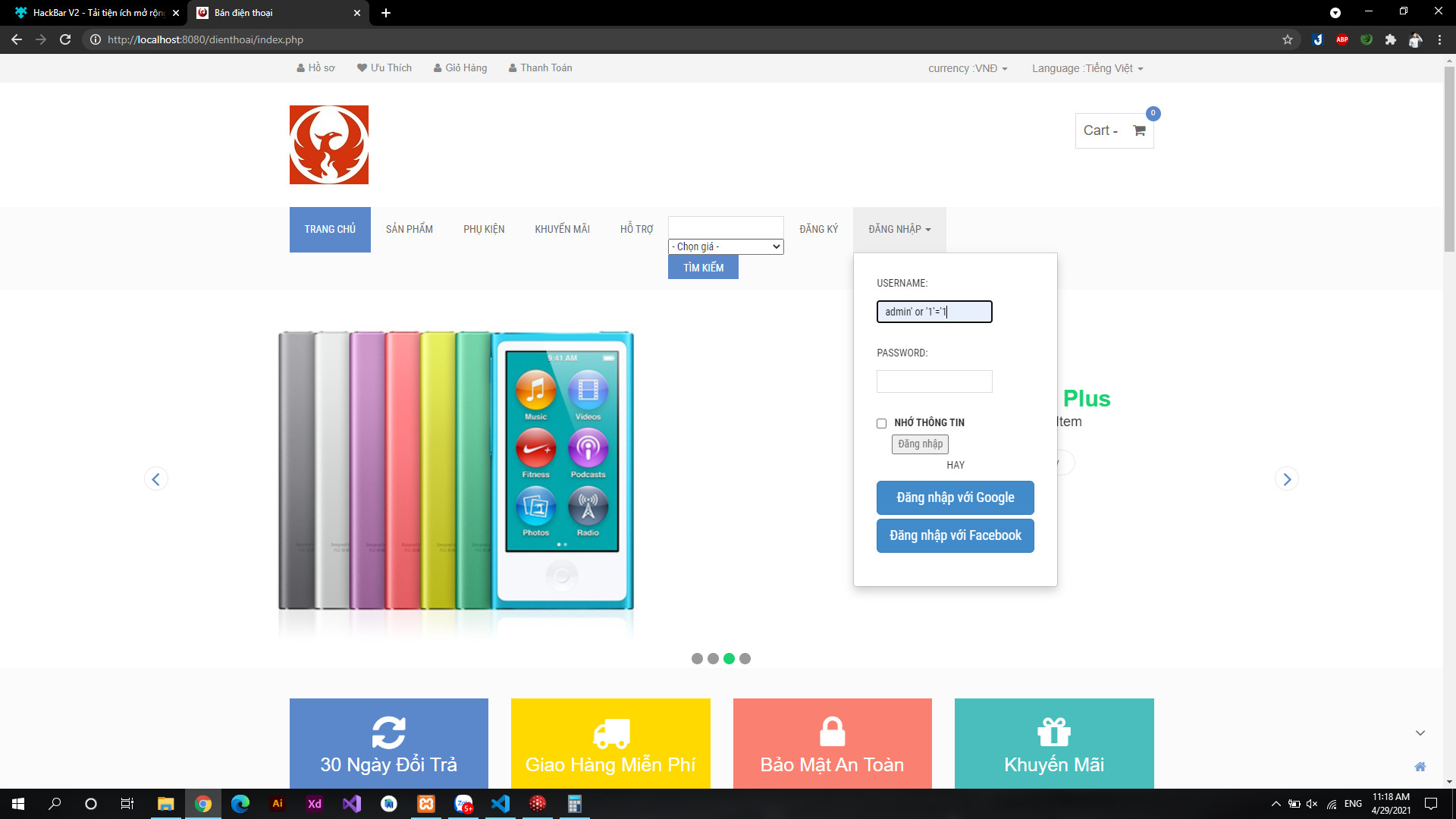
3.1. Demo kiểm tra lỗ hỏng trang web bằng công cụ Vega

Graphical user interface, application

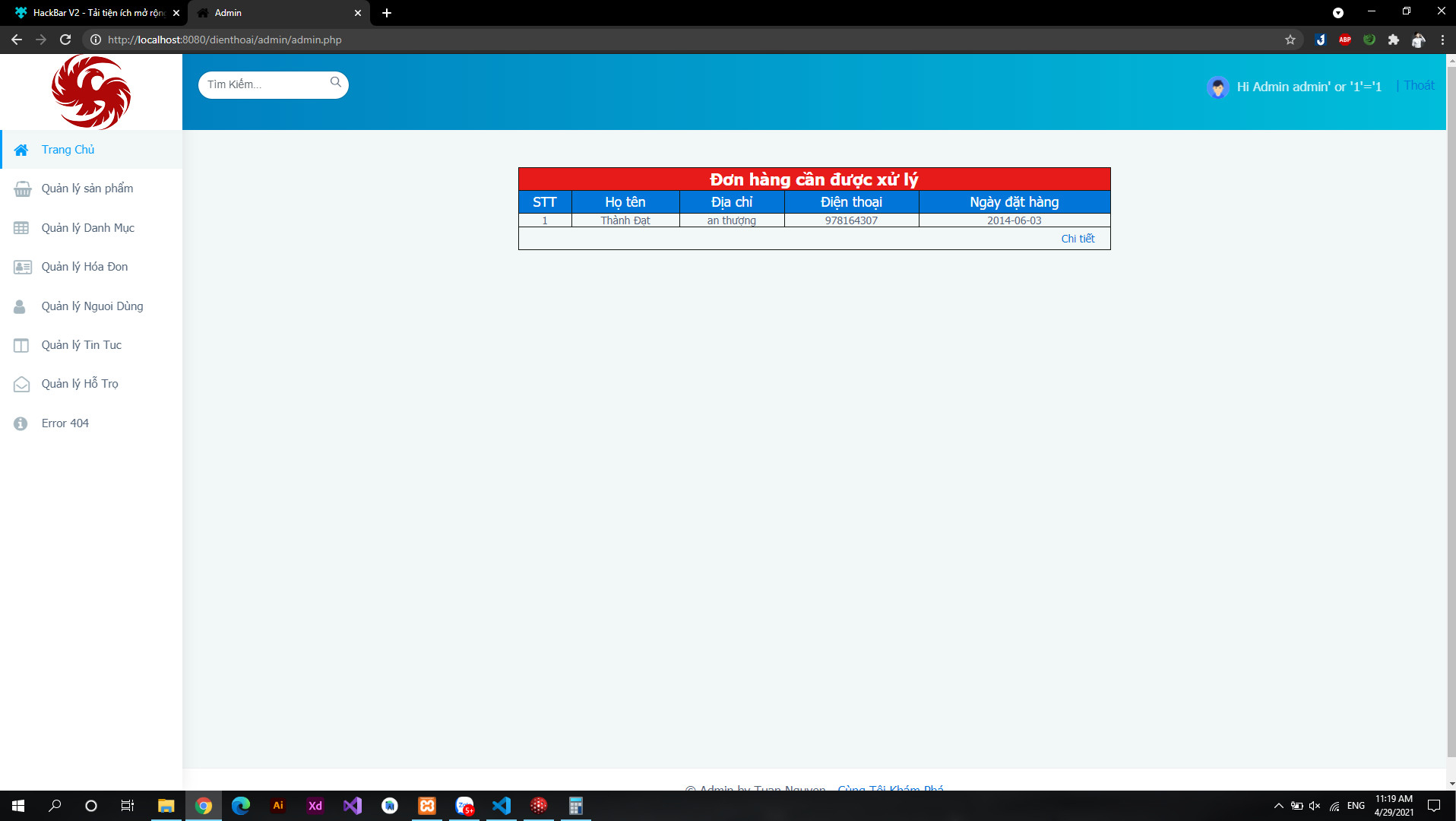
Description automatically generated

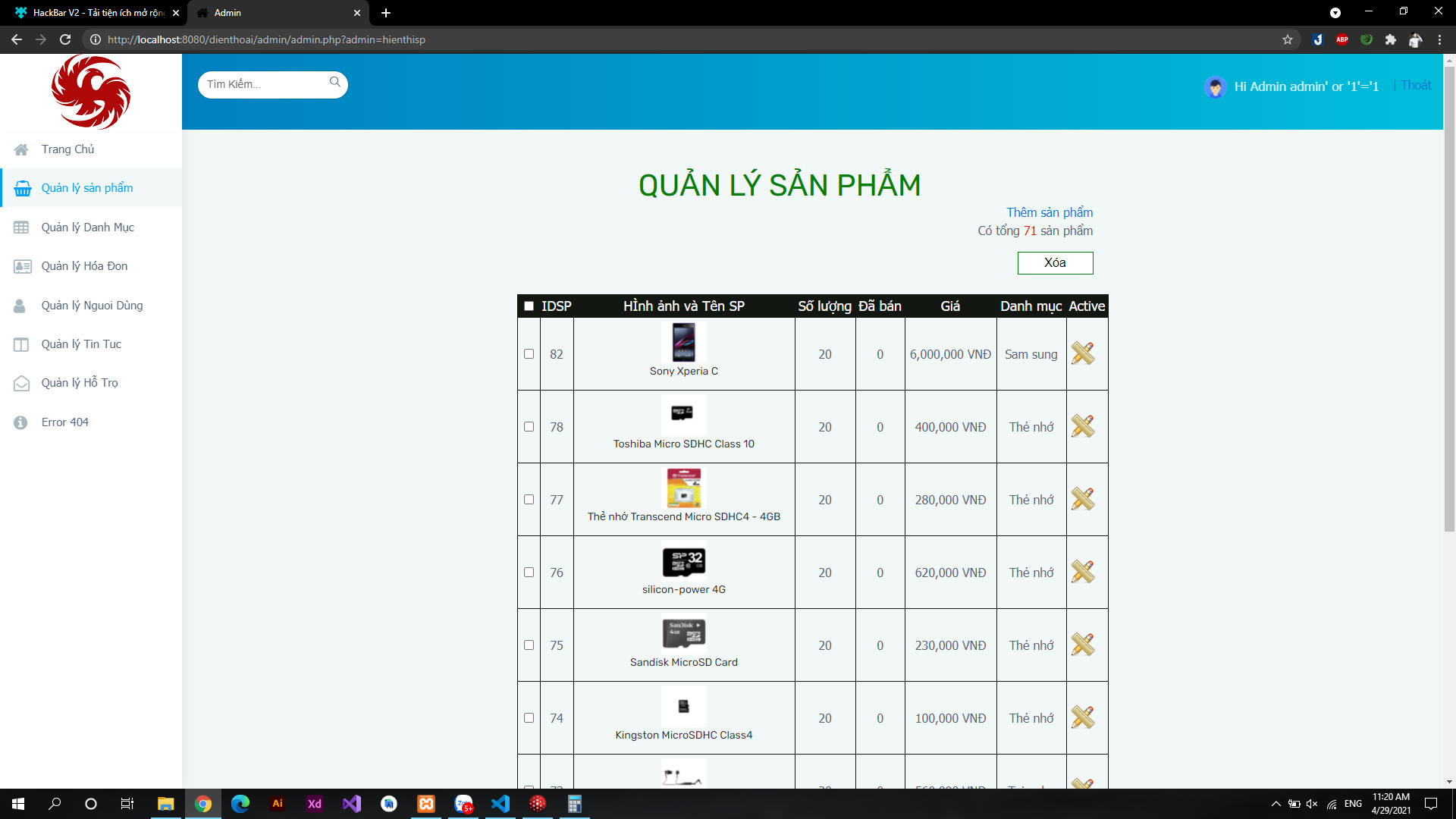
3.2. Tấn công lỗ hỏng SQL injection

Hình minh họa 1. Đăng nhập tài khoản admin bằng từ khóa luôn đúng với lỗ hỏng SQL injection



Hình minh họa 2. Đăng nhập thành công sử dụng các chức năng của một admin





3.3. Demo cách bảo về lỗ hỏng

Graphical user interface, text, application

Description automatically generated

Công cụ gợi ý cách bảo vệ

Hình ảnh minh họa code chặn các kí tự đặc biệt

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated