Tấn công SQL Injection (SQLi) thường liên quan đến việc kẻ tấn công nhập các lệnh SQL vào các trường nhập liệu của ứng dụng web để can thiệp vào truy vấn SQL do ứng dụng thực thi. Đây là cách tấn công phổ biến nhất, nhưng có nhiều cách cụ thể mà kẻ tấn công có thể thực hiện:

**1. Cách tấn công cơ bản**

* **Nhập lệnh SQL trực tiếp**: Kẻ tấn công nhập các lệnh SQL trực tiếp vào các trường nhập liệu như ô tìm kiếm, ô đăng nhập, hoặc form liên hệ để thay đổi hoặc thao túng truy vấn SQL.

Ví dụ: Nhập '; DROP TABLE users; -- vào trường nhập liệu để xóa bảng "users".

**2. Các kỹ thuật tấn công nâng cao**

* **Union-based SQL Injection**: Kẻ tấn công sử dụng câu lệnh UNION để hợp nhất kết quả của nhiều truy vấn. Điều này cho phép kẻ tấn công lấy dữ liệu từ các bảng khác.

Ví dụ: SELECT username, password FROM users WHERE id = 1 UNION SELECT credit\_card\_number, 1 FROM credit\_cards;

* **Error-based SQL Injection**: Kẻ tấn công cố gắng tạo ra lỗi SQL để trích xuất thông tin về cấu trúc cơ sở dữ liệu từ thông báo lỗi.

Ví dụ: AND 1=CONVERT(int,(SELECT @@version));

* **Blind SQL Injection**: Kẻ tấn công không nhận được kết quả trực tiếp từ truy vấn, nhưng có thể đưa ra các câu lệnh điều kiện để xác định thông tin dựa trên phản hồi của ứng dụng (chẳng hạn như đúng hoặc sai).

Ví dụ: AND 1=1 (đúng) hoặc AND 1=2 (sai).

* **Time-based Blind SQL Injection**: Kẻ tấn công sử dụng các lệnh trì hoãn thời gian để phát hiện SQL Injection dựa trên thời gian phản hồi của máy chủ.

Ví dụ: AND IF(1=1,SLEEP(5),0)

**3. Tấn công qua các điểm khác nhau**

* **Thủ tục lưu trữ (Stored Procedures)**: Tấn công qua các thủ tục lưu trữ của cơ sở dữ liệu.
* **Công cụ ORM (Object-Relational Mapping)**: Tấn công qua các lớp truy vấn tự động của ORM.

**Tóm lại**

SQL Injection chủ yếu liên quan đến việc nhập các lệnh SQL độc hại, nhưng có nhiều kỹ thuật và cách tiếp cận khác nhau mà kẻ tấn công có thể sử dụng. Việc bảo vệ ứng dụng khỏi các loại tấn công này yêu cầu các biện pháp bảo mật như kiểm tra đầu vào, sử dụng các truy vấn đã chuẩn bị (prepared statements), và các kỹ thuật ORM an toàn.

|  |  |
| --- | --- |
| **Dataset** | **Number of Queries** |
| **Malicious** | 11382 |
| **Normal** | 11384 |
| **Total** | **22766** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Model** | | **Accuracy** | **F1\_score** | **Recall** | **Precision** |
| **ML** | Random Forest | 0.996 | 0.996 | 0.994 | 0.997 |
| XGBoost | 0.995 | 0.995 | 0.993 | 0.997 |
| KNN | 0.993 | 0.993 | 0.993 | 0.993 |
| Decision Tree | 0.992 | 0.992 | 0.987 | 0.996 |
| SVM | 0.985 | 0.979 | 0.963 | 0.994 |

A graph showing a bar graph

AI-generated content may be incorrect.

**Bảng mô tả các đặc trưng của SQL Query**

| **Loại đặc trưng** | **Mô tả** |
| --- | --- |
| length | Độ dài của truy vấn SQL (tổng số ký tự). |
| whitespace\_count | Số lượng khoảng trắng trong truy vấn. |
| dangerous\_keywords | Số lần xuất hiện của các từ khóa SQL nguy hiểm như SELECT, DROP, DELETE, INSERT, UPDATE, v.v. |
| single\_quotes | Số lần xuất hiện dấu nháy đơn ' (dấu nháy có thể dùng để tạo payload SQL Injection). |
| double\_quotes | Số lần xuất hiện dấu nháy kép " (thường dùng trong truy vấn SQL). |
| backticks | Số lần xuất hiện dấu backtick ` (được sử dụng trong MySQL để bao quanh tên cột hoặc bảng). |
| comment\_markers | Số lần xuất hiện các ký tự đánh dấu comment như --, /\*, \*/ (thường dùng trong SQL Injection để bỏ qua phần còn lại của truy vấn). |
| semicolons | Số lần xuất hiện dấu ; (có thể được dùng để kết thúc một câu lệnh và bắt đầu câu lệnh SQL tiếp theo). |
| special\_chars | Số lần xuất hiện các ký tự đặc biệt như `=, (, ), <, >, |
| logical\_operators | Số lượng các toán tử logic như OR, AND, XOR, NOT, ` |
| dangerous\_functions | Số lần xuất hiện các hàm SQL nguy hiểm như char(), concat(), substring(), ascii(), hex(), v.v. |
| information\_schema | Kiểm tra xem truy vấn có chứa information\_schema hay không (có thể bị khai thác để liệt kê bảng, cột trong database). |
| table\_name | Kiểm tra xem truy vấn có chứa table\_name không (thường bị khai thác để truy vấn danh sách bảng trong database). |
| column\_name | Kiểm tra xem truy vấn có chứa column\_name không (thường bị khai thác để lấy danh sách cột trong database). |
| version\_query | Kiểm tra truy vấn có chứa version() hoặc @@version không (có thể bị dùng để xác định phiên bản database). |
| database\_query | Kiểm tra truy vấn có chứa database() hoặc schema() không (có thể bị khai thác để lấy tên database hiện tại). |
| time\_based | Kiểm tra xem truy vấn có chứa các hàm sleep(), benchmark(), pg\_sleep(), waitfor delay không (thường dùng trong SQL Injection dạng Time-based Blind). |
| file\_access | Kiểm tra truy vấn có chứa load\_file(), into outfile, into dumpfile, load data infile không (thường dùng để truy cập file hệ thống qua SQL Injection). |
| group\_concat | Kiểm tra truy vấn có chứa group\_concat() không (hàm này có thể dùng để trích xuất dữ liệu từ nhiều hàng một cách gọn gàng). |
| union\_select | Kiểm tra xem truy vấn có chứa UNION SELECT không (kỹ thuật phổ biến để kết hợp kết quả từ nhiều bảng, thường bị lạm dụng trong SQL Injection). |
| type\_conversion | Số lần xuất hiện các hàm chuyển đổi kiểu dữ liệu như CAST(), CONVERT() (có thể bị dùng để vượt qua các kiểm tra đầu vào). |
| hex\_encoding | Số lần xuất hiện các chuỗi mã hóa hex như %23, %27, %3D, %20, %22, %2D (có thể bị sử dụng để vượt qua firewall hoặc filter). |
| excessive\_whitespace | Kiểm tra xem truy vấn có khoảng trắng liên tiếp từ 3 ký tự trở lên hay không (có thể là dấu hiệu của payload SQL Injection được giấu đi). |
| stacked\_queries | Số lượng các câu truy vấn được xếp chồng (; nối nhiều truy vấn liên tiếp, thường bị cấm trong SQL Injection). |
| comment\_termination | Kiểm tra xem truy vấn có kết thúc bằng -- hoặc # không (có thể là dấu hiệu của kỹ thuật comment phần còn lại của truy vấn để tránh lỗi cú pháp). |
| boolean\_blind | Kiểm tra xem truy vấn có chứa các điều kiện boolean như 1=1, true=true, false=false, 1<>2, 0=0 không (thường bị khai thác trong SQL Injection kiểu Boolean-based Blind). |

Dưới đây là bảng mô tả 10 đặc trưng quan trọng được thể hiện trong biểu đồ:

|  |  |
| --- | --- |
| **Đặc trưng** | **Mô tả** |
| excessive\_whitespace | Kiểm tra xem truy vấn có chứa khoảng trắng liên tiếp từ 3 ký tự trở lên không. Đây có thể là dấu hiệu của payload SQL Injection được giấu đi. |
| special\_chars | Số lần xuất hiện các ký tự đặc biệt như `=, (, ), <, >, |
| whitespace\_count | Số lượng khoảng trắng trong truy vấn. Truy vấn có quá nhiều khoảng trắng có thể là dấu hiệu của một payload được cố tình làm rối. |
| length | Độ dài của truy vấn SQL (tổng số ký tự). Truy vấn quá dài có thể chứa payload tấn công. |
| comment\_termination | Kiểm tra xem truy vấn có kết thúc bằng -- hoặc # không. Đây là dấu hiệu của kỹ thuật comment phần còn lại của truy vấn để tránh lỗi cú pháp trong SQL Injection. |
| logical\_operators | Số lượng các toán tử logic như OR, AND, XOR, NOT, ` |
| comment\_markers | Số lần xuất hiện các ký tự đánh dấu comment như --, /\*, \*/. Những ký tự này có thể bị sử dụng để che giấu payload SQL Injection. |
| dangerous\_keywords | Số lần xuất hiện các từ khóa nguy hiểm như SELECT, DROP, DELETE, INSERT, UPDATE. Đây là các câu lệnh có thể bị khai thác để thao túng database. |
| single\_quotes | Số lần xuất hiện dấu nháy đơn '. Dấu nháy đơn thường được sử dụng để tạo payload SQL Injection. |
| double\_quotes | Số lần xuất hiện dấu nháy kép ". Một số hệ quản trị cơ sở dữ liệu cho phép sử dụng dấu nháy kép trong truy vấn SQL. |