BÁO CÁO BÀI THỰC HÀNH SỐ 8

KIỂM THỬ LỖ HỔNG SQL INJECTION

**Họ và tên:** Trần Thiên Nam

**MSSV:** 1050080275

**Lớp:** 10\_ĐH\_CNPM3

Bài làm

**Câu 1 (3.5 điểm)**

**Kiểm thử lỗ hổng SQL Injection trên các trang web**

**1. Xác định các tham số đầu vào và cách thức chúng được gửi tới server (0.25 điểm)**

* **Tham số đầu vào**: Bao gồm các tham số từ URL (phương thức GET), form nhập liệu (phương thức POST), trường ẩn trong form, giá trị lựa chọn trong form, hoặc tiêu đề HTTP (như Cookie).
* **Cách thức gửi**:
  + **GET**: Tham số truyền qua URL (ví dụ: ?id=1).
  + **POST**: Tham số truyền qua body của HTTP request từ form.
  + **Khác**: Cookie hoặc header HTTP.
* **Công cụ hỗ trợ**: Sử dụng Burp Suite để xác định tất cả tham số đầu vào bằng cách bắt và phân tích HTTP request.

**2. Phân tích và phán đoán kiểu truy vấn (0.25 điểm)**

* **Kiểu truy vấn**: Dựa trên hành vi của website:
  + **SELECT**: Thường xuất hiện khi website trả về dữ liệu (ví dụ: danh sách sản phẩm, thông tin người dùng).
  + **INSERT**: Khi thêm dữ liệu mới (ví dụ: thêm sản phẩm hoặc đăng ký tài khoản).
  + **UPDATE/DELETE**: Khi chỉnh sửa hoặc xóa dữ liệu.
* **Phán đoán**:
  + Quan sát URL hoặc form (ví dụ: ?cat=laptop gợi ý truy vấn SELECT).
  + Thử các giá trị như 1', any' or 1; # để xem phản hồi (lỗi hoặc trả về dữ liệu) nhằm xác định kiểu truy vấn.

**3. Phân tích và phán đoán vị trí của các giá trị tham số đầu vào trong câu truy vấn (0.5 điểm)**

* **Phân tích**:
  + Thử các giá trị đặc biệt như ' hoặc '); # để gây lỗi cú pháp SQL và quan sát thông báo lỗi.
  + Ví dụ: Nếu nhập laptop' gây lỗi cú pháp, tham số có thể nằm trong WHERE <cột> = '<tham số>'.
  + Thử any' or 1; # để xem dữ liệu trả về, xác định tham số nằm trong mệnh đề WHERE.
* **Phán đoán**:
  + Dựa trên lỗi cú pháp hoặc dữ liệu trả về, suy ra cấu trúc truy vấn (ví dụ: SELECT \* FROM table WHERE column = '<input>').
  + Sử dụng kỹ thuật như ORDER BY hoặc UNION để xác định số cột và vị trí tham số.

**4. Xác định mức độ ảnh hưởng của lỗ hổng trên mỗi giá trị đầu vào (2.5 điểm)**

* **Mục đích kiểm thử**:
  + Xác định khả năng khai thác lỗ hổng để lấy dữ liệu, sửa đổi dữ liệu, hoặc thực thi lệnh hệ thống.
  + Đánh giá mức độ nguy hiểm (ví dụ: truy cập trái phép, xóa dữ liệu, tạo backdoor).
* **Các kỹ thuật kiểm thử và giải thích**:
  + **Tautology-based**:
    - **Giá trị kiểm thử**: any' or 1; #.
    - **Mục đích**: Bỏ qua điều kiện xác thực (ví dụ: đăng nhập mà không cần mật khẩu).
    - **Kết quả**: Nếu trả về tất cả dữ liệu (ví dụ: danh sách sản phẩm), lỗ hổng cho phép truy cập trái phép.
    - **Ảnh hưởng**: Cao, vì kẻ tấn công có thể lấy toàn bộ dữ liệu từ bảng.
  + **UNION-based**:
    - **Giá trị kiểm thử**: laptop' UNION SELECT 1, database(), 3; #.
    - **Mục đích**: Lấy thông tin cơ sở dữ liệu (tên CSDL, phiên bản, người dùng).
    - **Kết quả**: Nếu trả về tên CSDL (ví dụ: webvul), lỗ hổng cho phép thăm dò cấu trúc CSDL.
    - **Ảnh hưởng**: Nghiêm trọng, vì kẻ tấn công có thể truy xuất dữ liệu nhạy cảm.
  + **Boolean-based Blind SQL Injection**:
    - **Giá trị kiểm thử**: laptop' and substring(database(),1,1)='w'; #.
    - **Mục đích**: Suy ra dữ liệu dựa trên phản hồi đúng/sai.
    - **Kết quả**: Nếu không trả về dữ liệu khi điều kiện sai, xác định được ký tự đầu của tên CSDL.
    - **Ảnh hưởng**: Trung bình, vì cần nhiều truy vấn để lấy dữ liệu, nhưng vẫn nguy hiểm.
  + **Time-based Blind SQL Injection**:
    - **Giá trị kiểm thử**: laptop' UNION SELECT 1, IF(SUBSTRING(database(),1,1)='w',BENCHMARK(500000,ENCODE('a','b')),null), 3; #.
    - **Mục đích**: Suy ra dữ liệu dựa trên thời gian phản hồi.
    - **Kết quả**: Nếu truy vấn chậm khi điều kiện đúng, xác định được thông tin CSDL.
    - **Ảnh hưởng**: Trung bình, tương tự Boolean-based, nhưng khó phát hiện hơn.
  + **Batched Query (Stacked Query)**:
    - **Giá trị kiểm thử**: any'; INSERT INTO users (username, password) VALUES ('hacker', 'pass'); #.
    - **Mục đích**: Thực thi nhiều câu lệnh SQL (thêm, xóa, sửa dữ liệu).
    - **Kết quả**: Nếu dữ liệu mới xuất hiện, lỗ hổng cho phép thay đổi dữ liệu.
    - **Ảnh hưởng**: Rất cao, vì có thể phá hủy hoặc chiếm quyền hệ thống.
* **Kết luận**: Mức độ ảnh hưởng phụ thuộc vào:
  + Loại truy vấn (SELECT, INSERT, UPDATE, DELETE).
  + Quyền của tài khoản CSDL (ví dụ: root có thể thực thi lệnh hệ thống).
  + Khả năng khai thác (truy xuất dữ liệu, sửa đổi, hoặc tạo backdoor).

**Câu 2 (3.5 điểm)**

**Kiểm thử lỗ hổng SQL Injection trên trang web** [**http://webvul.bkcs.vn/login.php**](http://webvul.bkcs.vn/login.php)

**1. Xác định các tham số đầu vào và cách thức chúng được gửi tới server (0.25 điểm)**

* **Tham số đầu vào**:
  + username và password từ form đăng nhập.
* **Cách thức gửi**:
  + Phương thức **POST**, dữ liệu được gửi qua body của HTTP request.
* **Công cụ**: Sử dụng Burp Suite để bắt request POST và xác định tham số:

text

Copy

POST /login.php HTTP/1.1

Content-Type: application/x-www-form-urlencoded

username=test&password=test

**2. Phân tích và phán đoán kiểu truy vấn, vị trí của các tham số trong câu truy vấn (0.25 điểm)**

* **Kiểu truy vấn**:
  + Form đăng nhập thường sử dụng truy vấn **SELECT** để kiểm tra thông tin người dùng.
  + Phán đoán: SELECT \* FROM users WHERE username='<username>' AND password='<password>'.
* **Vị trí tham số**:
  + username và password nằm trong mệnh đề WHERE, được so sánh trực tiếp với cột trong bảng.
  + Thử giá trị username=test' gây lỗi cú pháp, xác nhận tham số nằm trong dấu nháy đơn ('<input>').
  + Thử username=test' or 1; # trả về kết quả đăng nhập thành công, xác nhận tham số username có thể thao túng điều kiện WHERE.

**3. Xác định mức độ ảnh hưởng của lỗ hổng trên mỗi giá trị đầu vào (3 điểm)**

* **Mục đích kiểm thử**:
  + Kiểm tra khả năng vượt qua xác thực, lấy thông tin CSDL, hoặc thực thi lệnh nguy hiểm.
* **Kiểm thử trên tham số username**:
  + **Tautology-based**:
    - **Giá trị**: admin' or 1; #.
    - **Kết quả**: Đăng nhập thành công mà không cần mật khẩu.
    - **Giải thích**: Biểu thức or 1 luôn đúng, bỏ qua điều kiện mật khẩu.
    - **Ảnh hưởng**: Cao, cho phép truy cập trái phép vào tài khoản bất kỳ.
  + **UNION-based**:
    - **Giá trị**: admin' UNION SELECT 1, database(), 3; #.
    - **Kết quả**: Trả về tên CSDL (ví dụ: webvul).
    - **Giải thích**: Kết hợp kết quả truy vấn gốc với truy vấn lấy thông tin CSDL.
    - **Ảnh hưởng**: Nghiêm trọng, lộ thông tin cấu trúc CSDL.
  + **Boolean-based Blind SQL Injection**:
    - **Giá trị**: admin' and substring(database(),1,1)='w'; #.
    - **Kết quả**: Đăng nhập thành công nếu tên CSDL bắt đầu bằng w, ngược lại thất bại.
    - **Giải thích**: Suy ra từng ký tự của tên CSDL dựa trên phản hồi.
    - **Ảnh hưởng**: Trung bình, cần nhiều truy vấn nhưng vẫn nguy hiểm.
  + **Time-based Blind SQL Injection**:
    - **Giá trị**: admin' AND IF(SUBSTRING(database(),1,1)='w',BENCHMARK(500000,ENCODE('a','b')),null); #.
    - **Kết quả**: Phản hồi chậm nếu tên CSDL bắt đầu bằng w.
    - **Giải thích**: Thời gian phản hồi xác nhận điều kiện đúng/sai.
    - **Ảnh hưởng**: Trung bình, khó khai thác hơn nhưng vẫn khả thi.
* **Kiểm thử trên tham số password**:
  + Thử tương tự như username. Nếu password cũng dễ bị tấn công (ví dụ: pass' or 1; #), mức độ ảnh hưởng tương tự.
* **Kết luận**:
  + Lỗ hổng trên username và password cho phép vượt qua xác thực, lấy thông tin CSDL, và có thể khai thác dữ liệu nhạy cảm.
  + Mức độ ảnh hưởng: **Rất cao**, đặc biệt nếu tài khoản CSDL có quyền cao (ví dụ: root).

**Câu 3 (3 điểm)**

**Kiểm thử lỗ hổng SQL Injection trên trang web** [**http://webvul.bkcs.vn/index.php**](http://webvul.bkcs.vn/index.php)

**1. Xác định các tham số đầu vào và cách thức chúng được gửi tới server (0.25 điểm)**

* **Tham số đầu vào**:
  + Tham số trong URL (ví dụ: id trong index.php?id=1) hoặc form tìm kiếm (nếu có).
* **Cách thức gửi**:
  + **GET**: Tham số truyền qua URL.
  + **POST**: Nếu có form, tham số truyền qua body.
* **Công cụ**: Burp Suite để bắt request và xác định tham số (ví dụ: GET /index.php?id=1).

**2. Phân tích và phán đoán kiểu truy vấn, vị trí của các tham số trong câu truy vấn (0.25 điểm)**

* **Kiểu truy vấn**:
  + Trang index.php thường hiển thị nội dung động (ví dụ: bài viết, sản phẩm), gợi ý truy vấn **SELECT**.
  + Phán đoán: SELECT \* FROM table WHERE id=<id>.
* **Vị trí tham số**:
  + Tham số id nằm trong mệnh đề WHERE, có thể là số (id=1) hoặc chuỗi (id='1').
  + Thử id=1' hoặc id=1' or 1; # để xác định tham số có trong dấu nháy đơn hay không.
  + Nếu id=1' or 1; # trả về tất cả dữ liệu, tham số id nằm trong WHERE id='<id>'.

**3. Xác định mức độ ảnh hưởng của lỗ hổng trên mỗi giá trị đầu vào (2.5 điểm)**

* **Mục đích kiểm thử**:
  + Kiểm tra khả năng lấy dữ liệu, thăm dò CSDL, hoặc thực thi lệnh nguy hiểm.
* **Kiểm thử trên tham số id**:
  + **Tautology-based**:
    - **Giá trị**: 1' or 1; #.
    - **Kết quả**: Trả về tất cả bản ghi (ví dụ: tất cả bài viết).
    - **Giải thích**: Biểu thức or 1 bỏ qua điều kiện lọc, lấy toàn bộ dữ liệu.
    - **Ảnh hưởng**: Cao, lộ toàn bộ dữ liệu trong bảng.
  + **UNION-based**:
    - **Giá trị**: 1' UNION SELECT 1, user(), 3; #.
    - **Kết quả**: Trả về thông tin người dùng CSDL (ví dụ: root@localhost).
    - **Giải thích**: Kết hợp truy vấn gốc với truy vấn lấy thông tin hệ thống.
    - **Ảnh hưởng**: Nghiêm trọng, lộ thông tin nhạy cảm.
  + **ORDER BY**:
    - **Giá trị**: 1' ORDER BY 2; #.
    - **Kết quả**: Nếu thành công, xác định truy vấn có ít nhất 2 cột.
    - **Giải thích**: Giúp thăm dò cấu trúc bảng.
    - **Ảnh hưởng**: Trung bình, hỗ trợ khai thác sâu hơn.
  + **Boolean-based Blind SQL Injection**:
    - **Giá trị**: 1' and substring(version(),1,1)='5'; #.
    - **Kết quả**: Trả về dữ liệu nếu phiên bản MySQL bắt đầu bằng 5.
    - **Giải thích**: Suy ra thông tin dựa trên phản hồi đúng/sai.
    - **Ảnh hưởng**: Trung bình, cần nhiều truy vấn.
  + **Time-based Blind SQL Injection**:
    - **Giá trị**: 1' AND IF(SUBSTRING(version(),1,1)='5',BENCHMARK(500000,ENCODE('a','b')),null); #.
    - **Kết quả**: Phản hồi chậm nếu điều kiện đúng.
    - **Giải thích**: Xác nhận thông tin qua thời gian phản hồi.
    - **Ảnh hưởng**: Trung bình, khó khai thác nhưng khả thi.
* **Kết luận**:
  + Lỗ hổng trên tham số id cho phép lấy dữ liệu, thăm dò CSDL, và có thể khai thác thêm nếu quyền CSDL cao.
  + Mức độ ảnh hưởng: **Cao**, vì có thể truy xuất dữ liệu nhạy cảm và thăm dò hệ thống.

**Lưu ý chung**

* **Mục đích các giá trị kiểm thử**:
  + Các giá trị như ' hoặc '); # nhằm gây lỗi cú pháp để xác định lỗ hổng.
  + Các kỹ thuật như or 1, UNION, ORDER BY, hoặc BENCHMARK nhằm khai thác dữ liệu hoặc thăm dò cấu trúc CSDL.
* **Giải thích phán đoán**:
  + Thông báo lỗi cú pháp (ví dụ: MySQL syntax error) xác nhận tham số không được xử lý an toàn.
  + Dữ liệu trả về bất thường (ví dụ: tất cả bản ghi) hoặc thời gian phản hồi chậm xác nhận lỗ hổng có thể khai thác.
* **Công cụ hỗ trợ**: Burp Suite giúp bắt và sửa đổi request để thử các giá trị đầu vào phức tạp.
* **Khuyến nghị**: Để phòng ngừa, cần sử dụng prepared statements, kiểm tra đầu vào (input sanitization), và hạn chế quyền của tài khoản CSDL.