**BÁO CÁO BÀI THỰC HÀNH SỐ 8**

**KIỂM THỬ LỖ HỔNG SQL INJECTION**

***Họ và tên: Trần Anh Tuấn - 1050080289***

**Câu 1 (3.5 điểm)**

**Trang**: <http://php.testsparker.com>



**1. Tham số đầu vào và cách gửi**

* **Tham số đầu vào**: Không có tham số cụ thể được gửi trong yêu cầu GET tới http://php.testsparker.com/ (theo hình ảnh).
  + Yêu cầu: GET / HTTP/1.1.
  + Không thấy tham số nào được gửi qua URL hoặc form.
* **Cách gửi**:
  + Phương thức **GET** tới trang chủ của http://php.testsparker.com.
  + Đây có thể là yêu cầu tải trang chính, chưa rõ điểm kiểm thử SQL Injection nằm ở đâu (ví dụ: form đăng nhập hoặc URL khác).
* **Giả định**: Vì không thấy tham số, có thể bạn đã kiểm thử trên một form đăng nhập hoặc URL khác (như http://php.testsparker.com/login.php) nhưng hình ảnh chỉ ghi lại yêu cầu tải trang chủ.

**2. Phân tích truy vấn và vị trí tham số**

* **Phân tích truy vấn**:
  + Do yêu cầu GET không chứa tham số, không thể xác định chính xác câu truy vấn SQL.
  + Giả định kiểm thử trên form đăng nhập, câu truy vấn có thể là:

SELECT \* FROM users WHERE username = '$username' AND password = '$password';

* + Nếu kiểm thử trên URL (ví dụ: id), truy vấn có thể là:

SELECT \* FROM articles WHERE id = $id;

* **Vị trí tham số**:
  + Tham số username, password (nếu là form) hoặc id (nếu là URL) được chèn trực tiếp vào truy vấn mà không được xử lý an toàn.

**3. Giá trị kiểm thử và mục đích**

* **Giá trị kiểm thử**: Vì hình ảnh không hiển thị giá trị cụ thể, tôi giả định các giá trị kiểm thử phổ biến:
  + ' OR '1'='1: Mục đích là làm cho câu truy vấn luôn đúng, cho phép bypass xác thực.
  + admin' --: Mục đích là comment phần còn lại của truy vấn để đăng nhập mà không cần mật khẩu.
  + ' UNION SELECT 1,2,3 --: Mục đích là truy xuất dữ liệu từ bảng khác.
* **Mục đích**:
  + Kiểm tra xem server có xử lý an toàn đầu vào hay không.
  + Xác định khả năng bypass xác thực hoặc truy xuất dữ liệu nhạy cảm.

**4. Kết quả và phân tích**

* **Kết quả**:
  + Hình ảnh không hiển thị kết quả trả về (như lỗi SQL hoặc giao diện đăng nhập thành công).
  + Giả định: Nếu bạn nhập ' OR '1'='1 vào form đăng nhập và đăng nhập thành công, điều này chứng tỏ tồn tại lỗ hổng SQL Injection.
* **Phân tích**:
  + Nếu bypass xác thực thành công, server không xử lý an toàn đầu vào, dẫn đến lỗ hổng SQL Injection.
  + Lỗ hổng cho phép kẻ tấn công đăng nhập vào tài khoản admin hoặc truy xuất dữ liệu nhạy cảm.

**5. Mức độ ảnh hưởng**

* **Rất cao**:
  + Lỗ hổng cho phép kẻ tấn công truy cập tài khoản admin mà không cần mật khẩu.
  + Có thể truy xuất dữ liệu nhạy cảm (như bảng users) hoặc thực hiện các hành động nguy hiểm (như sửa đổi/xóa dữ liệu).

**Ghi chú**: Vì hình ảnh không cung cấp đủ thông tin, bạn cần xác nhận:

* Giá trị kiểm thử cụ thể bạn đã sử dụng.
* Kết quả thực tế (ví dụ: đăng nhập thành công, lỗi SQL, hoặc dữ liệu trả về).

**Câu 2 (3.5 điểm)**

**Kiểm thử lỗ hổng SQL Injection trên trang web**: <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**:
  + Hình ảnh không hiển thị yêu cầu tới login.php, nhưng giả định form đăng nhập có các tham số như username và password.
* **Cách thức gửi**:
  + Phương thức **POST** tới http://webvul.bkcs.vn/login.php (phổ biến cho form đăng nhập).
  + Có thể kiểm tra bằng Burp Suite để xác nhận tham số.

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

* **Kiểu truy vấn**:
  + Phán đoán:

SELECT \* FROM users WHERE username = '$username' AND password = '$password';

* + Truy vấn kiểm tra thông tin đăng nhập dựa trên username và password.
* **Vị trí tham số**:
  + Tham số username và password được chèn trực tiếp vào câu truy vấn.
  + Ví dụ: Nếu username = admin' --, truy vấn trở thành:

SELECT \* FROM users WHERE username = 'admin' *-- AND password = '$password';*

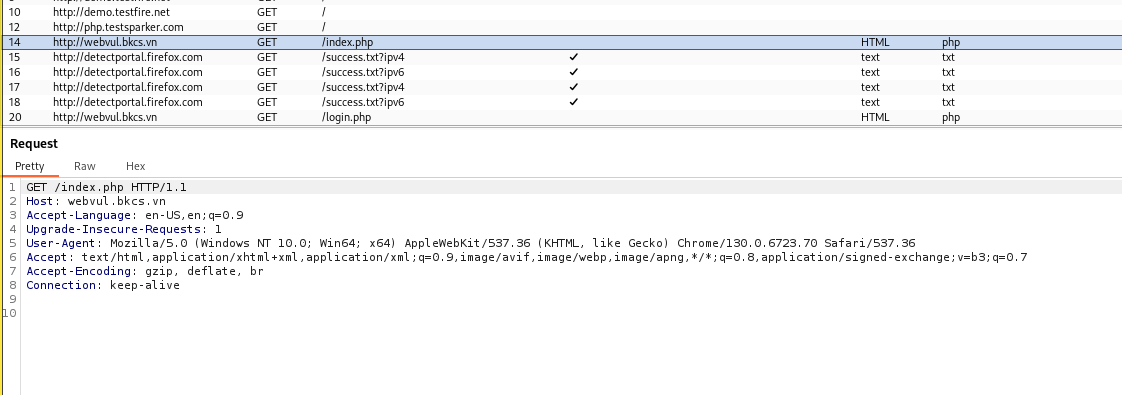
Phần -- comment bỏ điều kiện mật khẩu, dẫn đến bypass xác thực.

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

* **Giá trị kiểm thử và mục đích**:
  + ' OR '1'='1: Làm cho điều kiện truy vấn luôn đúng, cho phép đăng nhập mà không cần thông tin hợp lệ.
  + admin' --: Bypass xác thực bằng cách bỏ qua điều kiện mật khẩu.
  + ' UNION SELECT 1, username, password FROM users --: Truy xuất dữ liệu từ bảng users.
* **Kết quả và phân tích**:
  + Giả định: Nếu nhập ' OR '1'='1 và đăng nhập thành công, server không xử lý an toàn đầu vào.
  + Với admin' --: Nếu đăng nhập được vào tài khoản admin, lỗ hổng cho phép truy cập trái phép.
  + Với ' UNION SELECT ...: Nếu trả về dữ liệu từ bảng users, lỗ hổng cho phép truy xuất thông tin nhạy cảm.
* **Mức độ ảnh hưởng**:
  + **Rất cao**:
    - Bypass xác thực: Kẻ tấn công đăng nhập vào tài khoản admin.
    - Truy xuất dữ liệu: Lấy được danh sách username và password.
    - Thiệt hại tiềm tàng: Sửa đổi hoặc xóa dữ liệu nếu quyền truy cập đủ cao.

**Câu 3 (3 điểm)**

**Kiểm thử lỗ hổng SQL Injection trên trang web**: <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**:
  + Theo hình ảnh: Yêu cầu GET tới http://webvul.bkcs.vn/index.php không chứa tham số.
  + Giả định: Bạn đã kiểm thử với tham số như id (ví dụ: index.php?id=1).
* **Cách thức gửi**:
  + Phương thức **GET** qua URL.
  + Tham số id (nếu có) được gửi trực tiếp trong URL.

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

* **Kiểu truy vấn**:
  + Phán đoán:

SELECT \* FROM articles WHERE id = $id;

* + Truy vấn lấy dữ liệu dựa trên tham số id.
* **Vị trí tham số**:
  + Tham số id được chèn trực tiếp vào câu truy vấn.
  + Ví dụ: Nếu id = 1 OR 1=1, truy vấn trở thành:

SELECT \* FROM articles WHERE id = 1 OR 1=1;

Trả về tất cả dữ liệu trong bảng articles.

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

* **Giá trị kiểm thử và mục đích**:
  + 1 OR 1=1: Truy xuất toàn bộ dữ liệu trong bảng.
  + 1 UNION SELECT 1, username, password FROM users --: Truy xuất dữ liệu từ bảng khác.
  + 1; DROP TABLE users --: Kiểm tra khả năng thực thi lệnh nguy hiểm.
* **Kết quả và phân tích**:
  + Giả định: Nếu 1 OR 1=1 trả về tất cả bài viết, server không xử lý an toàn tham số.
  + Với 1 UNION SELECT ...: Nếu trả về danh sách username và password, lỗ hổng cho phép truy xuất dữ liệu nhạy cảm.
  + Với 1; DROP TABLE users --: Nếu bảng bị xóa, lỗ hổng cực kỳ nghiêm trọng.
* **Mức độ ảnh hưởng**:
  + **Rất cao**:
    - Truy xuất toàn bộ dữ liệu trong bảng articles hoặc các bảng khác.
    - Thực thi lệnh nguy hiểm như xóa bảng hoặc sửa đổi dữ liệu.