Báo cáo xây dựng Web, đánh giá ATTT và fix lỗi

--Demo: Nguyễn Đình Dũng—

1. Xây dựng web

1.1 Công nghệ sử dụng

- Môi trường cài đặt: Xampp trên Window, web server trên Linux

- Program: PHP, HTML

- Database: MySQL ( DB: bảng user, bảng posts,bảng comments, bảng avatars)



1.2 Tính năng trang web:

Trang web blog cá nhân (PHP thuần) bao gồm các tính năng:

- Đăng ký, đăng nhập, đăng xuất.

- Thêm, sửa, xóa blog;comment

- Cập nhật **password**,avatar

1.2.1 Tính năng đăng ký ( register)

Trang đăng kí cho phép người dùng tạo tài khoản để sử dụng các chức năng.



1.2.2 Tính năng đăng nhập (login)

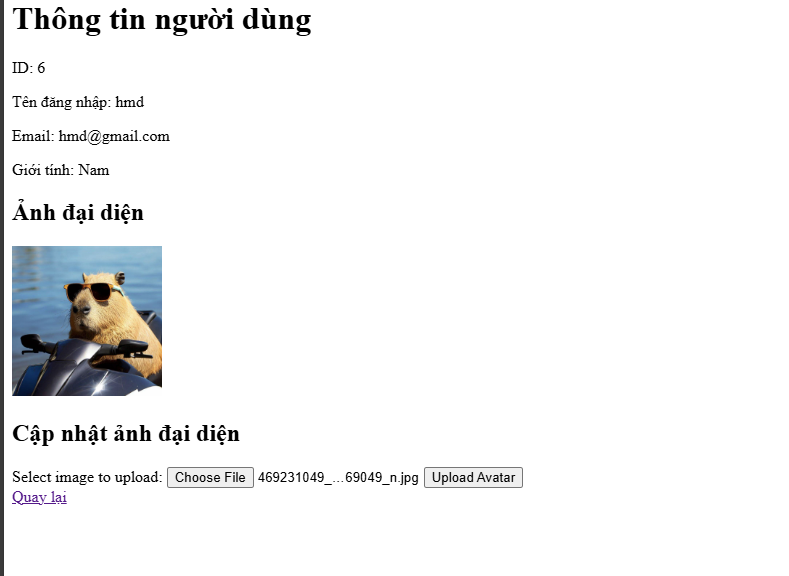
Trang login cho phép người dùng đăng nhập, in thông báo lỗi nếu nhập sai dữ liệu



1.2.3 Xem thông tin của người dùng

- Xem thông tin người dùng (id,username,email, giới tính)

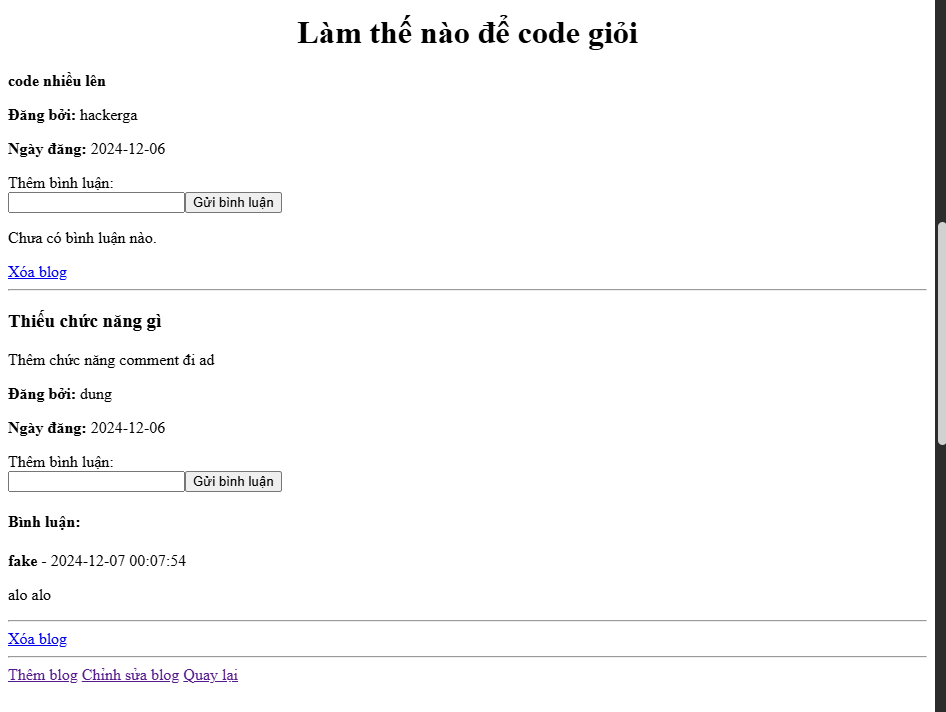
- **Tính năng thay đổi mật khẩu**, upload avatar ( **mỗi user sẽ có 1 avatar mặc định** )



1.2.4 Quản lý các blog người dùng

- Trang xem các blog, comment của người dùng

- Thêm sửa xóa blog



1.2.5 Tính năng đăng xuất

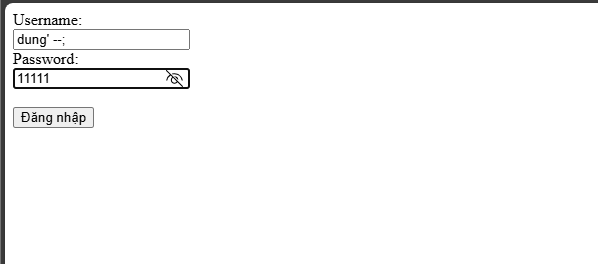
Xóa session hiện tại và quay trở về trang chủ.

2. Đánh giá An toàn trang Web và fix lỗi

2.1 Lỗi SQL Injection ở form đăng nhập:

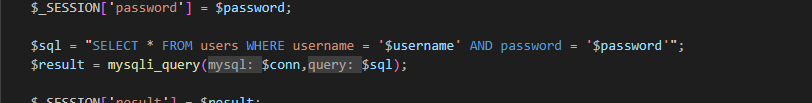
2.1.1 POC khai thác

Form đăng nhập phía người dùng nhận vào username và password:



Hình 2.1: POC payload SQLi

Đoạn code xử lý và truy vấn thông tin đăng nhập:



Hình 2.2: Code xử lý đăng nhập

Đoạn code xử lý trên dính lỗi SQLi do việc dữ liệu đầu vào của người dùng được xử lý trực tiếp vào truy vấn SQL:

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

Giả sử kẻ tấn công có username là ‘dung’, nếu sử dụng payload ‘dung’ – ‘, đoạn ‘ ‘- -‘ sẽ comment đoạn truy vấn còn lại, khi đó câu trên sẽ trở thành:

$sql = "SELECT \* FROM users WHERE username = 'dung'--' AND password = '$password'";

Truy vấn thực thi và trả về kết quả, truy cập và kiểm tra thông tin người dùng:

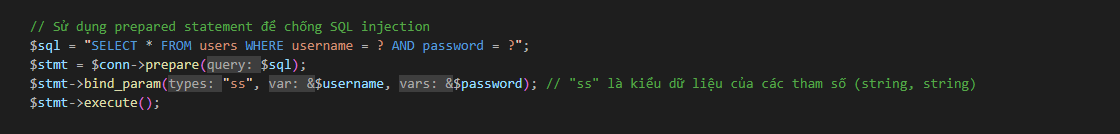


Hình 2.3 Trang thông tin cá nhân user

2.1.2 Ngăn chặn

Cách hiệu quả nhất là sử dụng SQL prepared statements.

Prepared statements tách biệt phần mã lệnh SQL và dữ liệu người dùng (sử dụng các placeholdes thay cho dữ liệu đầu vào) , giúp ngăn chặn việc chèn các mã độc hại vào câu lệnh.



Hình 2.4: Sử dụng prepare() và bind\_param()

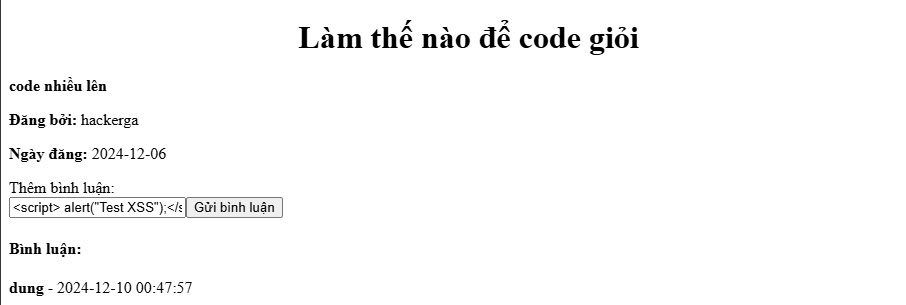
2.2 Lỗi Cross-Site Scripting (XSS)

- Lỗi XSS xảy ra do dữ liệu đầu vào không được xử lý, sau đó được lưu trữ, hiển thị lại trên trang web, gây thực thi script thực hiện các hành động.

2.2.1 POC khai thác

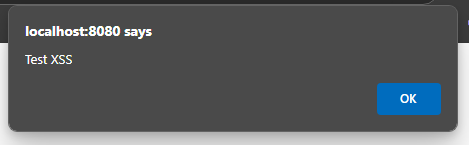
Tại chức năng comment, cho phép người dùng bình luận, test thử payload sau:

<script> alert(“Test XSS”);</script>



Hình 2.5: Thực hiện comment

Sau khi ấn Gửi bình luận, trang web hiện ra thông báo “TEST XSS”:



Hình 2.6: Alert “XSS” xuât hiện -> dính XSS

BỔ SUNG lấy COOKIE

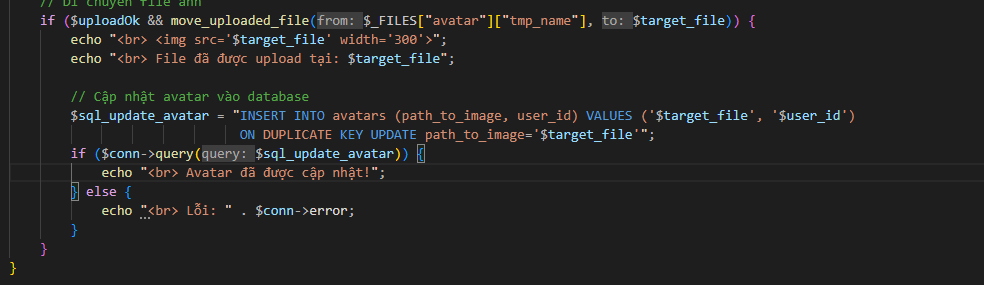
2.3 RCE thông lỗi File Upload (Reverse-shell)

2.3.1 POC khai thác

Trong mục xem thông tin người dùng, có tính năng upload file avatar:



Đoạn mã nguồn xử lý file avatar tải lên như sau:



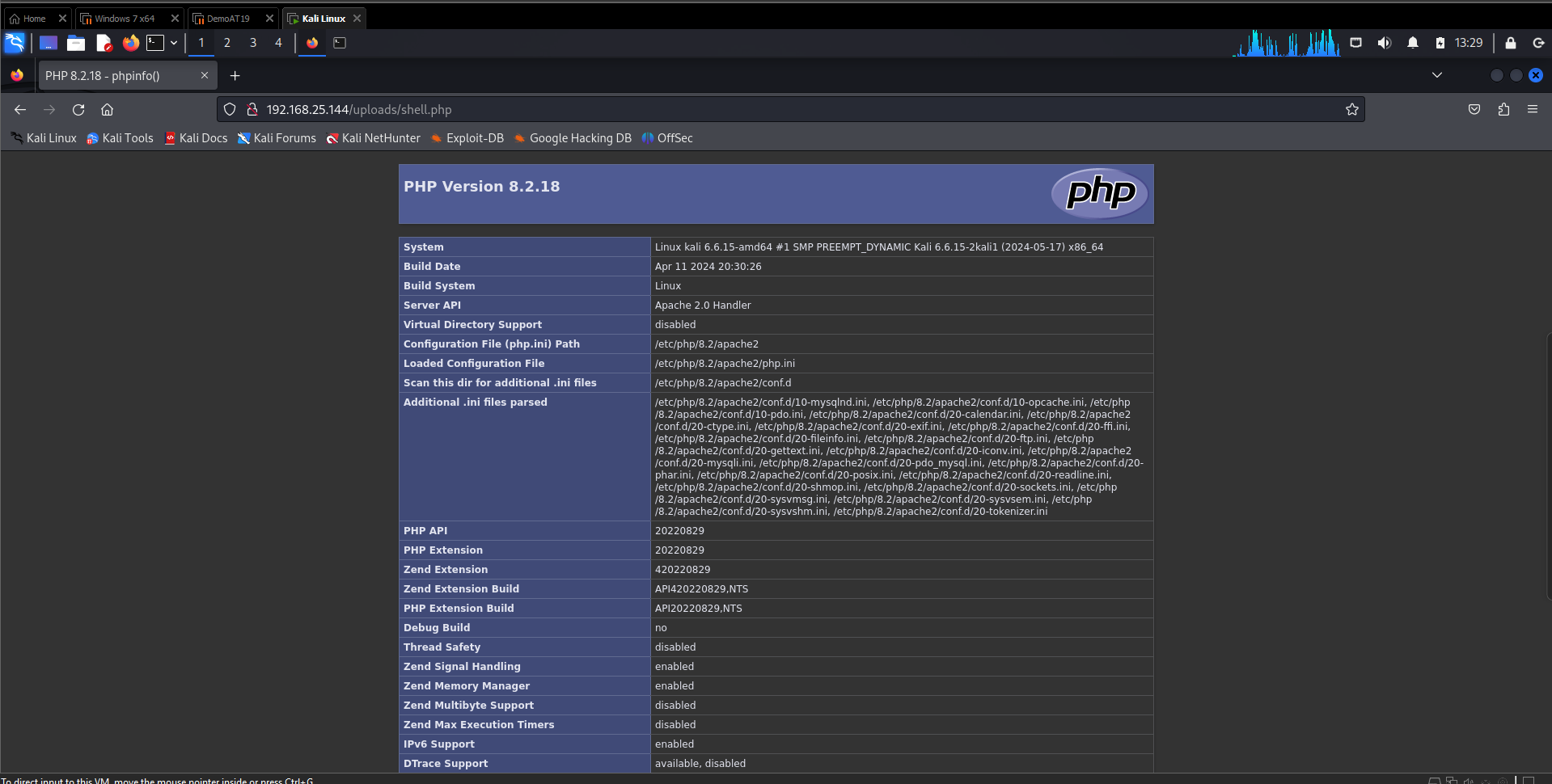
Các đoạn mã xử file upload avatar trên không có các biện pháp kiểm tra đầu vào định dạng, loại file, sự tồn tại của file, do đó có thể gây ra các vấn đề ghi đè file, upload shell RCE…

Tiến hành khai thác upload 1 file shell.php có nội dung:

<?php phpinfo(); ?>

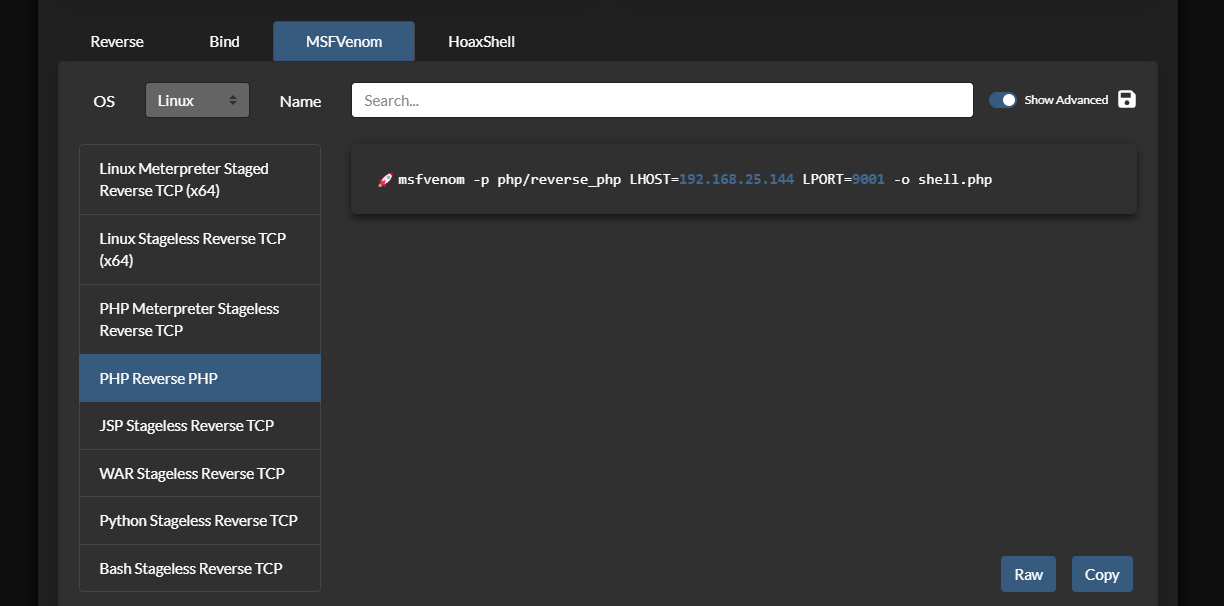


Từ đây ta có thể up lên 1 file php do ta kiểm soát:

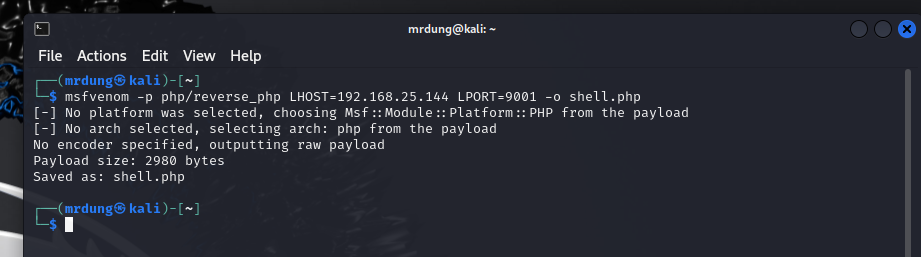


Hình 2.7: POC upload Shell

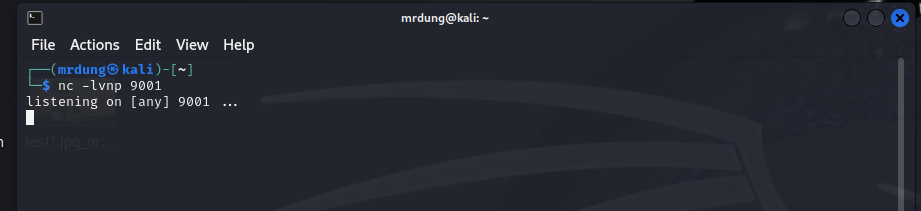
Tiến hành tạo 1 file reverse-shell PHP để khai thác như sau:



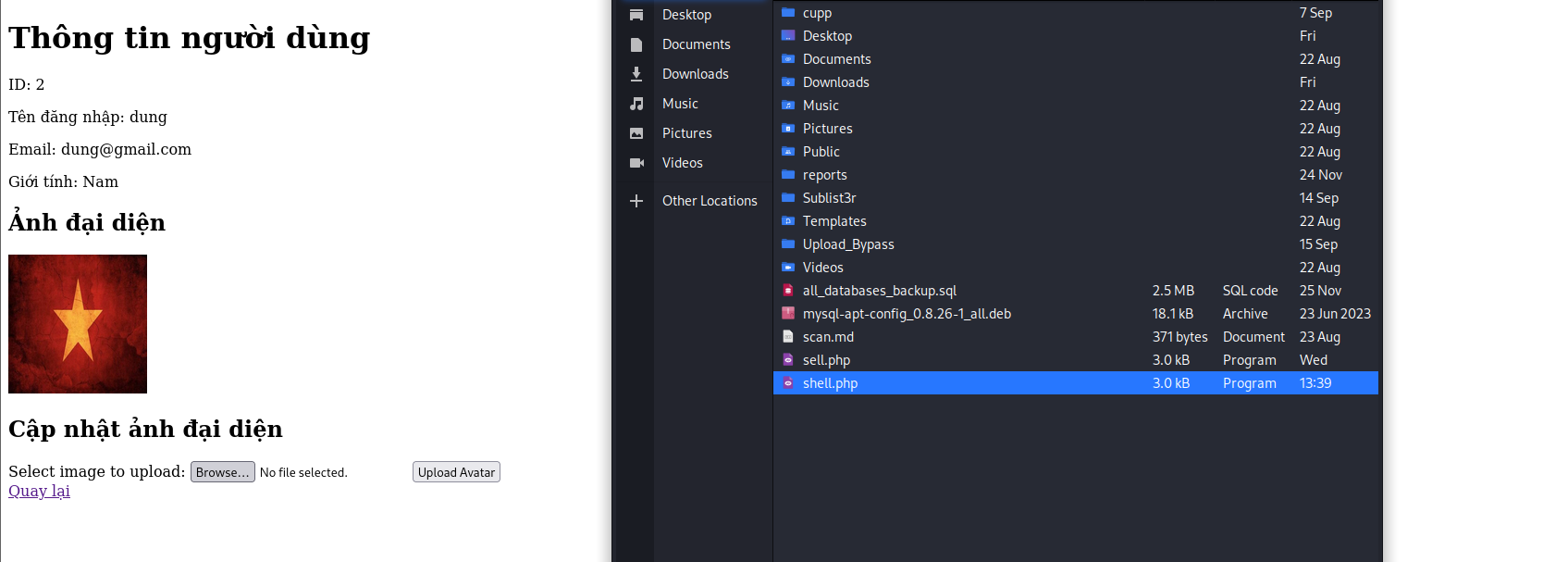
Mở termial kali-linux, chạy lệnh tạo shell.php sử dụng msfvenom:

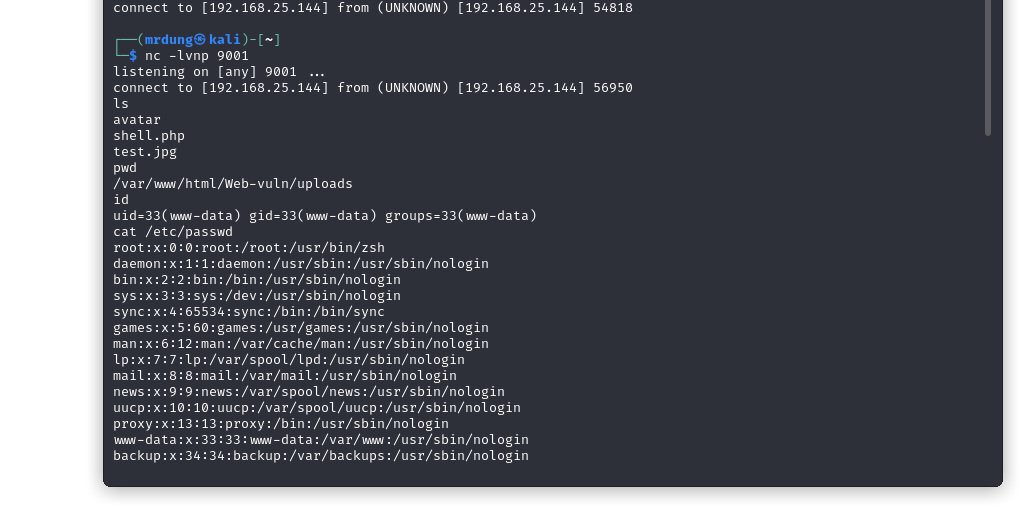


Tạo một listener nhận kết nối đến bằng netcat:



Tiến hành upload file shell.php trong mục upload avatar:





Sau khi upload file thành công, netcat nhận được một kết nối shell đến port 9001, kiểm tra, thực thi thử một số lệnh cơ bản như ls, pwd, cat /etc/passwd đều trả về kết quả -> Exploit thành công RCE via File Upload.

2.3.2 Ngăn chặn

- Sử dụng basename() để hạn chế Path Traversal.

- Thêm xử lý kiểm tra MIME type, kiểm tra đuôi file hợp lệ như jpg,png,jpeg, kiểm tra tính hợp lệ (width, height), giới hạn kích cỡ tải lên

3. Tổng kết

3.1. Tự xây dựng Web

- Tự build một web blog chứ vuln chạy PHP, apache kết nối MySQL

- Nắm được và thực hành logic việc nhận Request và xử lý phía Backend

3.2 Đánh giá và fix lỗi

- Tìm ra 1 số lỗ hổng như SQLi, XSS, File Upload trên trang web tự xây dựng

- Tham khảo và fix lỗi, lập trình an toàn cho các lỗi trên