浙江大学

本科实验报告

课程名称: 网络安全原理与实践

姓　　名:

学　　院: 计算机科学与技术学院

系: 计算机科学与技术系

专　　业:

学　　号:

指导教师: 卜凯

2021 年 4 月 1 日

浙江大学实验报告

课程名称：网络安全原理与实践

实验名称：Lab 02

1. Requirements

Find hidden flags.

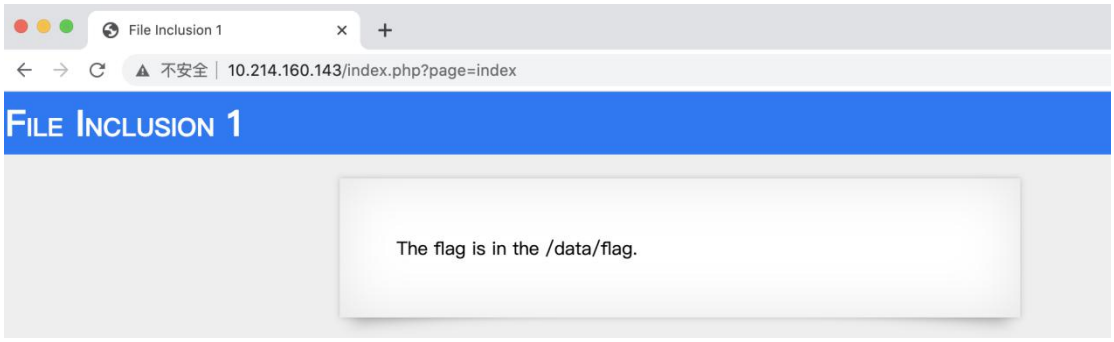
2. Environments

Browser: Chrome version 89.0.4389.90
System: MacOS Catalina, Kali Linux 2021.1

3. Processes

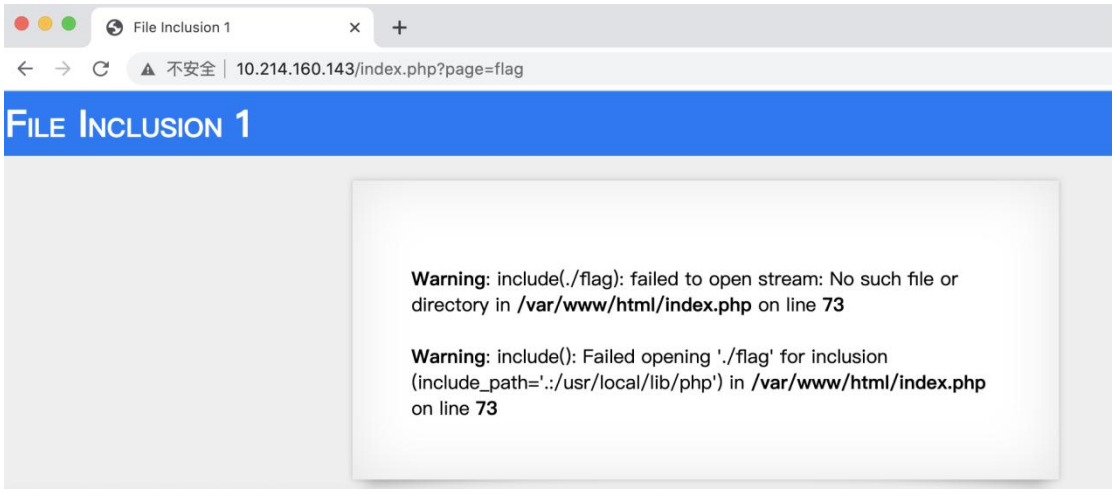
3.1 File Inclusion 1

The problem require us to get the flag which located at `/data/flag`.



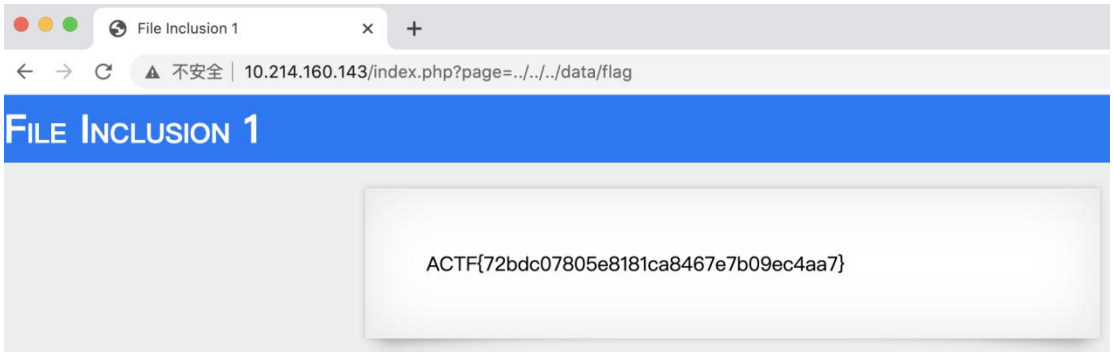
From the URL `http://10.214.160.143/index.php?page=index` we can guess the mechanism behind is PHP file inclusion.

Firstly, attempt with `page=flag`, and warnings occur.



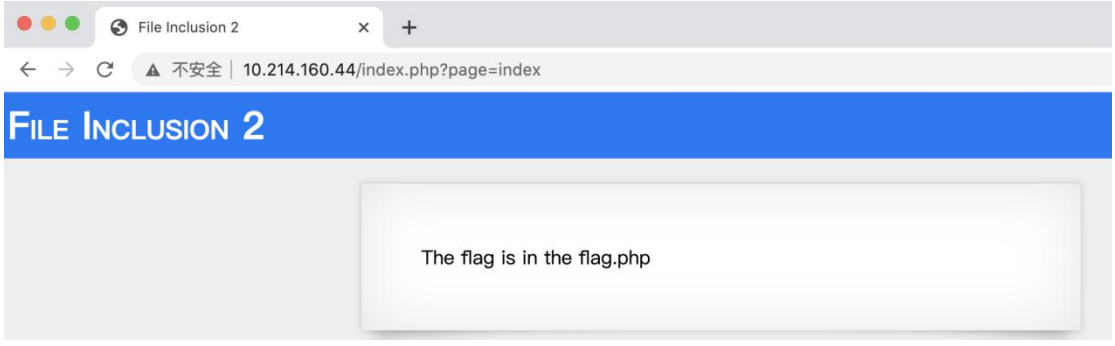
From the warning information we can get that the PHP source code is located at `/var/www/html/index.php` and then we can access the flag with relative address `../../../data/flag`.

Construct payload `page=../../../data/flag` and we can get the flag.



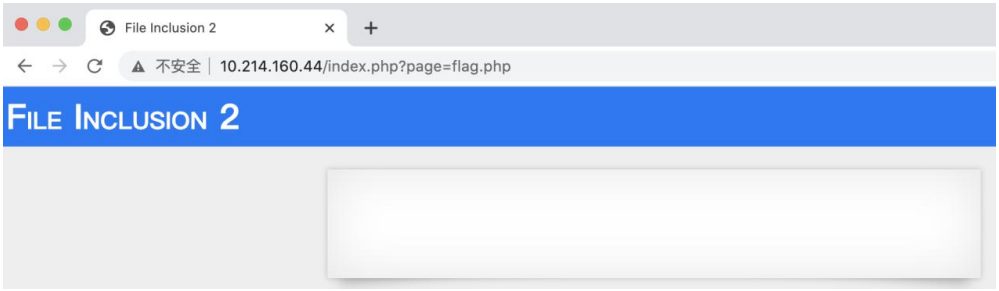
3.2 File Inclusion 2

In this problem, the flag is in a PHP file.



When we directly use `page=flag.php`, the content of flag.php won't be shown on

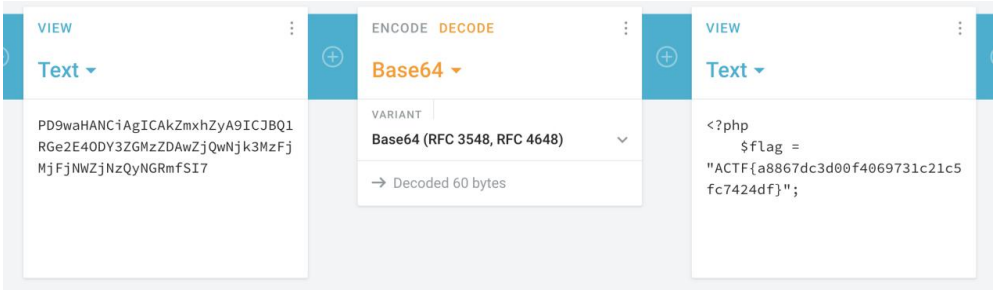
the page, since a PHP file would be run rather than simply displayed.



But we can use PHP pseudo protocol to get the content of flag.php as another form, like base64 encoded code. Construct the payload:

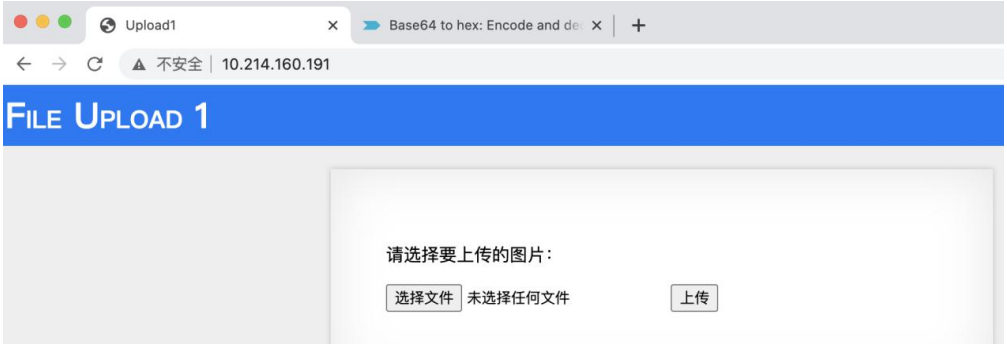
`page=php://filter/read=convert.base64-encode/resource=flag.php`

Decode the base64 code and we can get the flag.

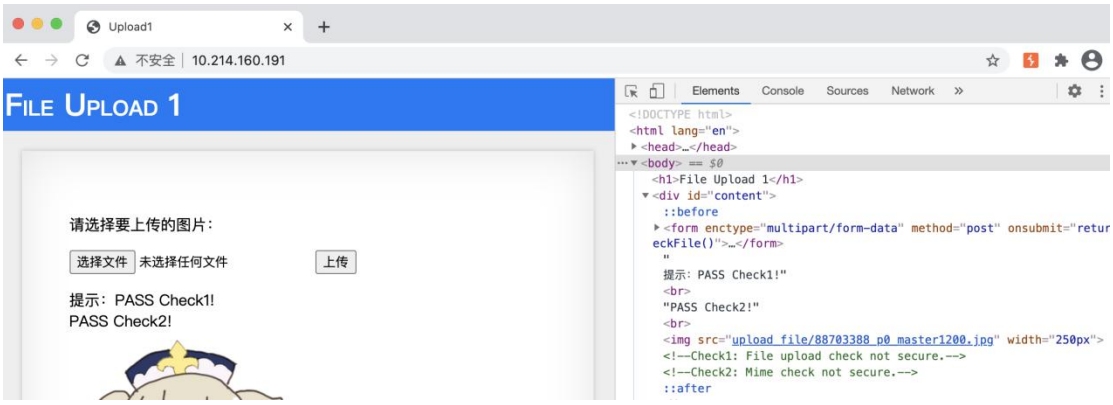


3.3 Upload 1

The flag is located at `/data/flag`, and the page allowed us to upload an image.



Upload a real image first.



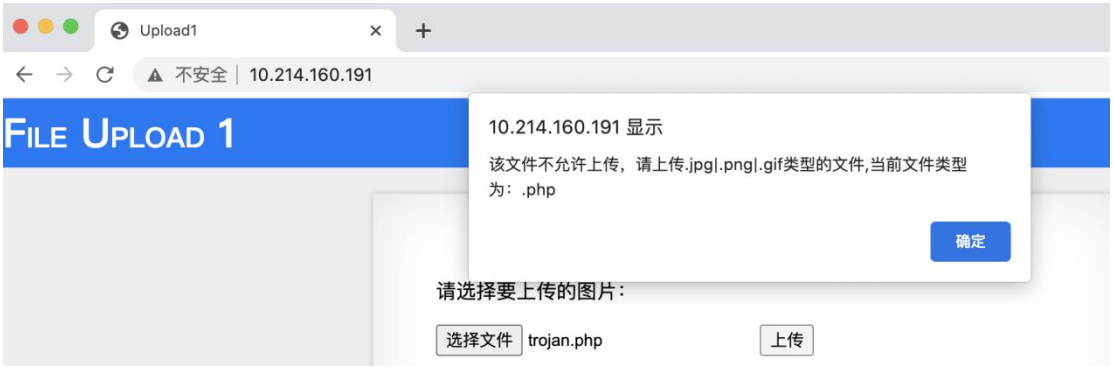
We can see the image was saved under a directory named upload_file.

We can try uploading a PHP file and access it externally to visit the flag.

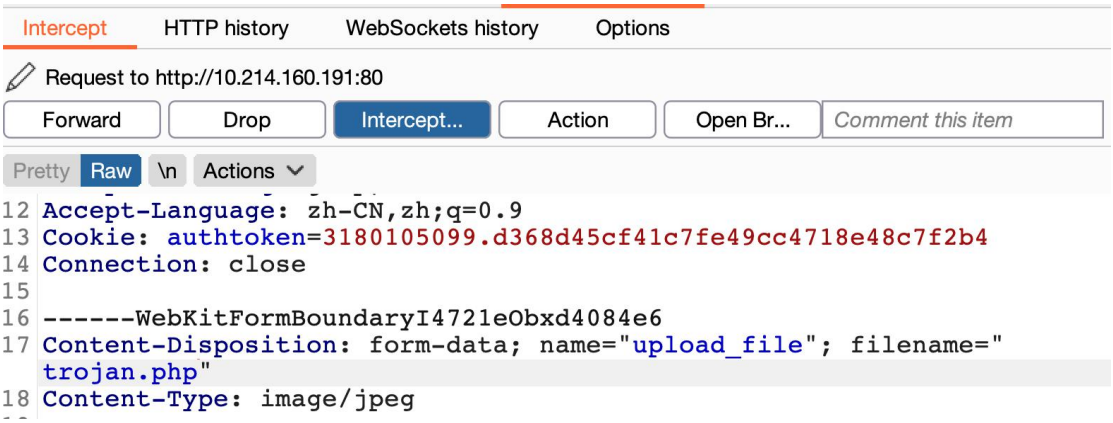
Construct a PHP Trojan:

```
<?php show_source("/data/flag");?>
```

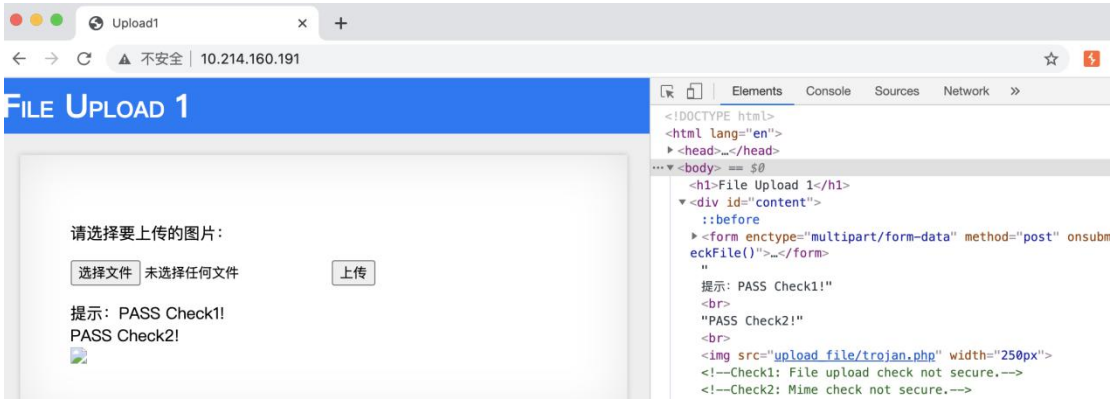
Name the PHP file as trojan.php and upload it.



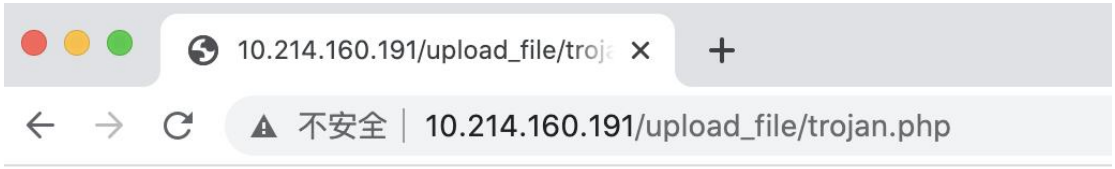
Since the warning tells us only jpg, png, or gif files can be uploaded, we can rename the PHP file as trojan.jpg and intercept the post packet after the front check, and modify the file name suffix back to php.



Forward the packet and we can see the PHP file was uploaded successfully.



Then access it from the URL bar and we can see the flag.

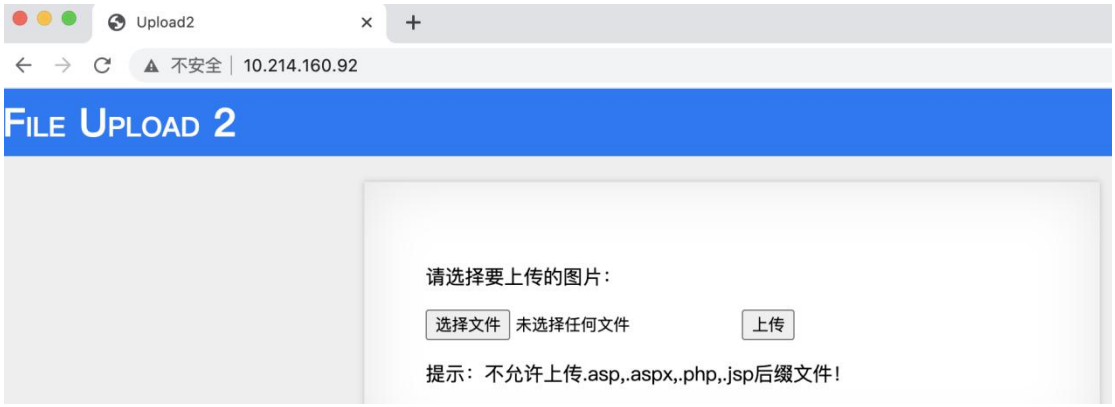



```
ACTF{a2cdd86a458242d42a17c2bf4feff069}
```

3.4 Upload 2

Firstly try the same way as in Upload 1.

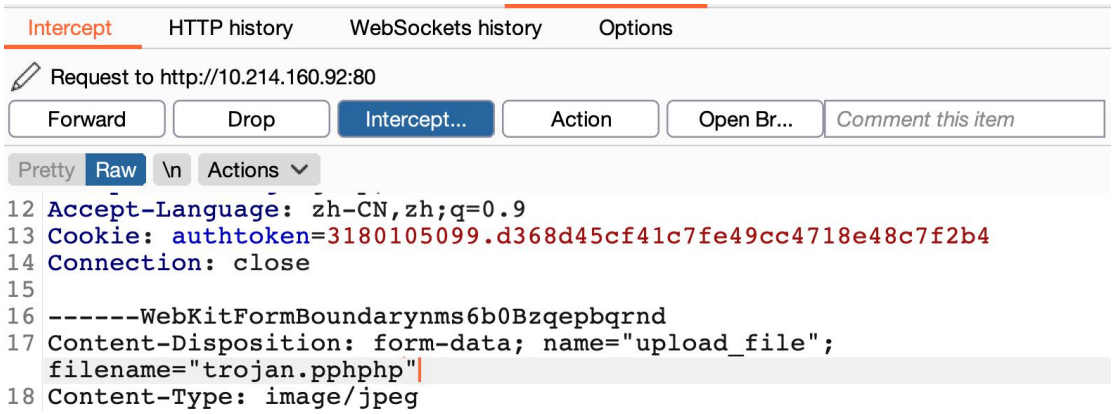
But it seems to not work this time.



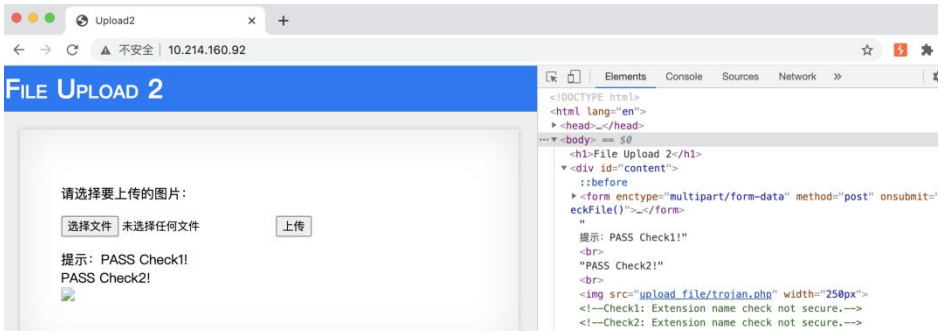
Since this failed, we can guess the check is on the server rather than the front end.

But we can still do some forgery to bypass this check, like rename the suffix to "pphphp" rather than "php".

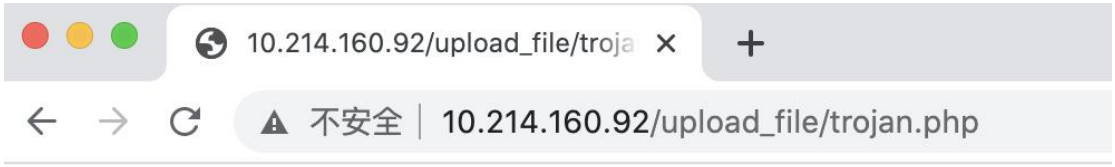
Intercept the post packet and modify the "filename" argument.



Forward the packet and we can see the check was bypassed successfully.



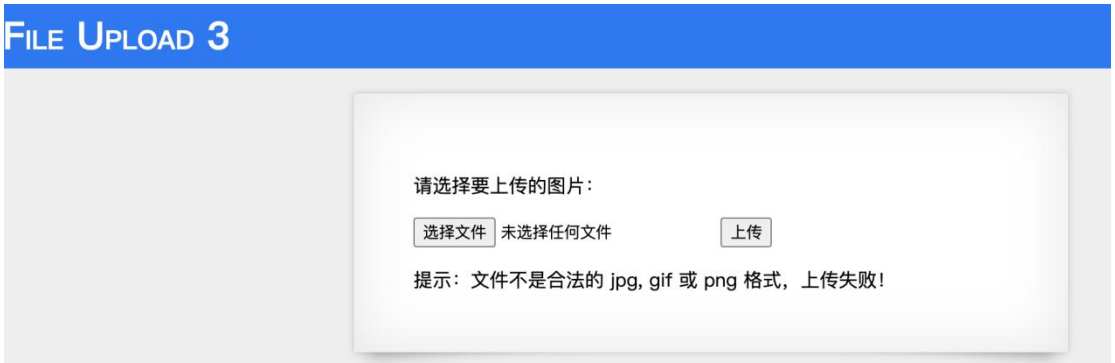
Visit the PHP file from the URL bar and we can get the flag.




```
ACTF{f671e83ddce45f25ac858d57b1b1b222}
```

3.5 Upload 3

Firstly try the same way as in Upload 1.



We see the server would check the file format, so we can try attaching the PHP Trojan to a real image.

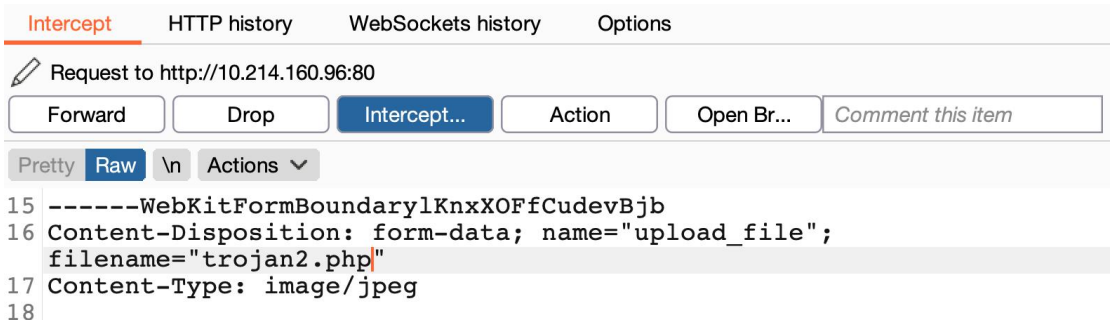
Create a empty image using Python.

```
from PIL import Image

image = Image.new('RGB', (100, 100))

image.save('img.jpg', 'jpeg')
```

And attach the PHP file to it

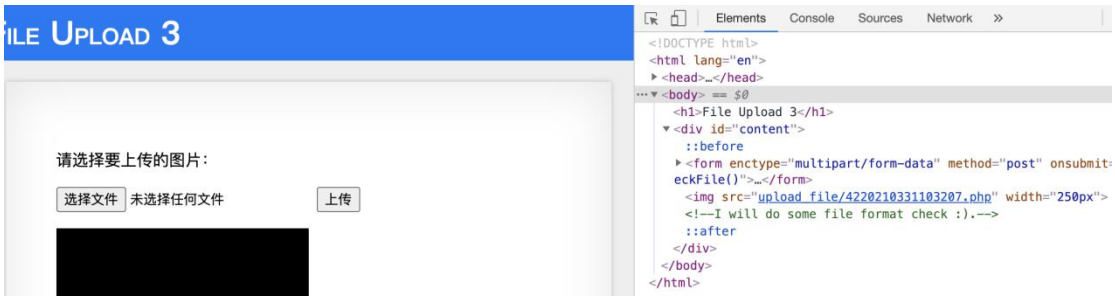
```
Bash> cat img.jpg trojan.php > trojan2.jpg
```

Upload the image and intercept the post packet.



Modify the file name suffix before forward it.

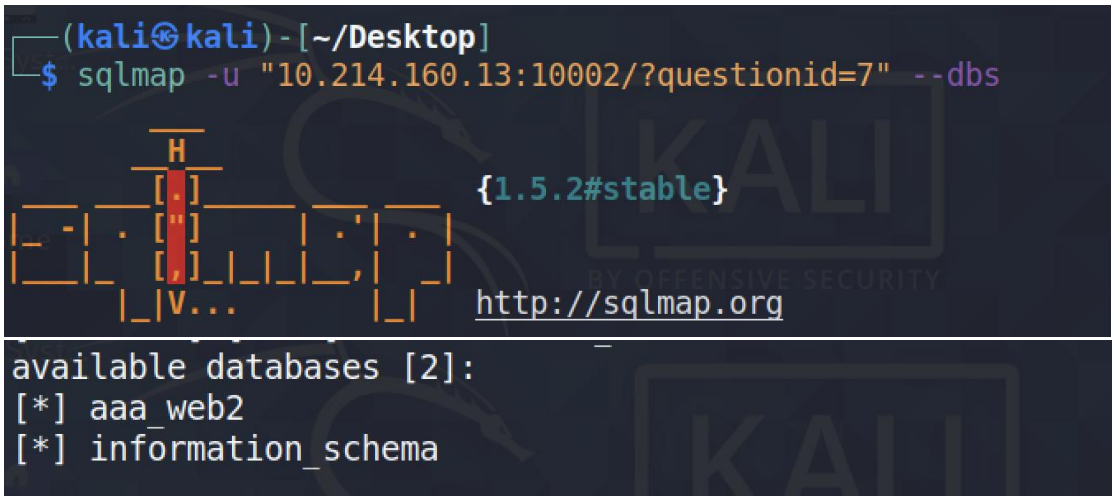
And we can see the check was bypassed successfully.



Visit the PHP file through the URL bar and we can get the flag.

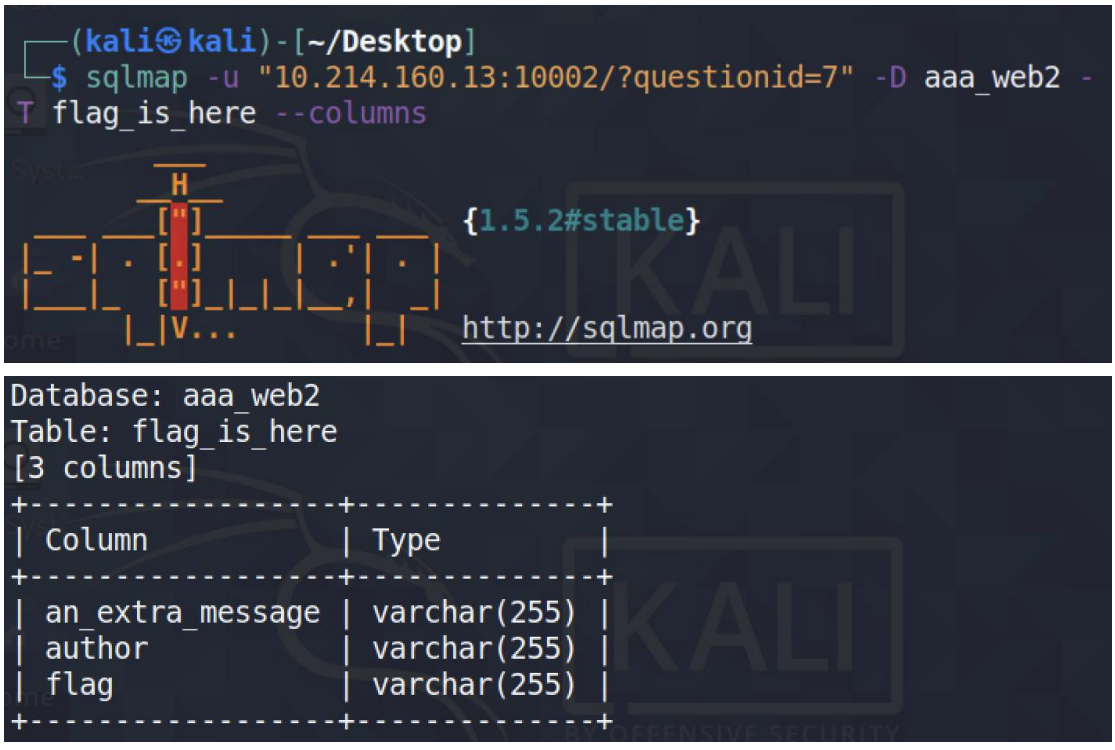
3.6 SQL Injection

From the hint we know that sqlmap is suggested.

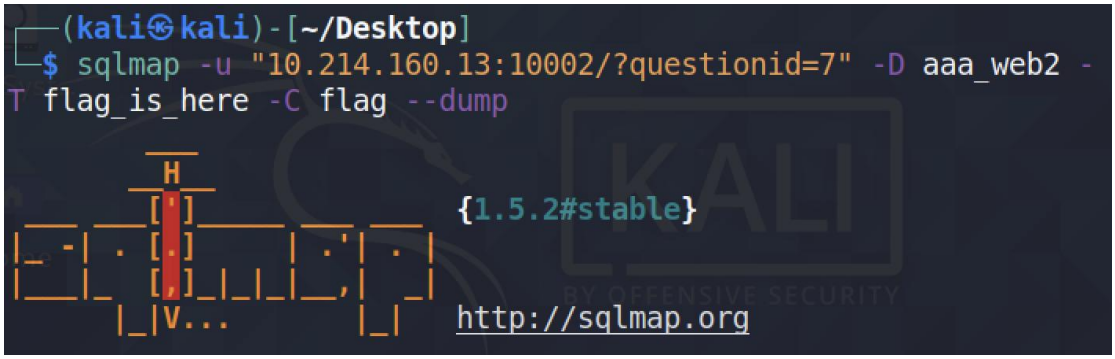


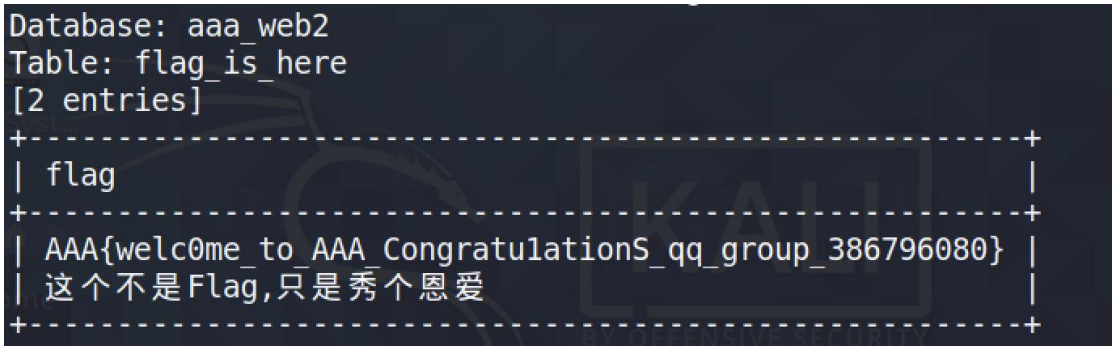
We can get the database name is “aaa_web2”.

Since the hints tell us the flag is stored in a table named “flag_is_here”, we can directly burst it.



Query the “flag” column and we can get the flag.





4. Experience and Thinking

All problems are easy but interesting and I have got a lot from them.

The two file inclusion problems introduce PHP inclusion and PHP pseudo protocols. And the three upload problems are mainly about file upload check. The last problem presents us the harm of SQL injection and a tool “sqlmap” to exploit it.

After finishing all problems, I have got more about Network Security. Hope to do better in the following study.