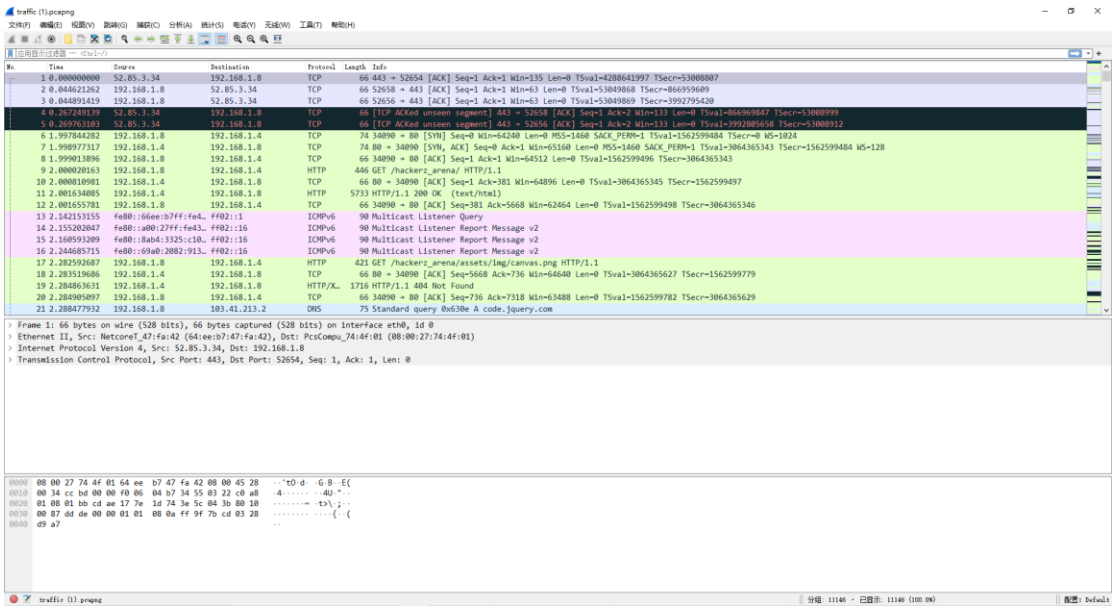
Compromised CTF Platform

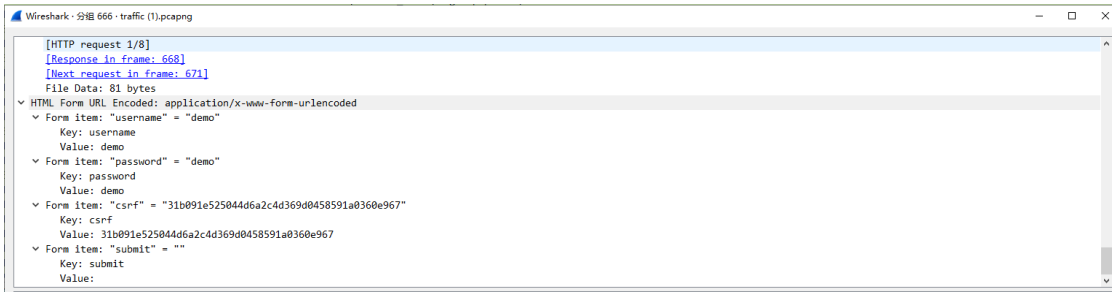
该题所提供的数据文件涉及非常多的其他题目，因此将这些题目放在一起解决。

该题的提示为：
I created a CTF platform of my own & hosted on a server. It seems like someone got access to my site. I have captured the traffic. Help me find out who he is.
因此我们需要找出登录该网站用户的用户名和密码，Flag 提交格式为：
KCTF{username_password}

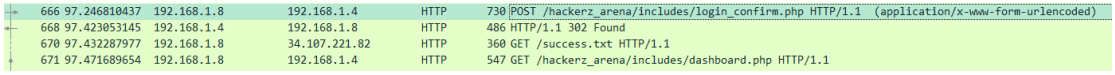
首先使用 Wireshark 打开题目提供的 traffic.prcpng 文件，观察服务器的通信情况。打开后如下：



由于用户登录为 POST 请求，因此，我们只需检索所有的 POST 请求，即可获取所有的登录信息。



但是只有序号为666的POST请求会跳转到dashboard.php文件而其他的均跳转到login.php，因此只有序号为 666 的 POST 登录成功。所以用户名密码应该均为 demo。

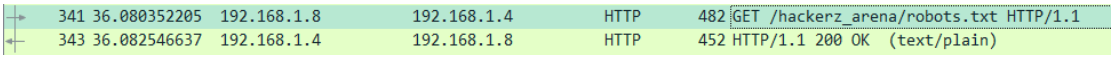


所以 flag 为 KCTF{demo_demo}

Robots.txt

该题的提示为：
What is the file path in robots.txt?
因此我们需要找出 robots 协议中的地址，flag 提交格式为：
KCTF{ /path/path}

首先在 traffic.prcpng 文件中搜索对 robots.txt 的请求：



可以看到，第 341 号为向 robots.txt 的请求，因此第 343 号即为服务器的返回数据，其中应包含 robots.txt 的内容。

返回数据如下：

```
User-agent: *\r\n
\r\n
Disallow : /includes/users.php\r\n
```

所以 robots.txt 中的文件路径为“/includes/users.php”。
所以 flag 为：
KCTF{/includes/users.php}

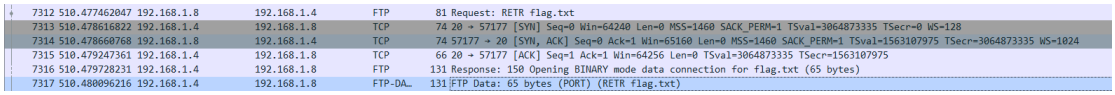
FTP Flag

该题的提示为：

What is the ftp flag?

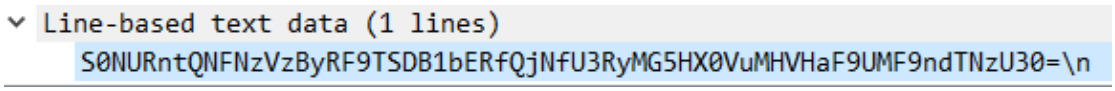
因此我们需要找出隐藏着的 flag。

直接在 traffic.prcpng 文件中搜索 flag.txt，得到如下结果：



可以看到序号为 7312 发起了对 flag.txt 的请求，而 7317 返回了请求。

因此，打开 7317 的数据包：



显然，返回的数据为 base64 加密的密文。

解密可得明文如下：

KCTF{P4SsW0rD_SH0ulD_B3_Str0nG_En0uGh_T0_gu3sS}

PHP Version

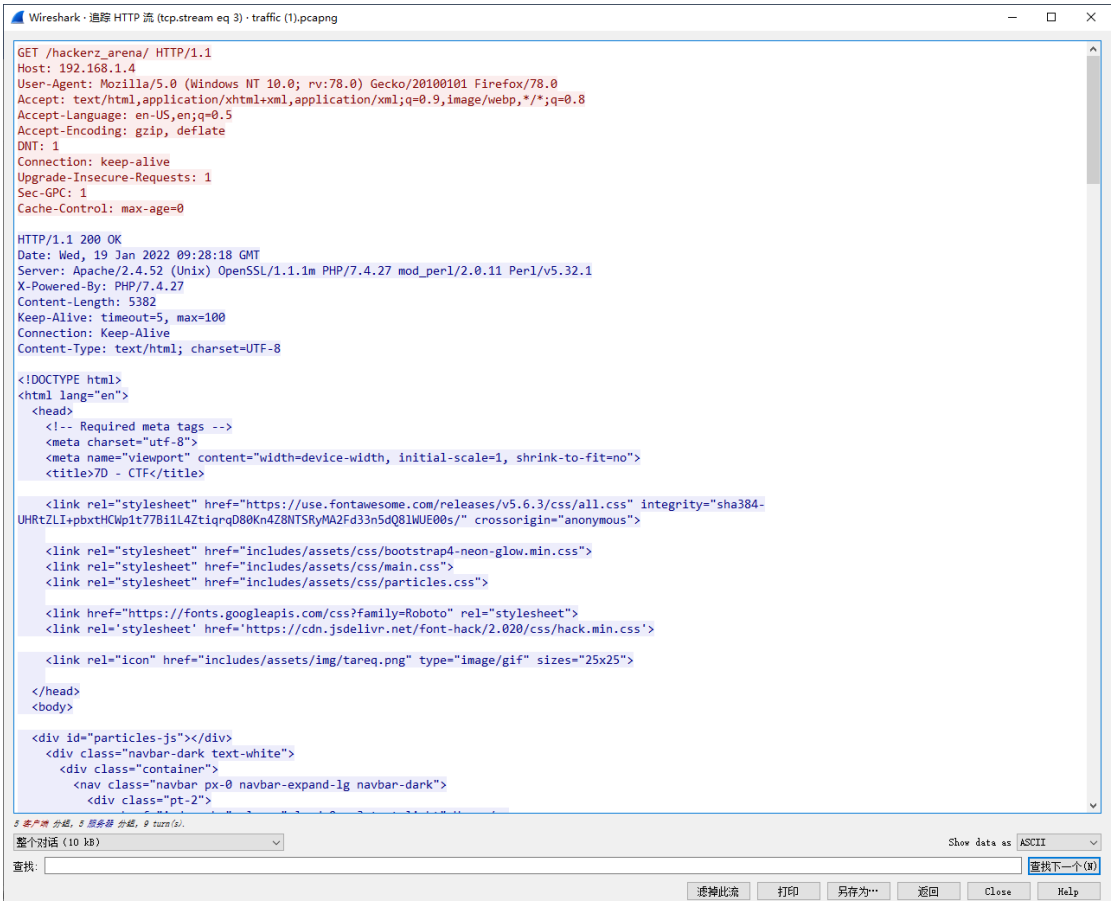
该题的提示为：

What version of php the server is using?

因此我们需要找出使用的 php 版本，flag 提交格式为：

KCTF{php/version}

任意打开一个请求的 http 流：



可以看到，在 http 数据中，server 一栏标明了 PHP 的版本：

```
Server: Apache/2.4.52 (Unix) OpenSSL/1.1.1m PHP/7.4.27 mod_perl/2.0.11 Perl/v5.32.1
```

为 7.4.27 版。

因此 Flag 为：

KCTF{PHP/7.4.27}

KCTF

该题的提示为：
It's all about kctf.
没啥作用，我们可以尝试直接搜索 kctf。

搜索结果如下：

```
7970 708.525601004 192.168.1.8    192.168.1.4    HTTP    489 GET /hackerz_arena/includes/kctf.jpg HTTP/1.1
```

可以看到，kctf 为.jpg 格式的图片。
因此使用 Wireshark 的导出功能，直接导出 kctf.jpg。

导出图片如下：

显然 Flag 为：

KCTF{Ev3rY_UsEr_1nPuT_SH0uLD_B3_S4niT1z3D}

Vuln\ Attacker\Database Flag

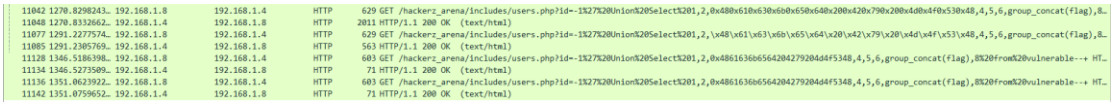
三道题需要联合食用。

Vuln 的提示为：
What vulnerability did the attacker exploited?
需要找出黑客利用的漏洞。
flag 提交格式为：
KCTF{vulnerability_name}

Attacker 的提示为：
What is the attacker name?
需要找出黑客的姓名。
flag 提交格式为：
KCTF{ATTACKERNAME}

Database Flag 的提示为：
What is the retrived flag from database?
需要找到隐藏在数据库中的 flag。

简要浏览所有的 http 请求，我们发现在最后存在很多异常的请求：



打开这些请求观察，发现这些请求使用了 sql 语句。

```
[GET /hackerz_arena/includes/users.php?id=-1%27%20Union%20Select%201,2,0x4861636b6564204279204d4f5348,4,5,6,group_concat(flag),8%20from%20vulnerable--+ HTTP/1.1\r\n]
```

很明显，黑客使用了 sql injection(sql 注入)黑入了网站。
因此 Vuln 的 flag 为：
KCTF{sql_injection}

找到了黑客的请求，那么打开黑客最后黑入的请求的响应数据包，编号为 11142。
可以在返回的 html 数据中发现这样一段：

```
<h1>HACKED BY MOSH 4</h1>\n
<p class="btn btn-rounded btn-outline-warning">Country : 6</p>\n
<p class="btn btn-rounded btn-outline-warning">Joining Date : S0NURntTcUw=,XzFOajNDN2kwbn0=</p>\n
```

很明显，黑客名字为 MOSH
因此 Attacker 的 flag 为：
KCTF{MOSH}

而且很明显，数据库中的数据为

```
Joining Date : S0NURntTcUw=,XzFOajNDN2kwbn0=
```

显然为两段 base64 密文，分别解密为：KCTF{SqL 和_1Nj3C7i0n}
因此 Database Flag 的 Flag 为其拼接结果：
KCTF{SqL_1Nj3C7i0n}

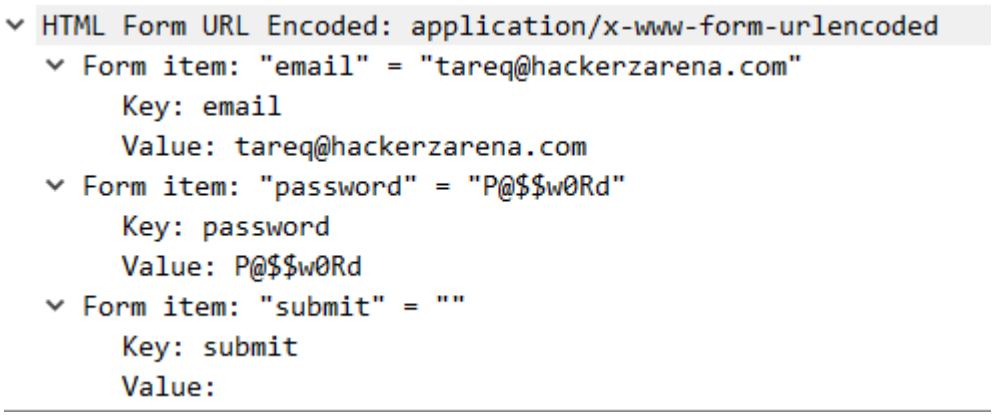
Admin Arena

该题的提示为：
What is the Admin Arena email id & password?
Flag 的格式为：
KCTF{email_password}

显然，email 登录使用的是 POST 请求，那么就寻找所有 POST 请求，查看是否存在 email 以及密码。

经过浏览，我们找到了一个 POST 请求：

在路径中发现了题目中的文字 admin_arena。
打开该数据包，可以找到 email 和 password：



因此，email 为：tareq@hackerzarena.com
password 为：P@$$w0Rd

所以，flag 为：
KCTF{tareq@hackerzarena.com_P@$$w0Rd}

至此，关于 traffic.prcpng 文件的题解决完成，剩余题目使用其他文件。

How's the Shark?

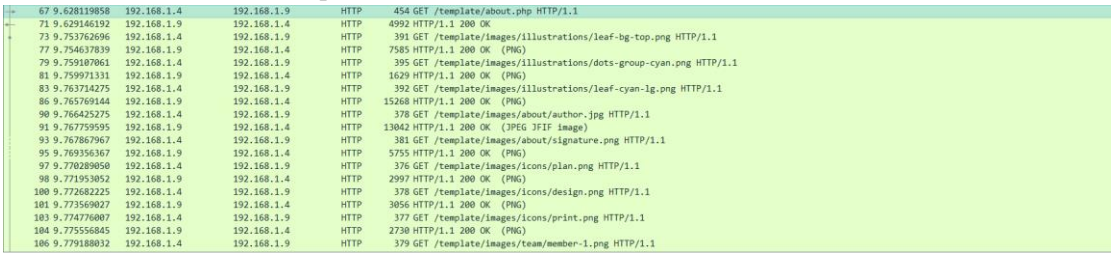
该题的提示为：

Find the flag from the following.

因此需要在提供的 data.pcapng 文件中找到 flag。

在文件中搜索 shark，但是没有找到。

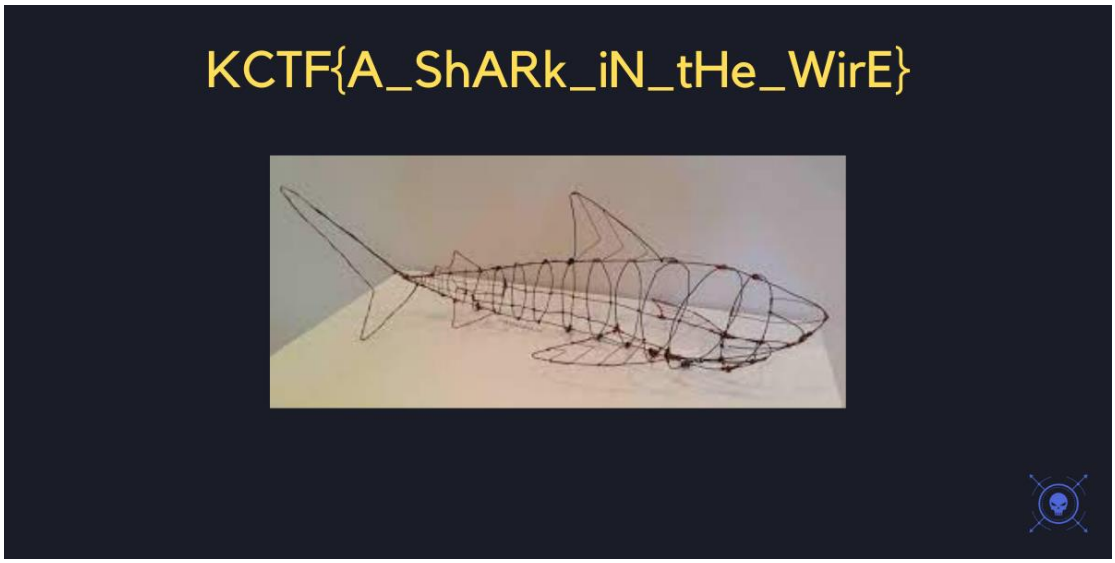
换个思路，观察所有的 http 数据包，结果如下：



67	9.628119858	192.168.1.4	192.168.1.9	HTTP	454	GET /template/about.php HTTP/1.1
71	9.629146192	192.168.1.9	192.168.1.4	HTTP	4992	HTTP/1.1 200 OK
73	9.753762696	192.168.1.4	192.168.1.9	HTTP	391	GET /template/images/illustrations/leaf-bg-top.png HTTP/1.1
77	9.754637839	192.168.1.9	192.168.1.4	HTTP	7585	HTTP/1.1 200 OK (PNG)
79	9.759107061	192.168.1.4	192.168.1.9	HTTP	395	GET /template/images/illustrations/dots-group-cyan.png HTTP/1.1
81	9.759971131	192.168.1.9	192.168.1.4	HTTP	1629	HTTP/1.1 200 OK (PNG)
83	9.763714275	192.168.1.4	192.168.1.9	HTTP	392	GET /template/images/illustrations/leaf-cyan-lg.png HTTP/1.1
86	9.765769144	192.168.1.9	192.168.1.4	HTTP	15268	HTTP/1.1 200 OK (PNG)
90	9.766425275	192.168.1.4	192.168.1.9	HTTP	378	GET /template/images/about/author.jpg HTTP/1.1
91	9.767759595	192.168.1.9	192.168.1.4	HTTP	13042	HTTP/1.1 200 OK (JPEG JFIF image)
93	9.767867967	192.168.1.4	192.168.1.9	HTTP	381	GET /template/images/about/signature.png HTTP/1.1
95	9.769356367	192.168.1.9	192.168.1.4	HTTP	5755	HTTP/1.1 200 OK (PNG)
97	9.770289050	192.168.1.4	192.168.1.9	HTTP	376	GET /template/images/icons/plan.png HTTP/1.1
98	9.771953052	192.168.1.9	192.168.1.4	HTTP	2997	HTTP/1.1 200 OK (PNG)
100	9.772682225	192.168.1.4	192.168.1.9	HTTP	378	GET /template/images/icons/design.png HTTP/1.1
101	9.773569027	192.168.1.9	192.168.1.4	HTTP	3056	HTTP/1.1 200 OK (PNG)
103	9.774776007	192.168.1.4	192.168.1.9	HTTP	377	GET /template/images/icons/print.png HTTP/1.1
104	9.775556845	192.168.1.9	192.168.1.4	HTTP	2730	HTTP/1.1 200 OK (PNG)
106	9.779188032	192.168.1.4	192.168.1.9	HTTP	379	GET /template/images/team/member-1.png HTTP/1.1

可以看到，基本都是在访问图片。因此我们使用 Wireshark 导出访问的所有图片。

浏览所有图片后，有一张名为 something%20something.png 的图片提供了 Flag：



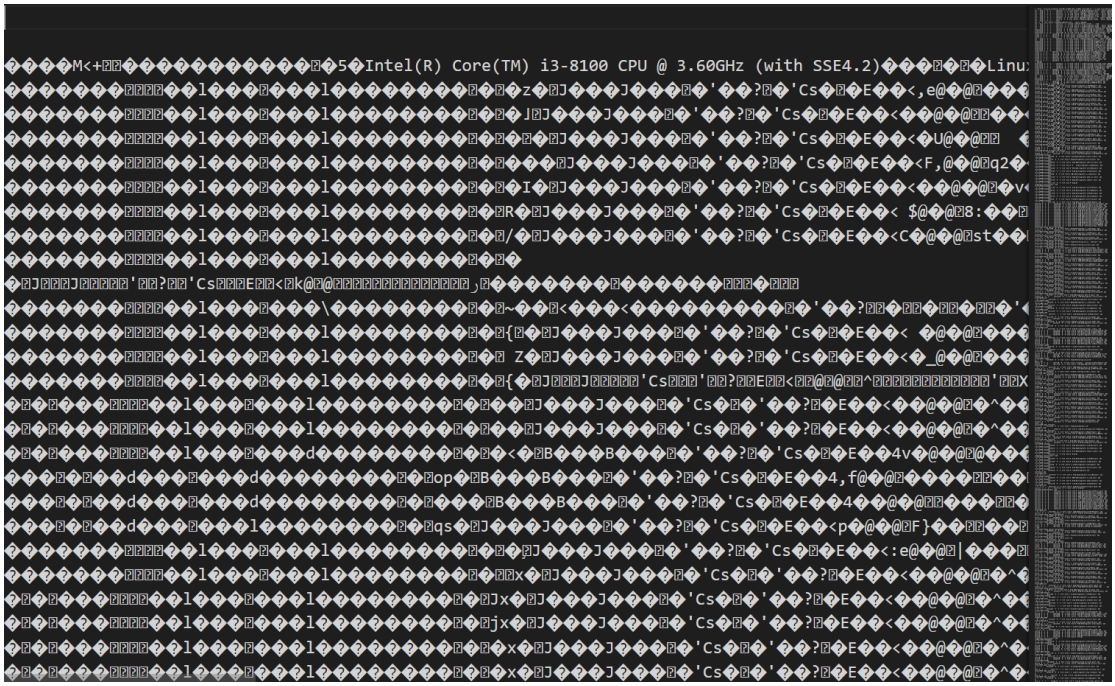
因此 Flag 为：

KCTF{A_ShARk_iN_tHe_WirE}

Find the Flag

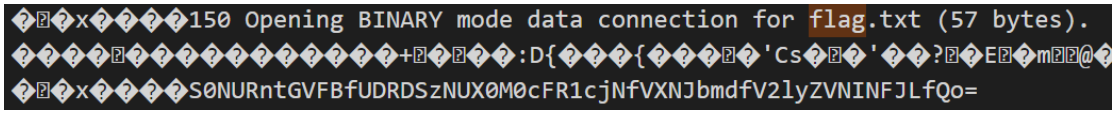
该题的提示为：
Find the flag from the following file.
因此需要在提供的 file 文件中找到 flag。

直接在 vscode 中强行打开 file 文件，如下所示：



存在很多乱码，但是也有可以阅读的部分，比如：首行的 cpu 型号。

直接搜索 flag 进行尝试，有 7 个搜索结果，其中某个搜索结果为：



很明显，该语句之后的就是 flag.txt 的内容，而紧接着该语句的就是一个 base64 加密的密文字符串：

```
S0NURntGVFBfUDRDSzNUX0M0cFR1cjNfVXNJbmdfV2lyZVNINFJLfQo=
```

进行解密得 flag：
KCTF{FTP_P4CK3T_C4pTur3_UsIng_WireSH4RK}