西普实验吧部分逆向题writeup（一）

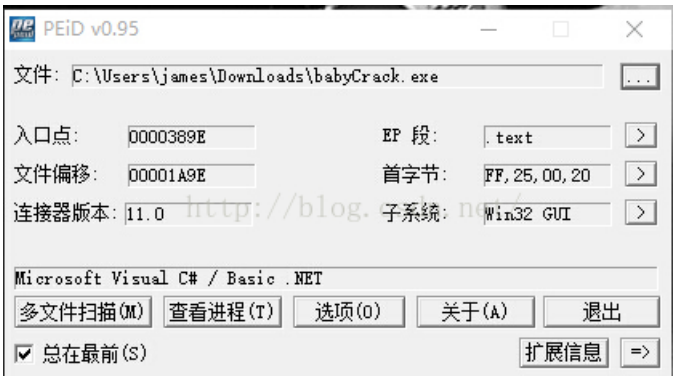
原创

caterpillarous 于 2015-12-19 20:23:17 发布 8480 收藏 2
分类专栏： 逆向之路 文章标签： 逆向 二进制 CTF

本博客已经弃用，我的新博客地址：*http://jujuba.me/*

1. baby Crack

本题难度就像它的名字一样，简单得很。下载程序，先用PEID查一下



发现没壳，就是C#写的程序

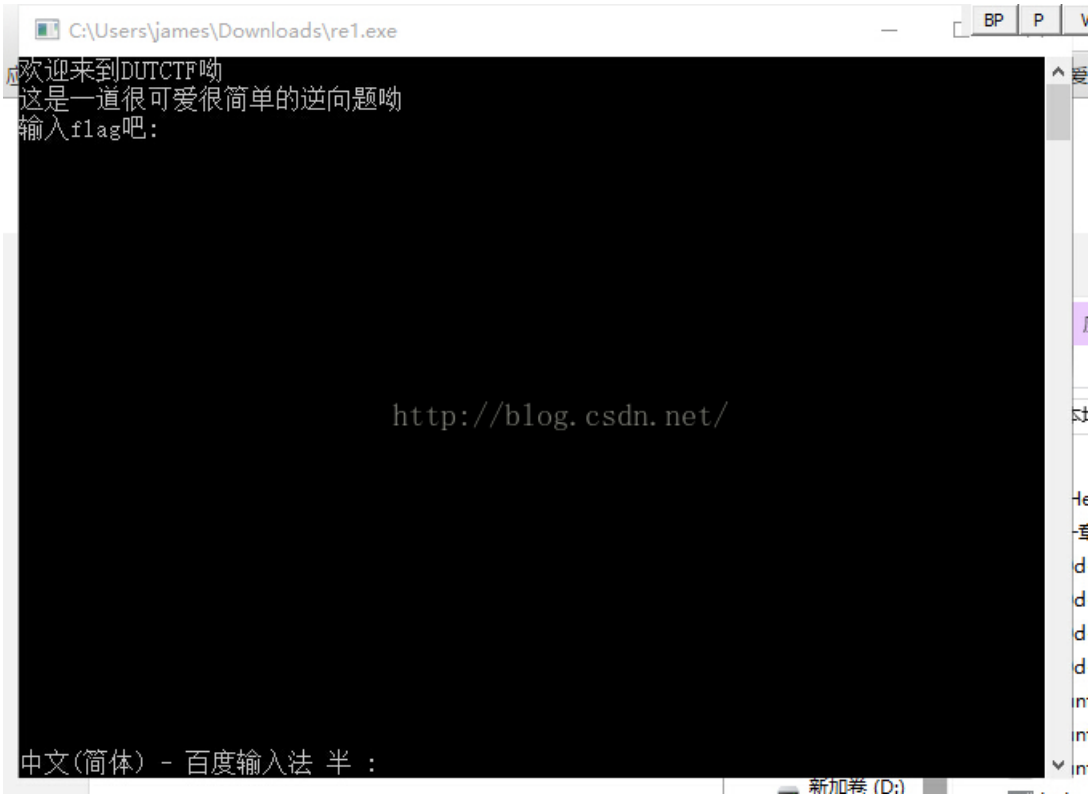
再用IDA打开，往下拉几行即可看到KEY

```
                int32 V4,
                bool V5)
  nop
  ldc.i4.0
  stloc.0
  ldarg.0
  ldfld     class [System.Windows.Forms]System.Windows.Forms.TextBox Easy_CM.Form1::textBox1
  callvirt  instance string [System.Windows.Forms]System.Windows.Forms.Control::get_Text()
  stsfld    string Easy_CM.Config::user
  ldsfld    string Easy_CM.Config::user
  stloc.1
  ldstr     aHctfBaby_ctsvl          // "hctf{bABy_CtsvlmE_!}"
  stloc.2
  ldloc.2
  ldloc.1
  callvirt  instance int32 [mscorlib]System.String::CompareTo(string)
  stloc.s   4
  ldloc.s   4
  ldc.i4.0
  ceq
```

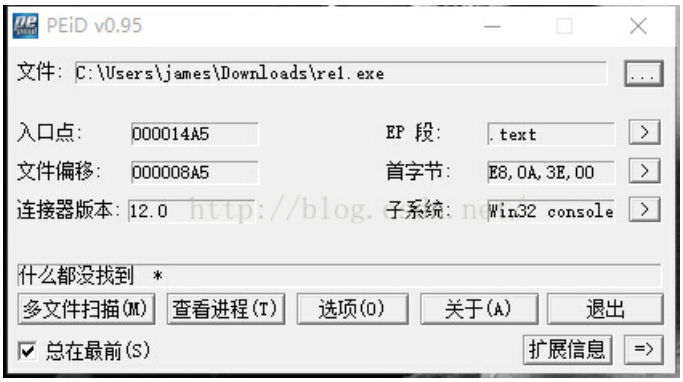
KEY： hctf{bABy_CtsvlmE_!}

2. 你会吗

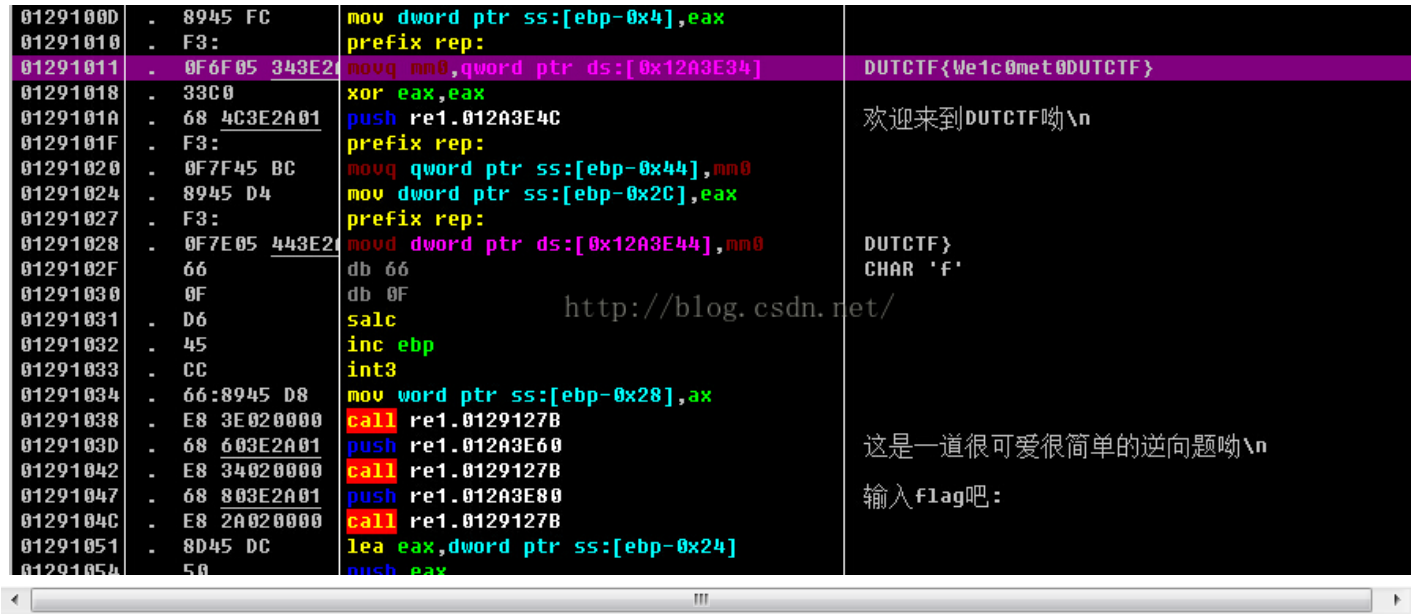
下载完先打开看下：



PEID查一下：



别急，用OD打开看看，搜索一下字符串：



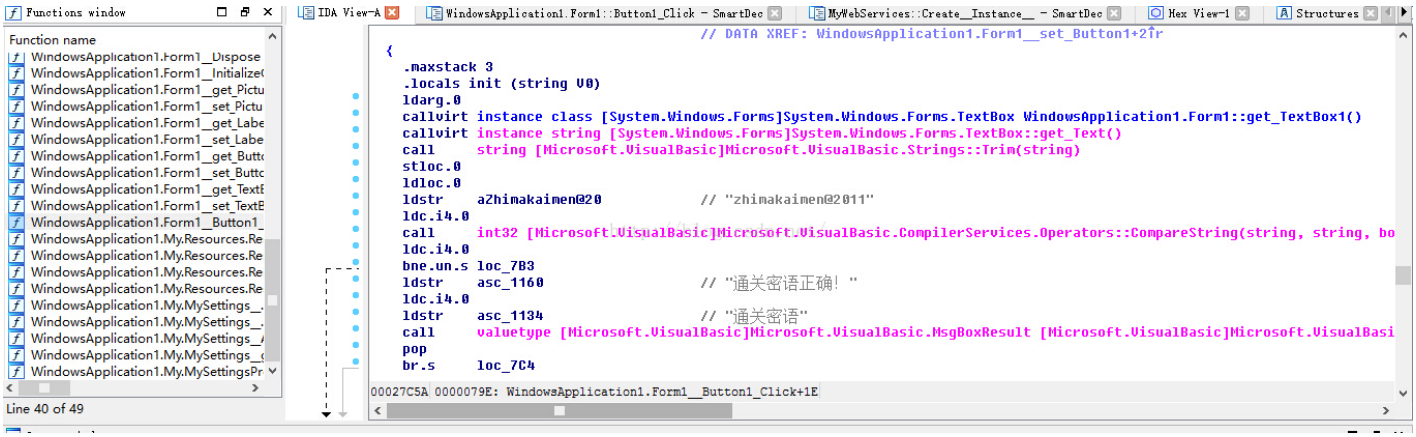
居然直接就看到了KEY....DUTCTF{We1c0met0DUTCTF}

3. 阿拉丁神灯

PEID 查了一下 又是C#的 那就直接用IDA打开了

这次往下找啊找感觉找不到，直接到左边的函数列表里看看有没有关键的

找到一个叫WindowsApplication1.Form1__Button1_Click 的函数，点进去，又看到KEY了...

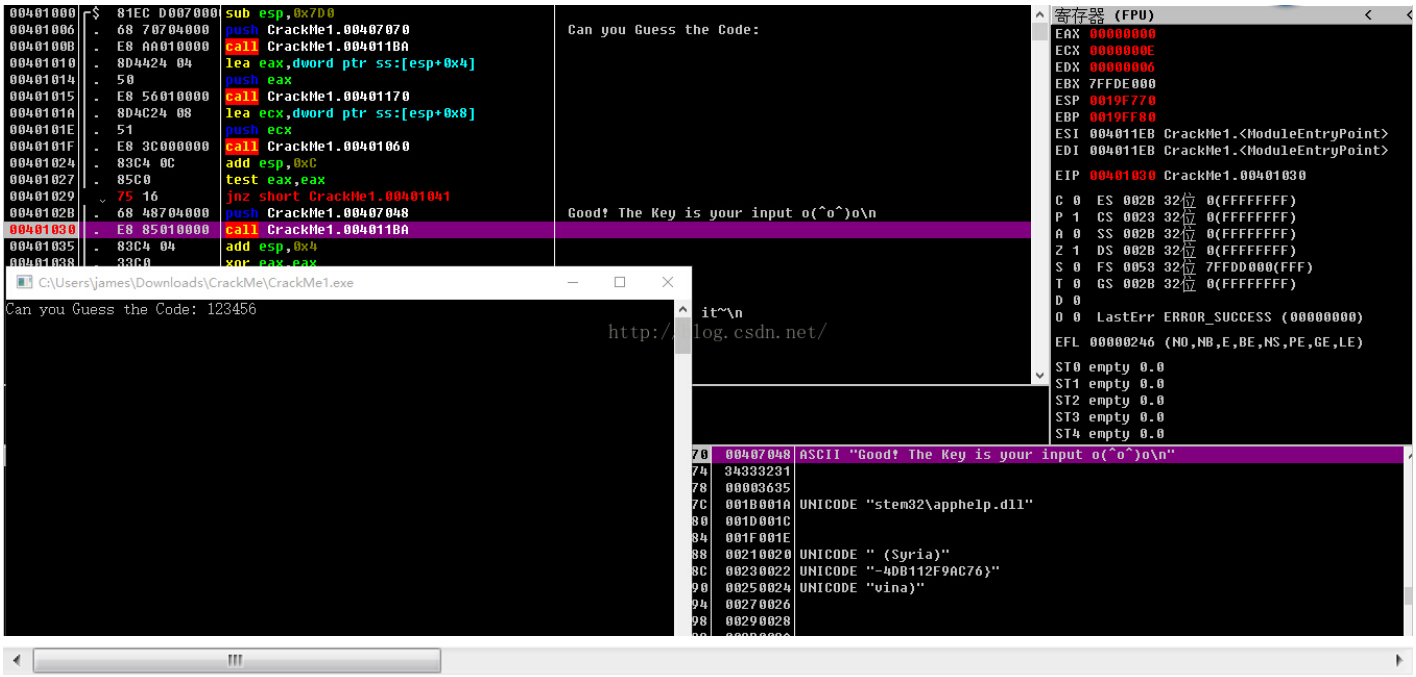


结果又看到了：zhimakaimen@2011

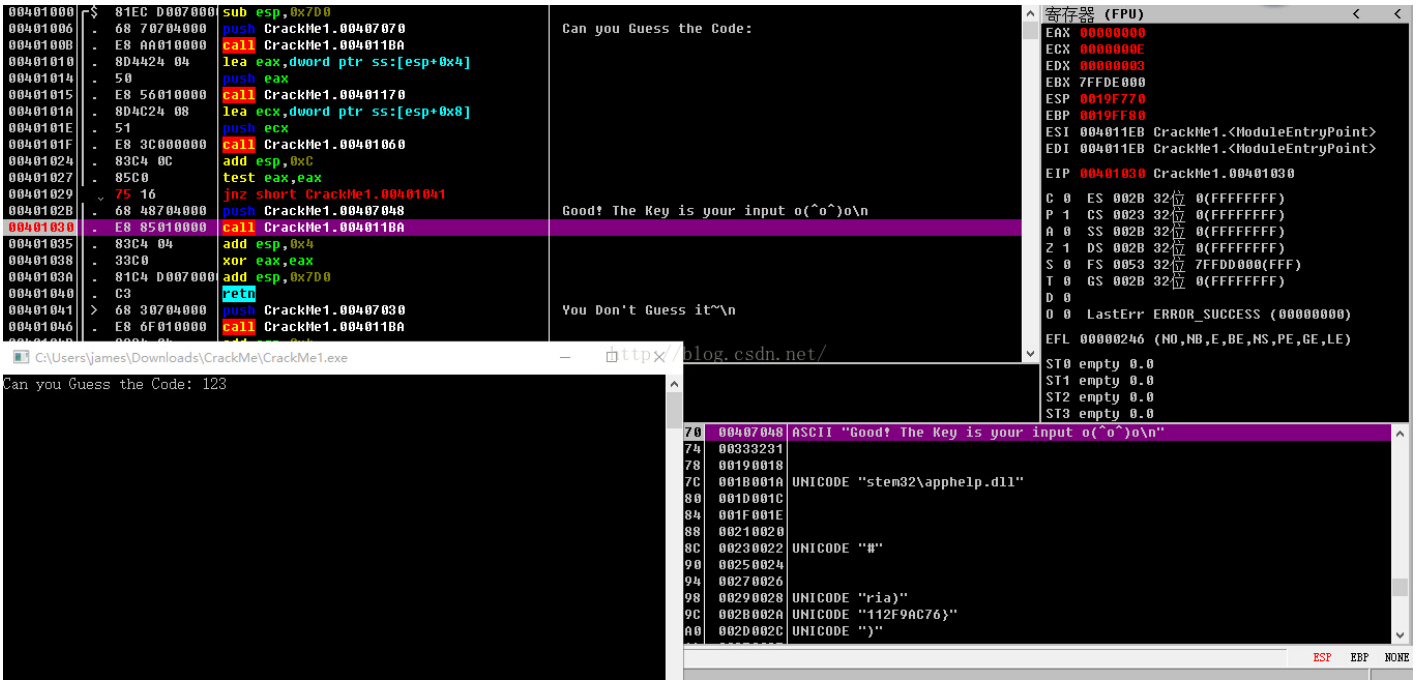
去网站输入之后得到KEY 小明向灯神许愿道~ 灯神啊~ 给我过关的Key吧~ 灯神说道\KEY:UnPack&Crack2011!!

4. 证明自己吧

查壳——没壳——C++编写成的——放入OD调试——搜索字符串——发现成功的字符串附近有跳转函数，改之–



看到右下角出现了类似KEY的东西，别急，换个数字试试



右下角的KEY类似物变化了，由此可知我们需要的KEY应该是临时算出来的，但是这个又不是注册机。我技术不

打开IDA，先F5反编译MAIN函数：

IDA View-A | Pseudocode-C | Pseudocode-B | Pseudocode-A | Hex Vi

```
1  int __cdecl main(int argc, const char **argv, const char **envp)
2  {
3    int result; // eax@2
4    int v4; // [sp+0h] [bp-7D0h]@1
5
6    sub_4011BA((int)aCanYouGuessThe, v4);
7    gets((char *)&v4);
8    if ( sub_401060((const char *)&v4) )
9    {
10     sub_4011BA((int)aGoodTheKeyIsYo, v4);
11     result = 0;
12   }
13   else
14   {
15     sub_4011BA((int)aYouDonTGuessIt, v4);
16     result = 0;
17   }
18   return result;
19 }
```


我们只看重点，看到gets函数了吧，我们输入的字符串从gets函数进入

看if-else分支的函数调用，if调用的是aGoodTheKey，而else调用的则是aYouDonTGuessIT，很明显，我们要走

去到sub_4011BA函数，F5反编译：

```
signed int __cdecl sub_401060(const char *a1)
{
  unsigned int v1; // edx@2
  unsigned int v2; // edx@4
  unsigned int v3; // edx@6
  int v5; // [sp+Ch] [bp-10h]@1
  int v6; // [sp+10h] [bp-Ch]@1
  int v7; // [sp+14h] [bp-8h]@1
  __int16 v8; // [sp+18h] [bp-4h]@1
  char v9; // [sp+1Ah] [bp-2h]@1

  v5 = dword_40708C;
  v6 = dword_407090;
  v8 = word_407098;
  v9 = byte_40709A;
  v7 = dword_407094;
  if ( strlen(a1) == strlen((const char *)&v5) )
  {
    v1 = 0;
    if ( strlen(a1) != 0 )
    {
      do
        a1[v1++] ^= 0x20u;
      while ( v1 < strlen(a1) );
    }
    v2 = 0;
    if ( strlen((const char *)&v5) != 0 )
    {
      do
        *((_BYTE *)&v5 + v2++) -= 5;
      while ( v2 < strlen((const char *)&v5) );
    }
    v3 = 0;
    if ( strlen((const char *)&v5) == 0 )
      return 1;
    while ( *((_BYTE *)&v5 + v3 + a1 - (const char *)&v5) == *((_BYTE *)&v5 + v3) )
    {
      ++v3;
      if ( v3 >= strlen((const char *)&v5) )
        return 1;
    }
  }
  return 0;
}
```

我们先看第一段算法：

```
if ( strlen(a1) == strlen((const char *)&v5) )
  {
    v1 = 0;
    if ( strlen(a1) != 0 )
    {
      do
        a1[v1++] ^= 0x20u;
      while ( v1 < strlen(a1) );
    }
```

看到我们传进来的字符串叫做a1，假如a1的长度等于v5的长度，则开始逐个元素做异或运算，和20异或
我们点进40708C地址看一看v5：

```
.data:00407030 aYouDonTGuessIt db 'You Don',27h,'t Guess it~',0Ah,0
.data:00407030                                         ; DATA XREF: _main:loc_401041↑o
.data:00407045                 align 4
.data:00407048 aGoodTheKeyIsYo db 'Good! The Key is your input o(^o^)o',0Ah,0
.data:00407048                                         ; DATA XREF: _main+2B↑o
.data:0040706D                 align 10h
.data:00407070 aCanYouGuessThe db 'Can you Guess the Code: ',0 ; DATA XREF: _main+6↑o
.data:00407089                 align 4
.data:0040708C dword_40708C    dd 48195768h            ; DATA XREF: sub_401060+3↑r
.data:00407090 dword_407090    dd 78586E50h            ; DATA XREF: sub_401060+8↑r
.data:00407094 dword_407094    dd 58196A54h            ; DATA XREF: sub_401060+39↑r
.data:00407098 word_407098     dw 65Eh                 ; DATA XREF: sub_401060+13↑r
.data:0040709A byte_40709A     db 0                    ; DATA XREF: sub_401060+1D↑r
.data:0040709B                 align 4
```

在此要注意的是v5存储的是地址，即需要比较的字符串的首地址，整个字符串应为 68571948 506e5878 546a1958 5e06H

再看第二段：

```
v2 = 0;
    if ( strlen((const char *)&v5) != 0 )
    {
      do
        *((_BYTE *)&v5 + v2++) -= 5;
      while ( v2 < strlen((const char *)&v5) );
    }
```

v5中的每个字节中存储的值-5

最后一段则是：

```
v3 = 0;
    if ( strlen((const char *)&v5) == 0 )
      return 1;
    while ( *((_BYTE *)&v5 + v3 + a1 - (const char *)&v5) == *((_BYTE *)&v5 + v3) )
   {
      ++v3;
     if ( v3 >= strlen((const char *)&v5) )
        return 1;
    }
```

可以看出这一段是在比较逐个a1和v5中的元素，while的循环条件前半部分有代码混淆，其实那一堆就相当于*（v3+a1）

现在我们的问题简化为：

我们输入了一段字符串，假设叫code

encode=code^20

当encode=str-5时条件就成立，此时的code就是我们想要的KEY

我们还需要知道的就是一个数连续对另一个数求异或两个，此时得到的数不变

所以我们只要把（str-5）^20就能得到key了

C++代码如下：

```
#include <iostream>
using namespace std;
int main(void)
{
 string code="\x68\x57\x19\x48\x50\x6e\x58\x78\x54\x6a\x19\x58\x5e\x06";
 for(int i=0;i<14;i++)
  code[i]=(code[i]-5)^0x20;
 cout<<code<<endl;
 return 0;
}
```

输出结果：

顺便贴个Python代码：

```
code=(0x68,0x57,0x19,0x48,0x50,0x6e,0x58,0x78,0x54,0x6a,0x19,0x58,0x5e,0x06)
for i in code:
    i=(i-5)^0x20
    print chr(i)
```

```
>>> ================================ RESTART ================================
>>>
C
r
4
c
k
I
s
S
o
E
4
s
y
!
>>>
```

KEY:Cr4ckIsSoE4sy!