ctf之逆向常见题型

原创

thesatrx 于 2022-01-23 21:38:49 发布
文章标签： 安全

开头

逆向的题型有很多很多种，那些没见过的，以后再说

写下这篇文章我顺便复习一下，好多东西都快忘了怎么做了，前面那些逆向基础知识，有什么用呢，就一个用，让你明白为什么要那么做，和数学一样，解题的方法绝对不止一种，做一道题看wp的时候，建议多看几份，学习一下各种不同的解法并动手得出flag，看会了不代表你真的动手就会了，计算机专业实践性非常强，一道题多解法，以后做题的话可以给自己更多选择，就可以打破一道题几个小时硬着头皮上了

逆向常见题型：

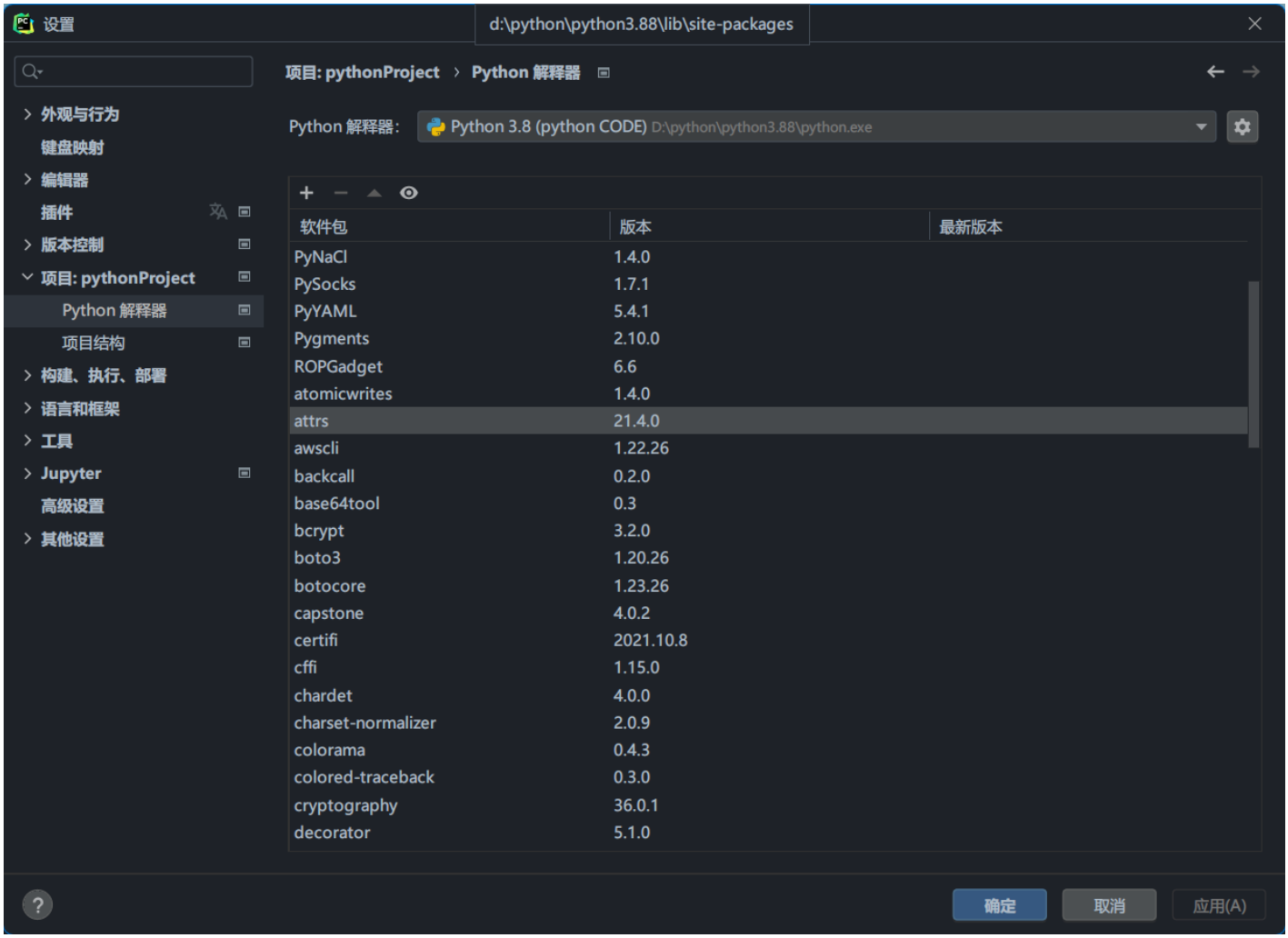
> base N(N代表正整数哈哈哈)base家族系列
>
> 用壳阻碍逆向：upx(压缩壳)，其他保护壳，压缩壳有想法的可以去了解一下，反正我觉得目前没用，比如另一个压缩壳（穿山甲壳），最难脱的虚拟机保护壳，其他还有一些壳，记不住了（穿山甲壳应该是压缩壳，记不清了，错了来说我，吾爱虚拟机有脱壳机，这壳好像开源了）（脱壳是非常爽的，建议合理脱壳，有的壳脱脱就行了，乱发要进局子）
>
> 经典的迷宫题，有点点杂项的味道
>
> 混淆类的题型： 花指令，利用脏字节混淆骗IDA，OD
>
> Z3，做一些算法，或者数学里面的运算（下面会详细说）
>
> tea家族：常见的一般tea,xxtea，xtea出现不多
>
> 高级语言打包阻碍逆向：我遇到过C#,java,py的反编译，建议去找好工具jd反编译java的class字节码文件，还有dnspy反编译C#,py的反编译我用的在线网站（我相信我那一堆工具里面绝对有）
>
> 只能动态调试的题目：因为你不动态调试，打断点，改变程序走向，那么程序必将往错误的分支走
>
> 模运算步长加密：这个目前还不怎么了解
>
> 非自然程序流程
>
> 最后还有一堆算法需要了解

前置条件

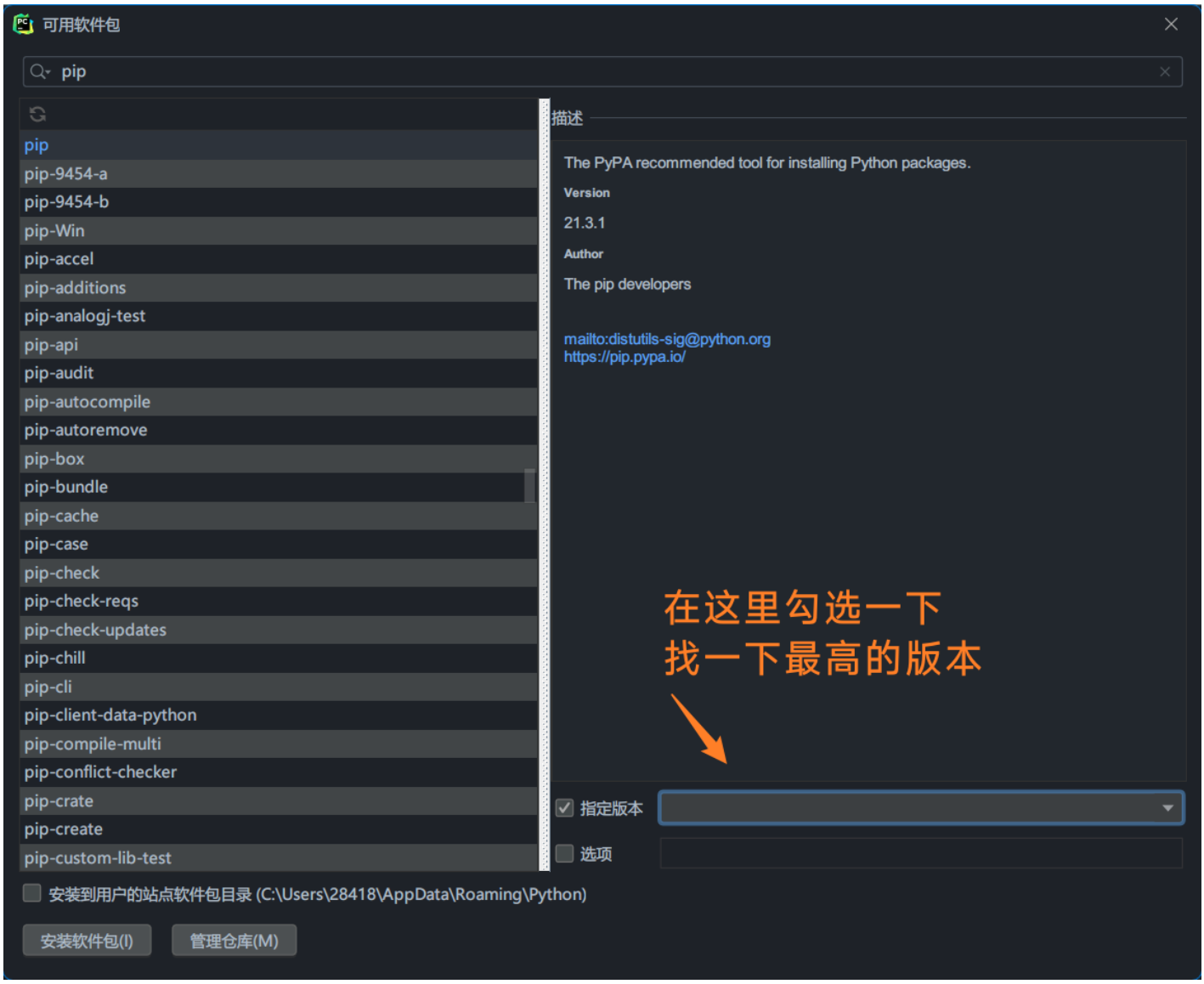
你要准备好你的环境，工具等，然后要了解一下每一种题型的大概模样（什么是大概模样，每种题型都做一道题吧，以后遇到可以类比 ）

环境的话正好我到处找了一下ctf常见库分享一下

python库安装可以命令行pip install xxx，或者去pycharm里面设置



建议先把pip更新到最新



gmpy2库

Crypto库（不建议安装这个库,建议安装pycrtpyo这个库，这crypto好像挂了，坑死我了好几十次）

sympy库

z3库

hashlib库

base64库

Pillow库

requests库（逆向不太需要在意，这个是web同志常见的库）

pwntools库

等等，里面包含了各个方向的库，其实我之前还找到20多个库，好像对ctf不太重要，这里就不推荐了

> 解释一下，Python正因为有了这些库，会比其他语言简单好上手，我个人理解是，这些库就相当与c语言和C++里面的函数，或者java里面的方法，如果在java,c里面做的话，需要自己写（所以这就是python为啥是被用作ctf首选解题脚本语言）python还有一些东西，在数学上面非常优秀，所以大数据，人工智能等等对python的依赖高，但不代表python学好了，你就年薪几十万，在这里奉劝一句除非你是学数据，人工智能等专业，可以将python作为第一语言，因为他们还有各自专业领域的东西需要python辅助，其他专业千万别把python作为第一语言，除非你真6，非要用python做一些东西我也无话可说

工具准备

> 吾爱爱盘里面一堆，学我，全下下来，要啥找啥，或者吾爱虚拟机，那个虚拟机真用不习惯

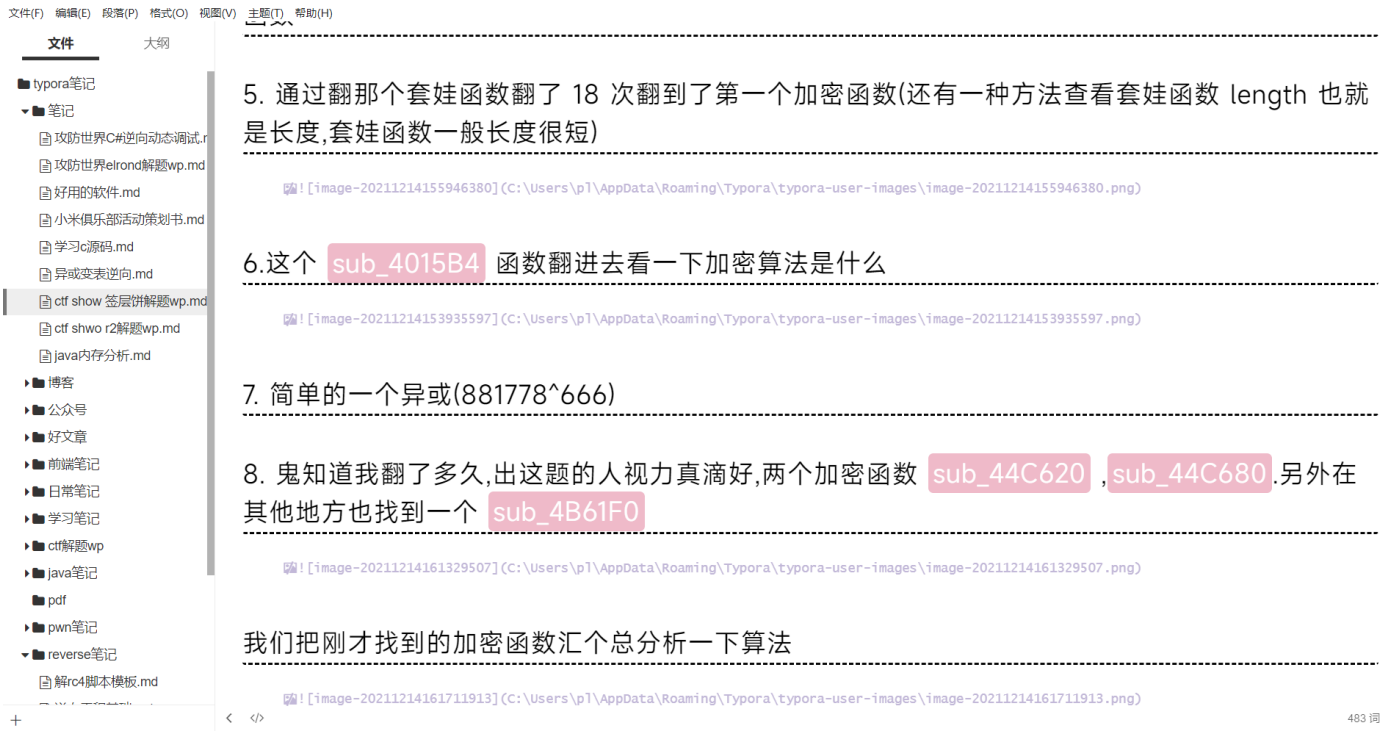
实战学习

z3：

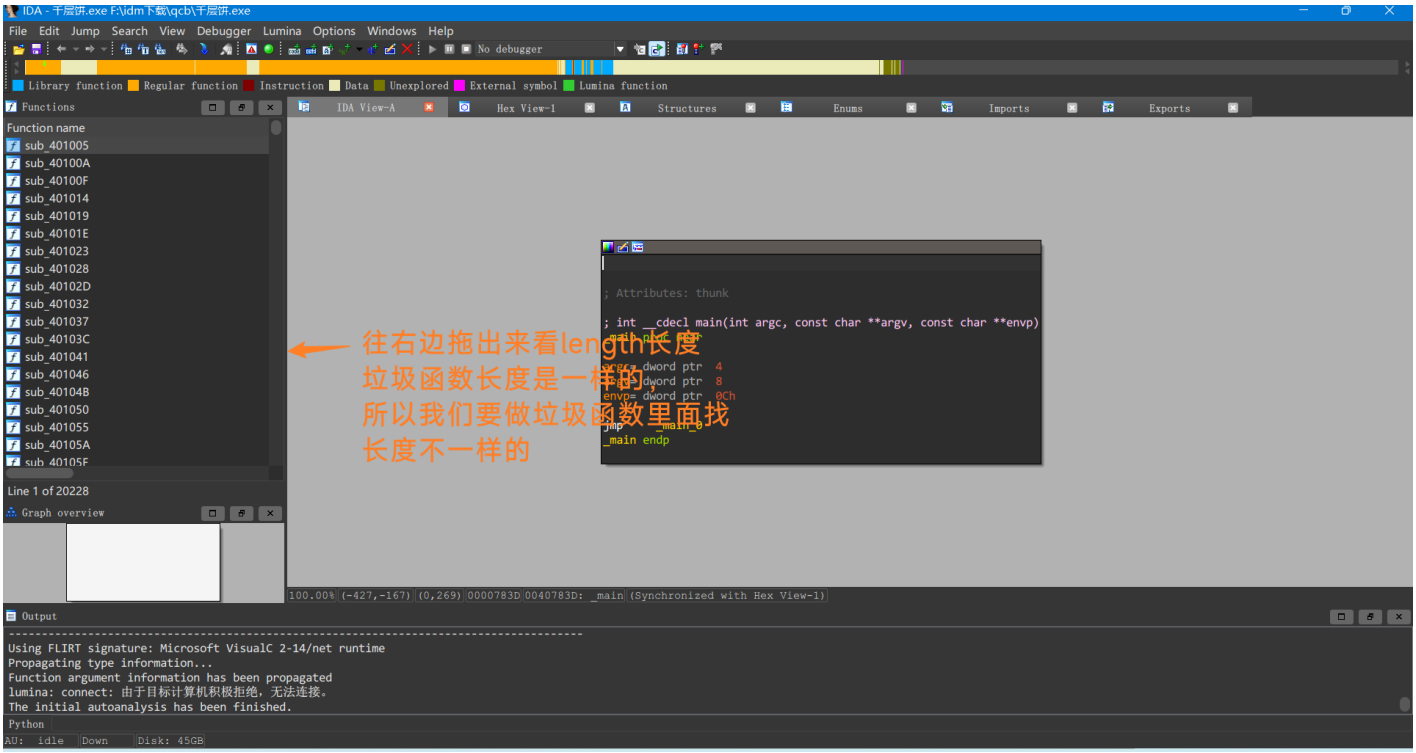
首先我打算讲一道题，千层饼，这道健神的题太6了，我在b站曾经看视频，就有大佬把这道题当例题讲（健神yyds）

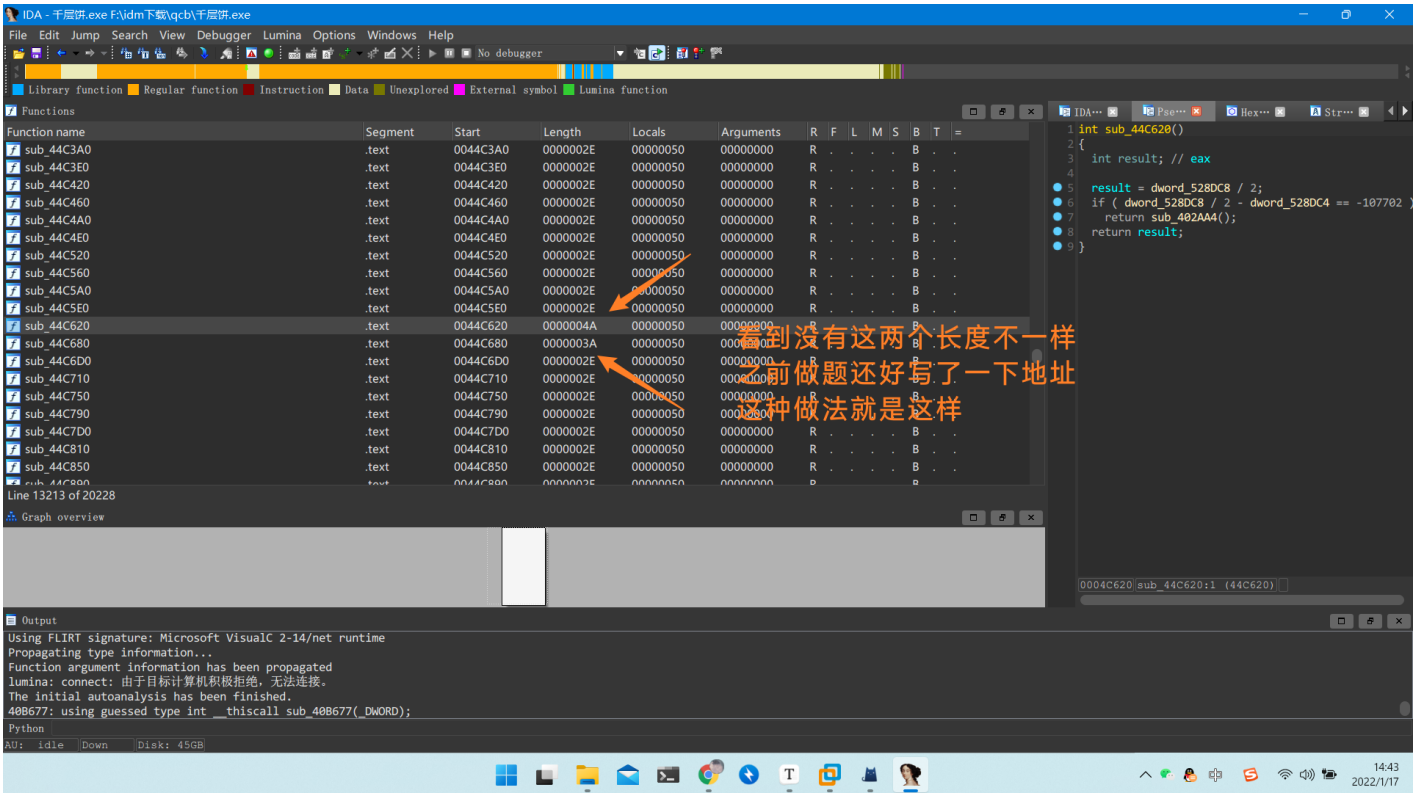
首先这道题我之前的做法是找关键函数，然后手算出来的，不过由于重装系统，之前写作没有打图片，图片没了（说到这里就心酸我的好多wp图片没了呜呜呜）

好了先讲一下第一种解法然后再讲z3（附上之前由健神指导做出来的这道题，也感谢另一位逆向朋友的指导，然后做出的wp）



简单讲一下做法，





好了，今天重点是z3

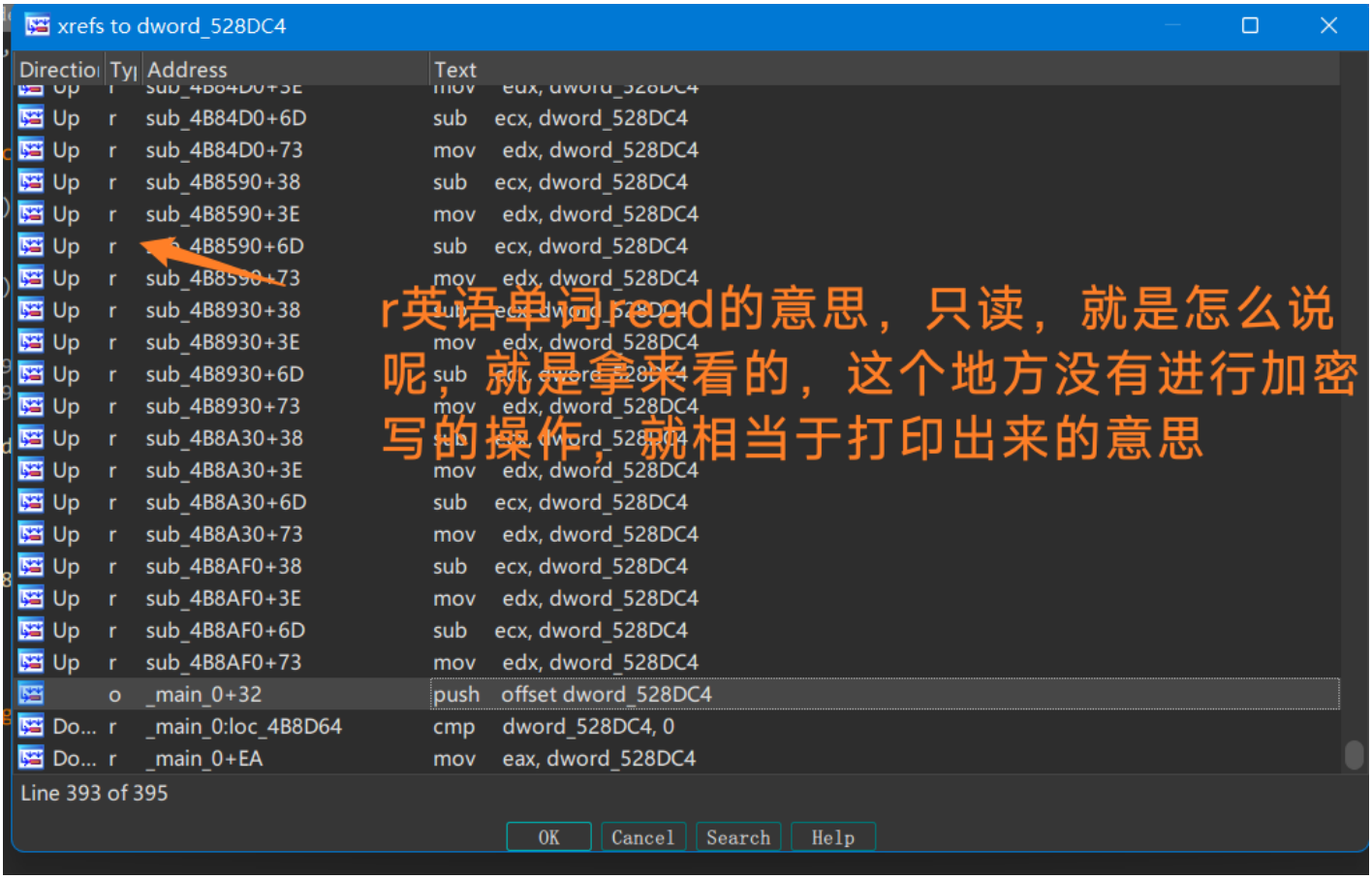
首先反编译直接看伪代码

```
int __cdecl main_0(int argc, const char **argv, const char **envp)
{
  __time32_t v3; // eax

  printf("Hello!Welcome to ctfshow,You need to input two numbers\n");
  printf("number1:");
  scanf("%d", &dword_528DC4);
  printf("\n");
  printf("number2:");
  scanf("%d", &dword_528DC8);
  v3 = time(0);
  sub_4B9370(v3);
  dword_528DC0 = (rand() - 999) % 500;
  if ( dword_528DC0 > 9999999 )
    printf(&byte_523064);
  if ( dword_528DC4 > 0 && dword_528DC8 > 0 )
  {
    sub_40431D();
    if ( dword_525A30 )
    {
      printf("%d", dword_528DC0);
      printf("Key Error");
    }
    else
    {
      printf("yeah!Your flag:ctfshow{c52e1e1a33%d0e%dc}", dword_528DC4, dword_528DC8);
    }
  }
  else
  {
    printf("Error");
  }
000B8D91 _main_0:19 (4B8D91)
```

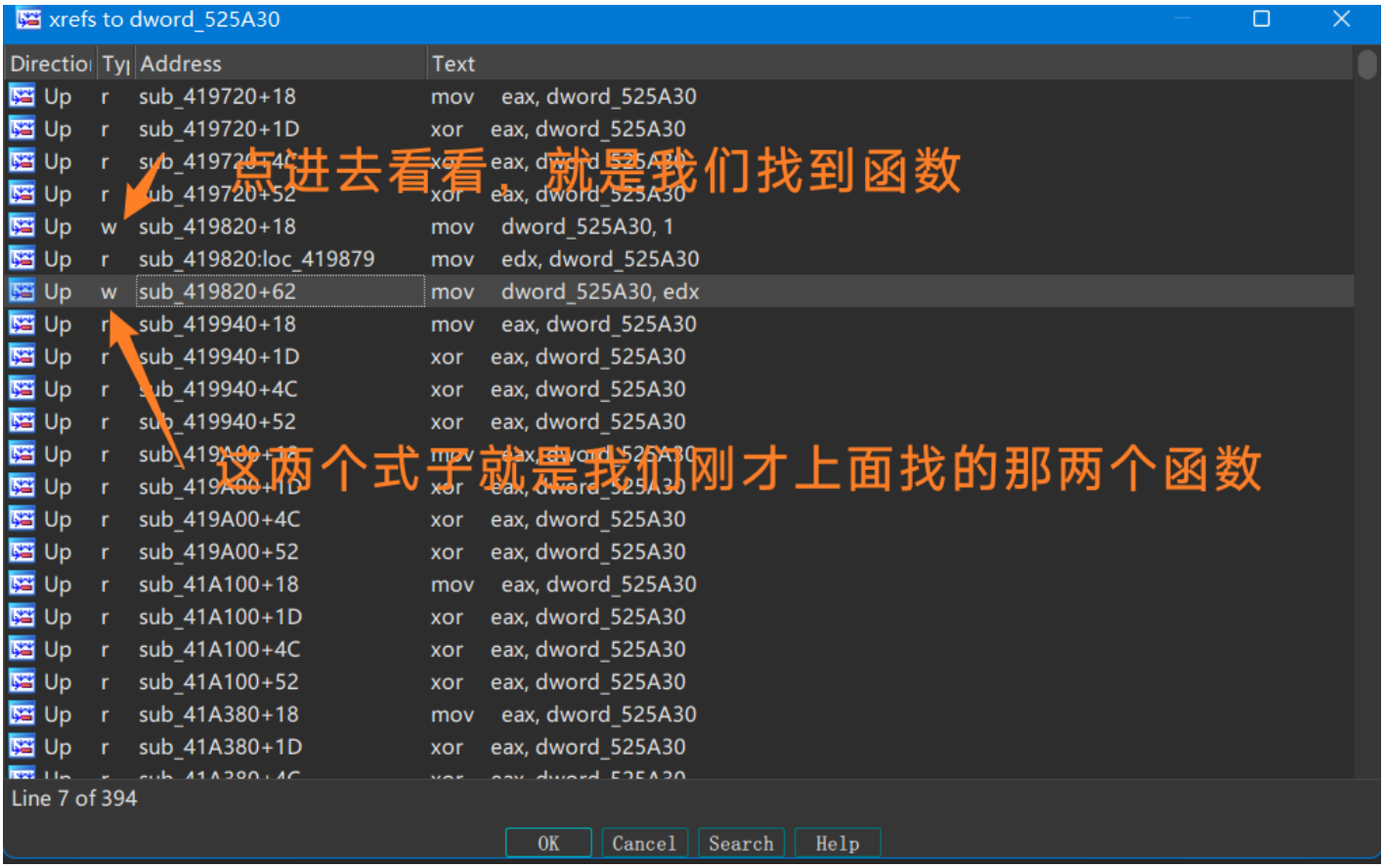
然后分析一下，我们找一下，输入输出条件判断地方的数据情况，点击dword_525A30按下快捷键x

假吧意思我点了第一个scanf后面的数据查看了情况



r英语单词read的意思，只读，就是怎么说呢，就是拿来看的，这个地方没有进行加密写的操作，就相当于打印出来的意思

> 在这里解释一下r代表只读的意思（只起到打印的作用）这是个人理解哈
>
> w代表只写的操作
>
> o忘了。。。。

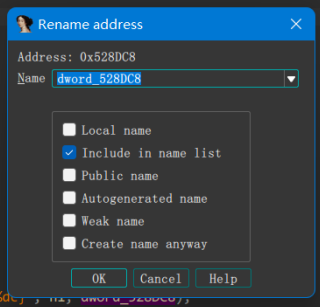


为了便于我们好分析，建议回去改一下名称，因为这些名称是ida自己生成的

```
int __cdecl main_0(int argc, const char **argv, const char **envp)
{
  __time32_t v3; // eax

  printf("Hello!Welcome to ctfshow,You need to input two numbers\n");
  printf("number1:");
  scanf("%d", &n1);
  printf("\n");
  printf("number2:");
  scanf("%d", &dword_528DC8);
  v3 = time(0);
  sub_4B9370(v3);
  dword_528DC0 = (rand() - 999) % 500;
  if ( dword_528DC0 > 9999999 )
    printf(&byte_523064);
  if ( n1 > 0 && dword_528DC8 > 0 )
  {
    sub_40431D();
    if ( dword_525A30 )
    {
      printf("%d", dword_528DC0);
      printf("Key Error");
    }
    else
    {
      printf("yeah!Your flag:ctfshow{c52e1e1a33%d0e%de} ", n1, dword_528DC0);
    }
  }
  else
  {
    printf("Error");
  }
000B8D0E main_0:10 (4B8D0E)
```

点击后按n快捷键改名字，上面我已经改了



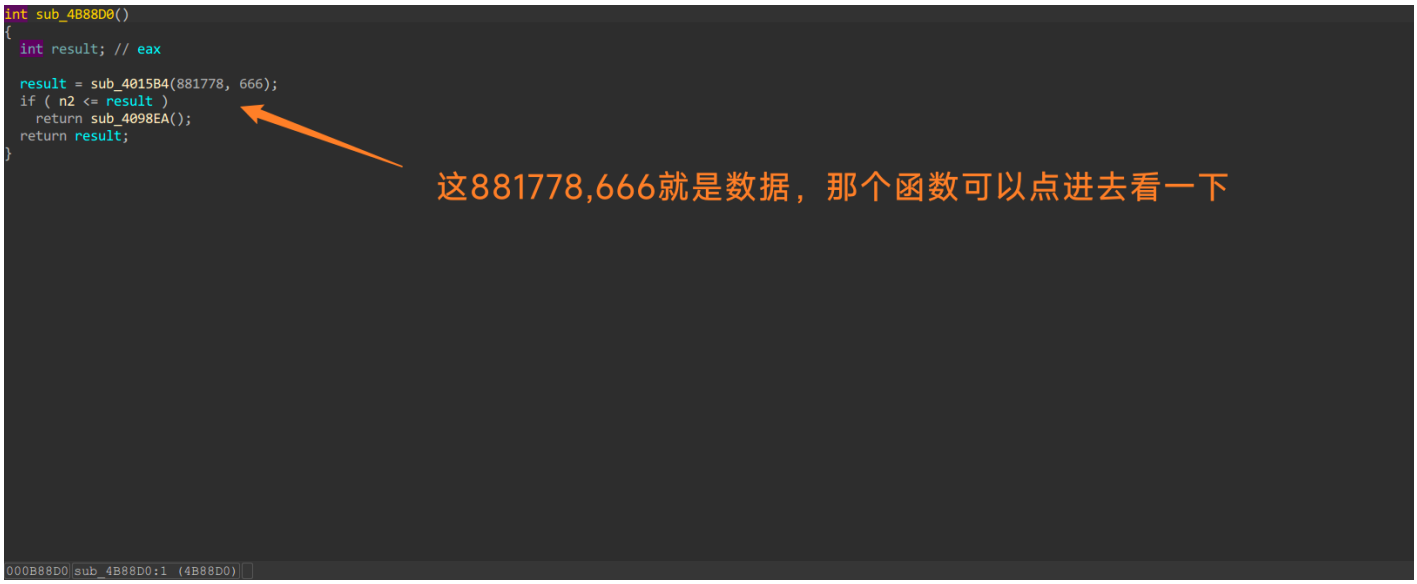

```
int sub_419820()
{
  dword_525A30 = 1;
  if ( dword_528DC0 * dword_528DC0 * n2 - 1877 * dword_528DC0 + 1 < 0
    || dword_528DC0 * dword_528DC0 * n2 - 1877 * dword_528DC0 != -1 )
  {
    dword_525A30 ^= 1u;
  }
  return sub_408724();
}
00019838 sub_419820:3 (419838)
```

发现这里还有个东西，我们在这里也改一下名字

```
int sub_419820()
{
  dword_525A30 = 1;
  if ( index * index * n2 - 1877 * index + 1 < 0 || index * index * n2 - 1877 * index != -1 )
    dword_525A30 ^= 1u;
  return sub_408724();
}
00019838 sub_419820:3 (419838)
```

名字随便改，把这个式子记录下来，然后下面还有个函数同样记录下来，分析一下，发现少一个数据

其实这道题还是需要自己翻一下那个函数，还好就在17层



里面好像有个异或

我们已经得到数据了，然后就是写z3脚本了

前置条件安装z3库

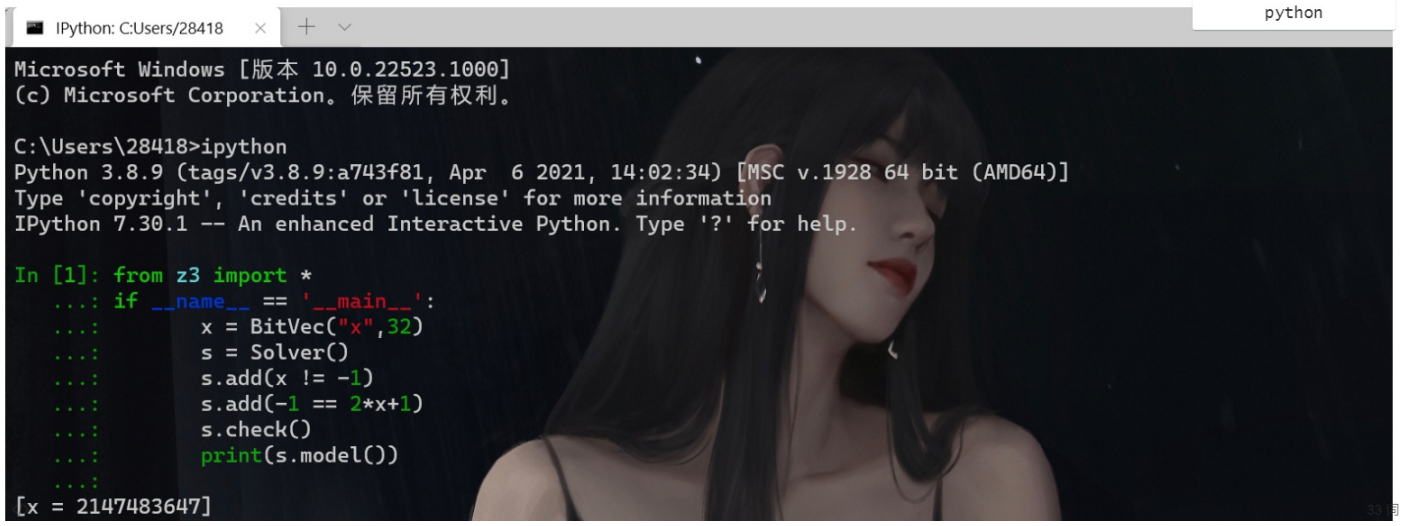
```
pip install z3-solver
```

```
from z3 import*

x, y = BitVecs('x,y',32)
x = BitVecs ('x',32)
s = Solver()
s.add(y/2-x == -107702)
s.add(y & 1 == 0)
s.add(y ^ 333509 == x)
s.add(881778^666)
print(s.check())
'SAT'
print(s.model())
//这个脚本是python2别人写的，所以要用python2跑
```

z3的话还有一道题，大家也可以去试一下另一道题（步骤差不多）

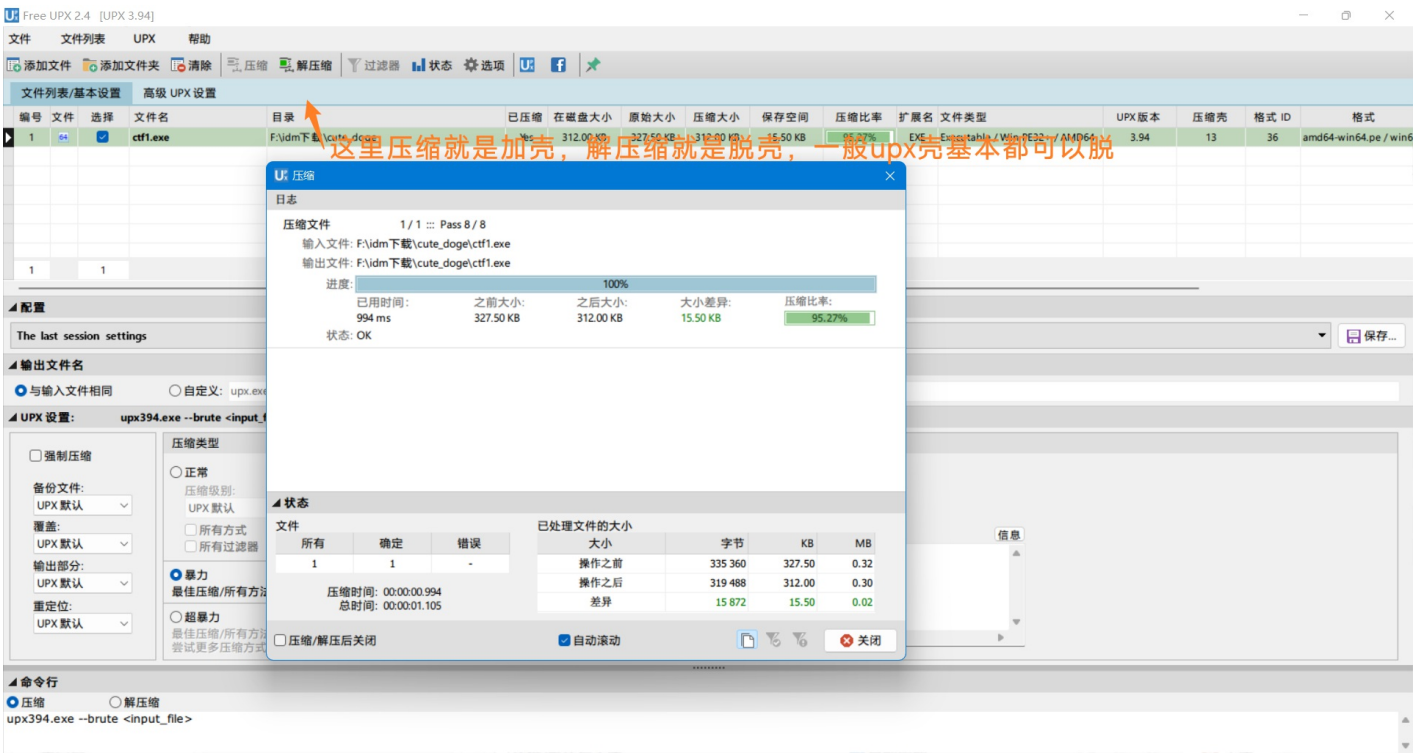
```
from z3 import *
if __name__ == '__main__':
        x = BitVec("x",32)
        s = Solver()
        s.add(x != -1)
        s.add(-1 == 2*x+1)
        s.check()
        print(s.model())
```




```
from z3 import *
if __name__ == '__main__':
        x = BitVec("x",32)
        s = Solver()
        s.add(x != -1)
        s.add(-1 == 2*x+1)
        s.check()
        print(s.model())
```

脱壳

学习加壳，然后自己再脱壳



base家族

查码表，不过谁去查呀，随波逐流干就完事

……