

ctf之逆向常见题型

原创

thesatrx 于 2022-01-23 21:38:49 发布 3270 收藏

文章标签: [安全](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/thesat/article/details/122657729>

版权

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)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, xtea, 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

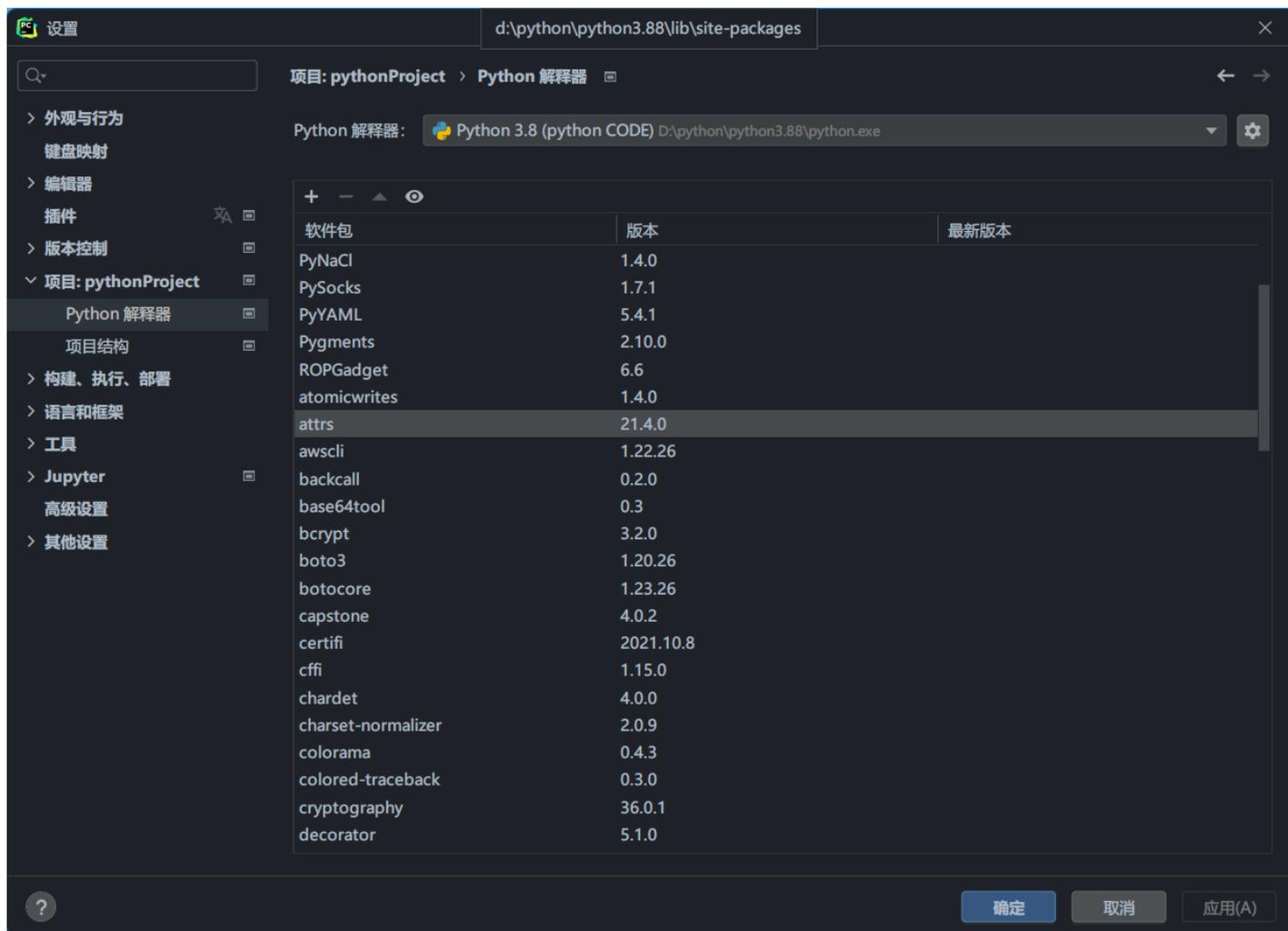
最后还有一堆算法需要了解

前置条件

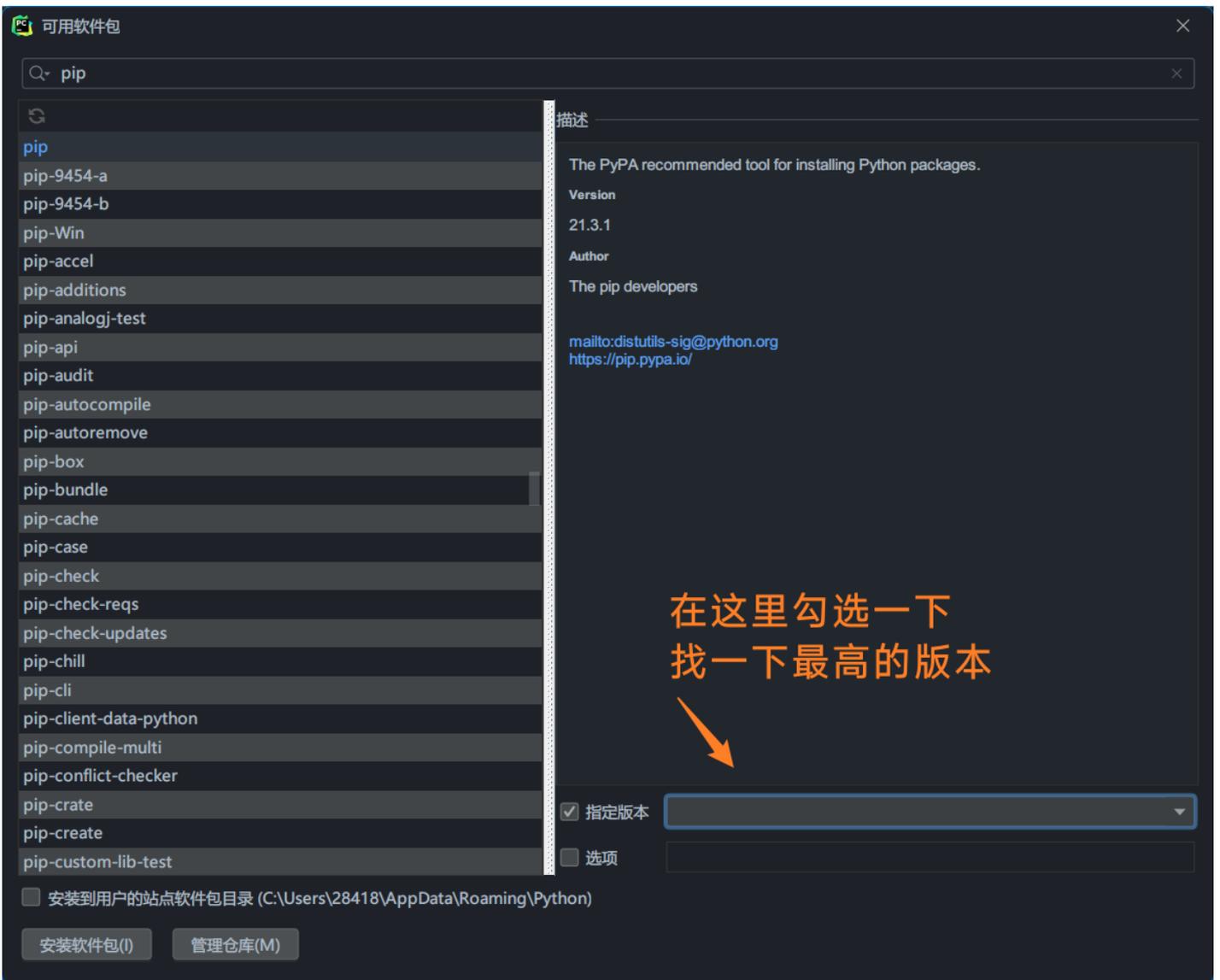
你要准备好你的环境, 工具等, 然后要了解一下每一种题型的大概模样(什么是大概模样, 每种题型都做一道题吧, 以后遇到可以类比)

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pycryptpyo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

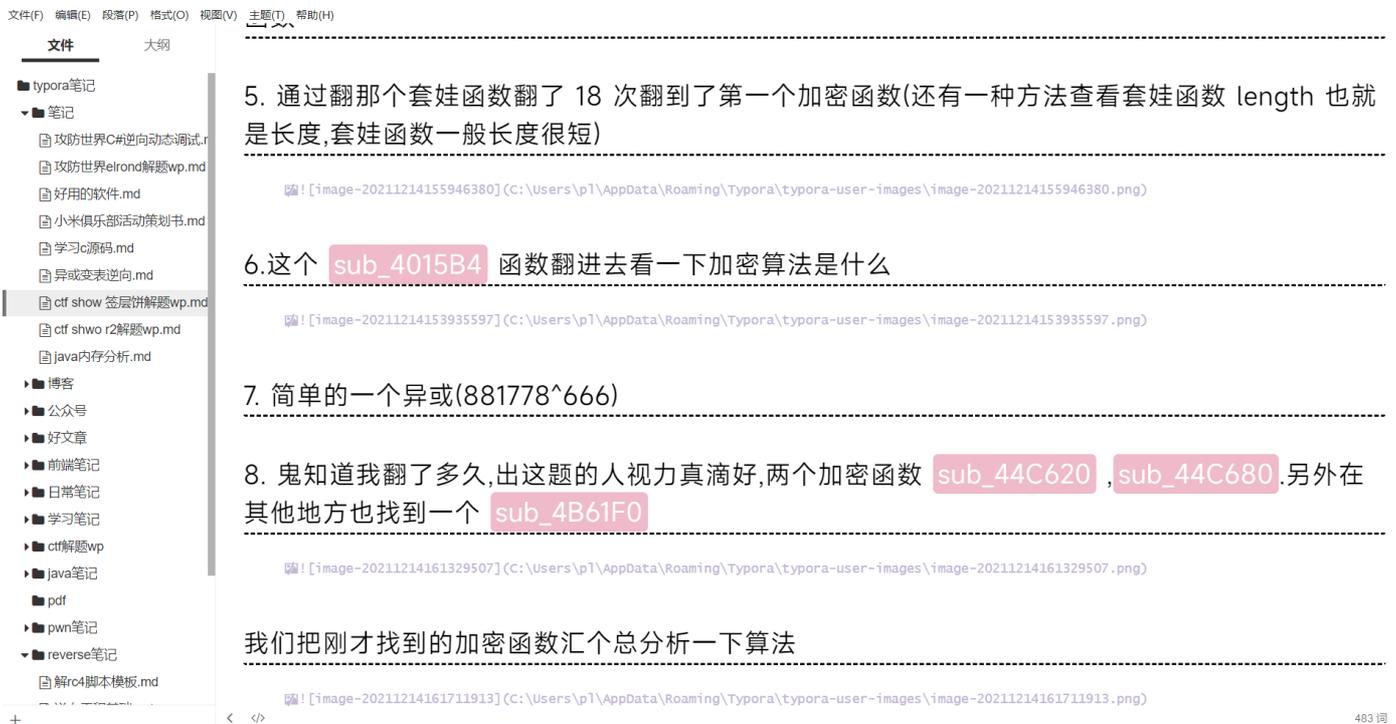
实战学习

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)

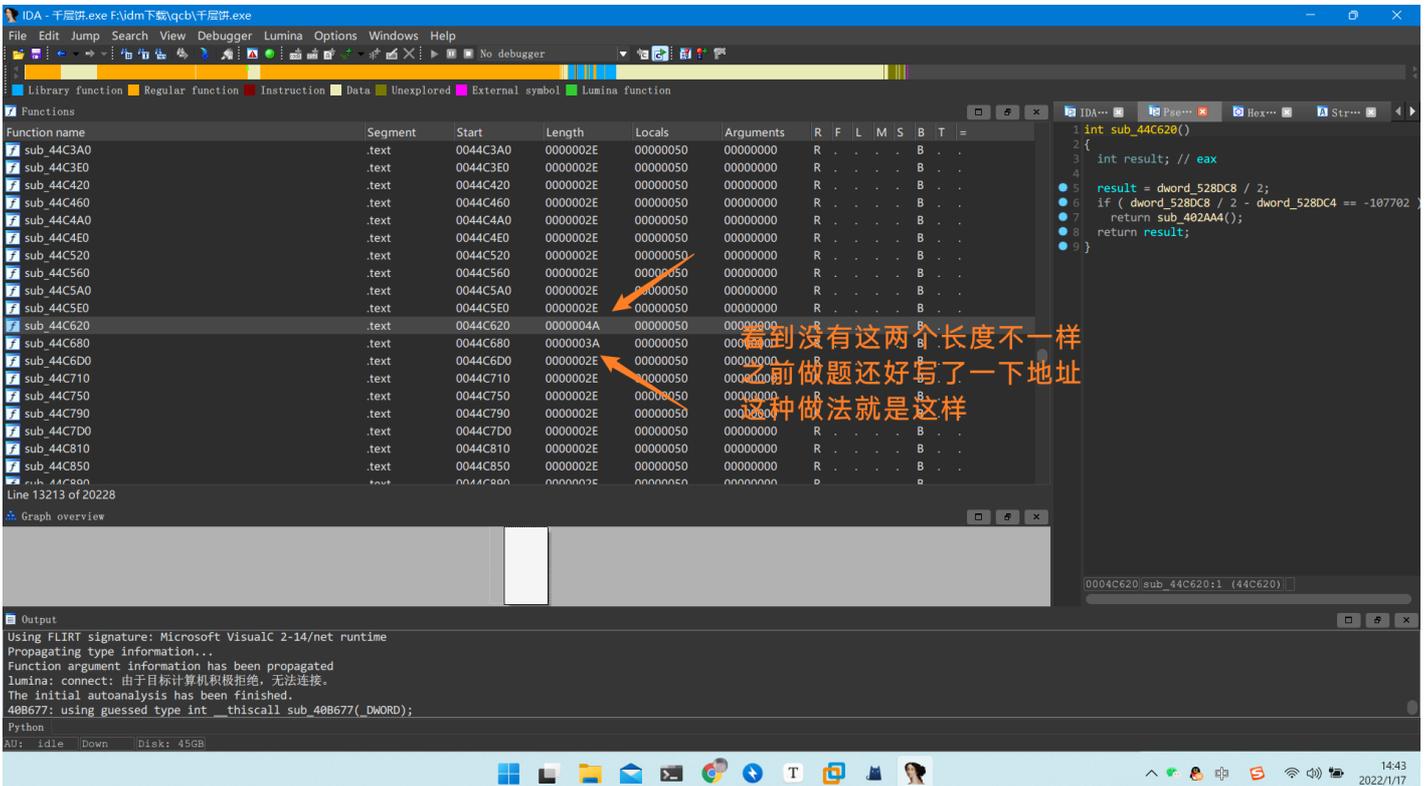
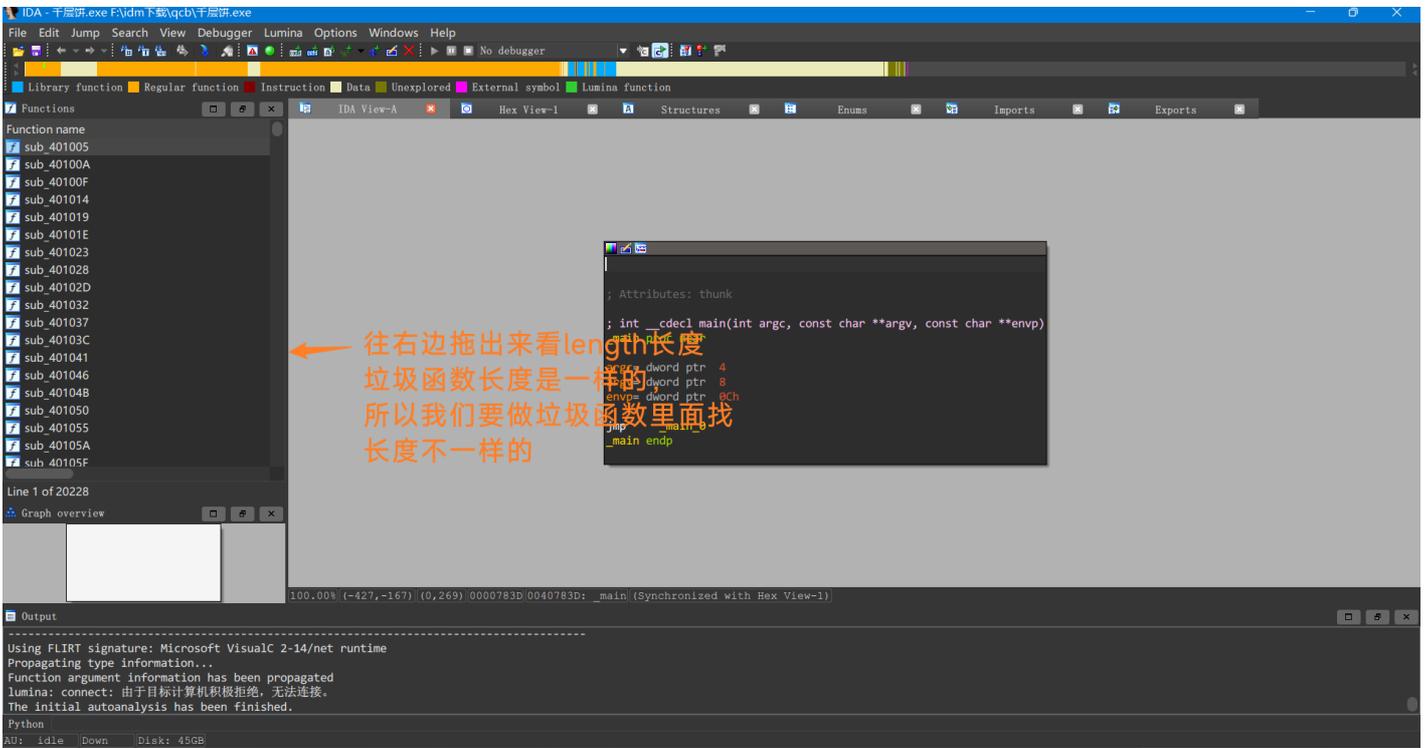


The image shows a screenshot of a Typora text editor. On the left is a file explorer sidebar with a tree view containing various folders and files, including 'typora笔记', '笔记', '攻防世界C#逆向动态调试.r', '攻防世界elrond解题wp.md', '好用的软件.md', '小米俱乐部活动策划书.md', '学习c源码.md', '异或变表逆向.md', 'ctf show 签到讲解wp.md', 'ctf show r2解题wp.md', 'java内存分析.md', '博客', '公众号', '好文章', '前端笔记', '日常笔记', '学习笔记', 'ctf解题wp', 'java笔记', 'pdf', 'pwn笔记', 'reverse笔记', and '解rc4脚本模板.md'. The main editor area displays a list of notes, each separated by a dashed line. The notes are:

- 5. 通过翻那个套娃函数翻了 18 次翻到了第一个加密函数(还有一种方法查看套娃函数 length 也就是长度,套娃函数一般长度很短)
- 6. 这个 `sub_4015B4` 函数翻进去看一下加密算法是什么
- 7. 简单的一个异或(881778^6666)
- 8. 鬼知道我翻了多久,出这题的人视力真滴好,两个加密函数 `sub_44C620`, `sub_44C680`.另外在其他地方也找到一个 `sub_4B61F0`

Below the list, there is a section titled '我们把刚才找到的加密函数汇个总分析一下算法'.

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");
    }
}
```

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

Direction	Type	Address	Text
Up	r	sub_4B84D0+3E	mov ecx, dword_528DC4
Up	r	sub_4B84D0+6D	sub ecx, dword_528DC4
Up	r	sub_4B84D0+73	mov edx, dword_528DC4
Up	r	sub_4B8590+38	sub ecx, dword_528DC4
Up	r	sub_4B8590+3E	mov edx, dword_528DC4
Up	r	sub_4B8590+6D	sub ecx, dword_528DC4
Up	r	sub_4B8590+73	mov edx, dword_528DC4
Up	r	sub_4B8930+38	sub ecx, dword_528DC4
Up	r	sub_4B8930+3E	mov edx, dword_528DC4
Up	r	sub_4B8930+6D	sub ecx, dword_528DC4
Up	r	sub_4B8930+73	mov edx, dword_528DC4
Up	r	sub_4B8A30+38	sub ecx, dword_528DC4
Up	r	sub_4B8A30+3E	mov edx, dword_528DC4
Up	r	sub_4B8A30+6D	sub ecx, dword_528DC4
Up	r	sub_4B8A30+73	mov edx, dword_528DC4
Up	r	sub_4B8AF0+38	sub ecx, dword_528DC4
Up	r	sub_4B8AF0+3E	mov edx, dword_528DC4
Up	r	sub_4B8AF0+6D	sub ecx, dword_528DC4
Up	r	sub_4B8AF0+73	mov edx, dword_528DC4
o		_main_0+32	push offset dword_528DC4
Do...	r	_main_0:loc_4B8D64	cmp dword_528DC4, 0
Do...	r	_main_0+EA	mov eax, dword_528DC4

Line 393 of 395

OK Cancel Search Help

r 英语单词 read 的意思，只读，就是怎么说呢，就是拿来查看的，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。。。

Direction	Type	Address	Text
Up	r	sub_419720+18	mov eax, dword_525A30
Up	r	sub_419720+1D	xor eax, dword_525A30
Up	r	sub_419720+4C	xor eax, dword_525A30
Up	r	sub_419720+52	xor eax, dword_525A30
Up	w	sub_419820+18	mov dword_525A30, 1
Up	r	sub_419820:loc_419879	mov edx, dword_525A30
Up	w	sub_419820+62	mov dword_525A30, edx
Up	r	sub_419940+18	mov eax, dword_525A30
Up	r	sub_419940+1D	xor eax, dword_525A30
Up	r	sub_419940+4C	xor eax, dword_525A30
Up	r	sub_419940+52	xor eax, dword_525A30
Up	r	sub_419A00+18	mov eax, dword_525A30
Up	r	sub_419A00+1D	xor eax, dword_525A30
Up	r	sub_419A00+4C	xor eax, dword_525A30
Up	r	sub_419A00+52	xor eax, dword_525A30
Up	r	sub_41A100+18	mov eax, dword_525A30
Up	r	sub_41A100+1D	xor eax, dword_525A30
Up	r	sub_41A100+4C	xor eax, dword_525A30
Up	r	sub_41A100+52	xor eax, dword_525A30
Up	r	sub_41A380+18	mov eax, dword_525A30
Up	r	sub_41A380+1D	xor eax, dword_525A30
Up	r	sub_41A380+4C	xor eax, dword_525A30

Line 7 of 394

OK Cancel Search Help

点进去看看就是我们找到函数

这两个式子就是我们刚才上面找的那两个函数

```

int sub_419820()
{
    dword_525A30 = 1;
    if ( dword_528DC0 * dword_528DC0 * dword_528DC8 - 1877 * dword_528DC0 + 1 < 0
        || dword_528DC0 * dword_528DC0 * dword_528DC8 - 1877 * dword_528DC0 != -1 )
    {
        dword_525A30 ^= 1u;
    }
    return sub_408724();
}

```

00019838:sub_419820:3 (419838)

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 sub_4B88D0()
{
    int result; // eax
    result = sub_4015B4(881778, 666);
    if ( n2 <= result )
        return sub_4098EA();
    return result;
}
```

这881778,666就是数据，那个函数可以点进去看一下

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

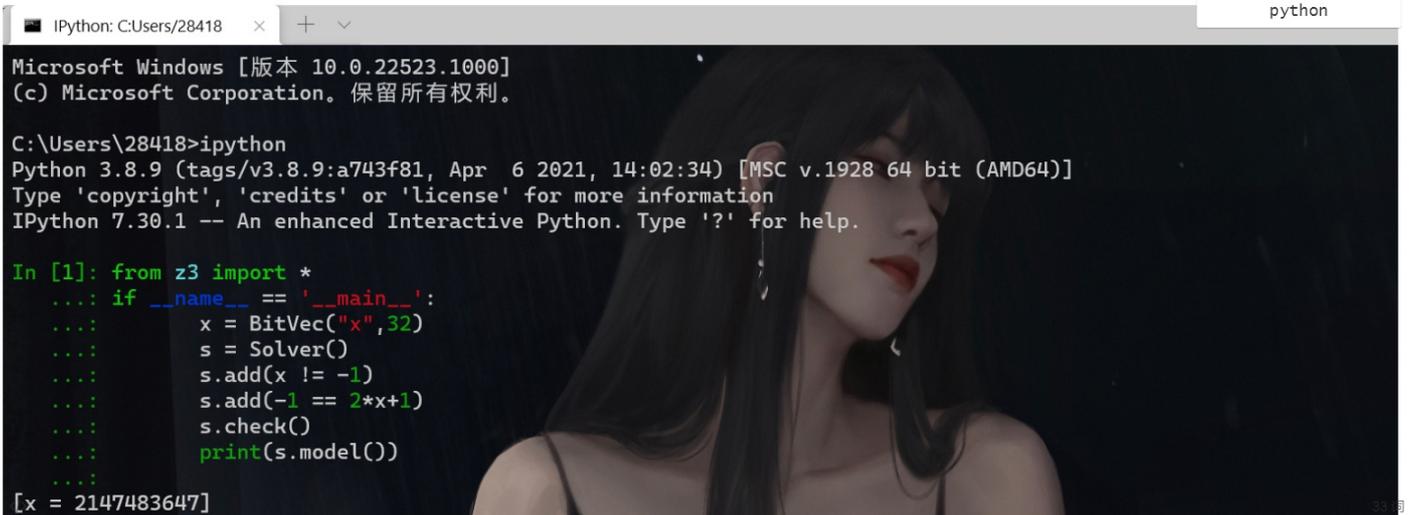
```
pip install z3-solver
```

```
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）

```
1 from z3 import *
2 if __name__ == '__main__':
3     x = BitVec("x",32)
4     s = Solver()
5     s.add(x != -1)
6     s.add(-1 == 2*x+1)
7     s.check()
8     print(s.model())
9
```



```
IPython: C:\Users\28418 x + v python
Microsoft Windows [版本 10.0.22523.1000]
(c) Microsoft Corporation。保留所有权利。

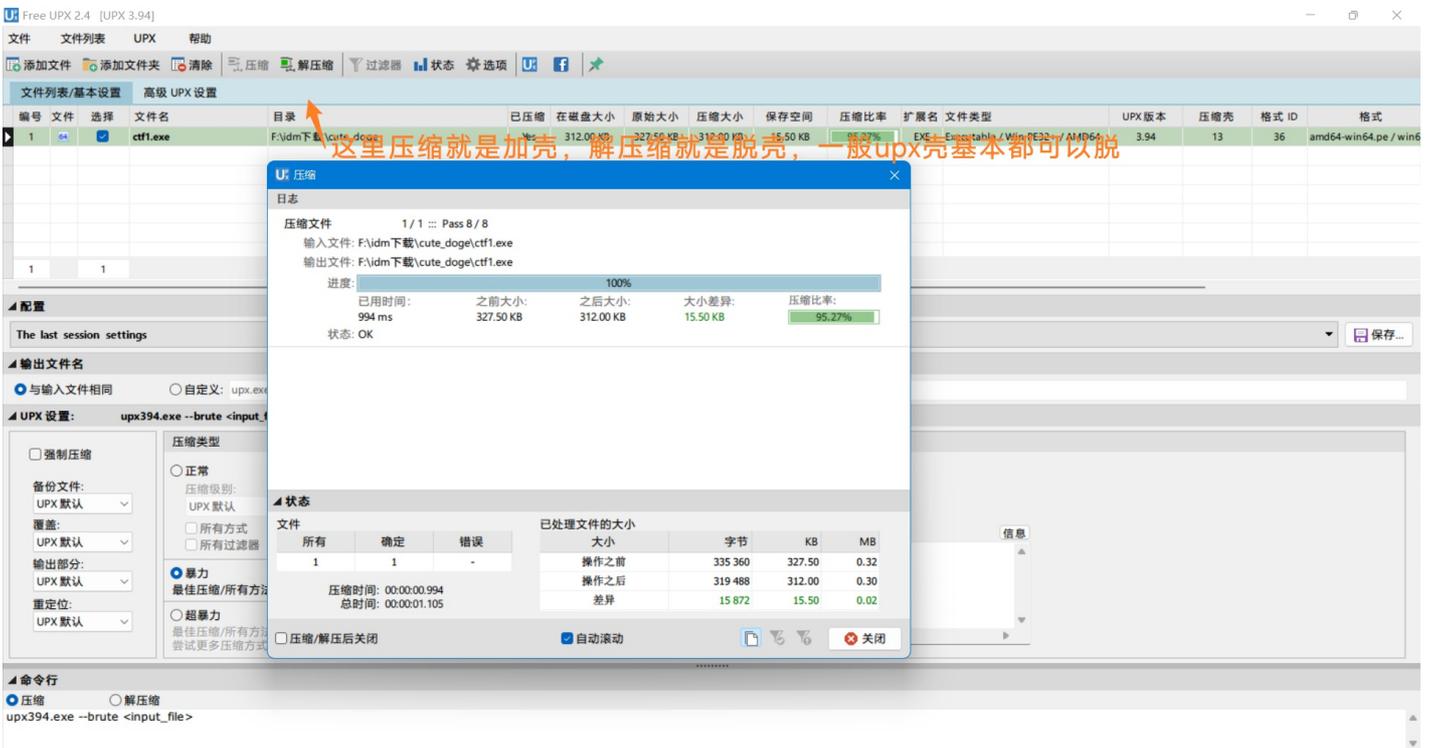
C:\Users\28418>ipython
Python 3.8.9 (tags/v3.8.9:a743f81, Apr 6 2021, 14:02:34) [MSC v.1928 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.30.1 -- An enhanced Interactive Python. Type '?' for help.

In [1]: 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())
...:
[x = 2147483647]
```

```
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

.....