whatamitoyou-小白详解

原创

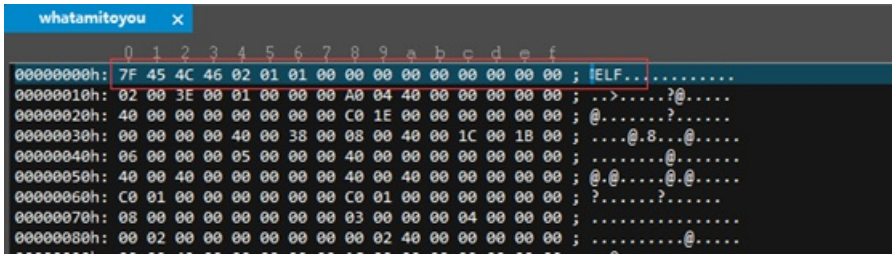
时光难逆 于 2019-08-15 12:10:08 发布 1831 收藏
文章标签： 逆向 CTF whatamitoyou

本文链接：https://blog.csdn.net/u014101410/article/details/99628404

实验吧题目的writeup（小白详细（图））是本人，不存在抄袭，writeup不知道成功没，查询不到，所以在这里简略备份记录一下研究成果；

UE打开

可以看到是文件头是ELF，内部有linux的库字符，使用gcc编译；去linux系统下看文件



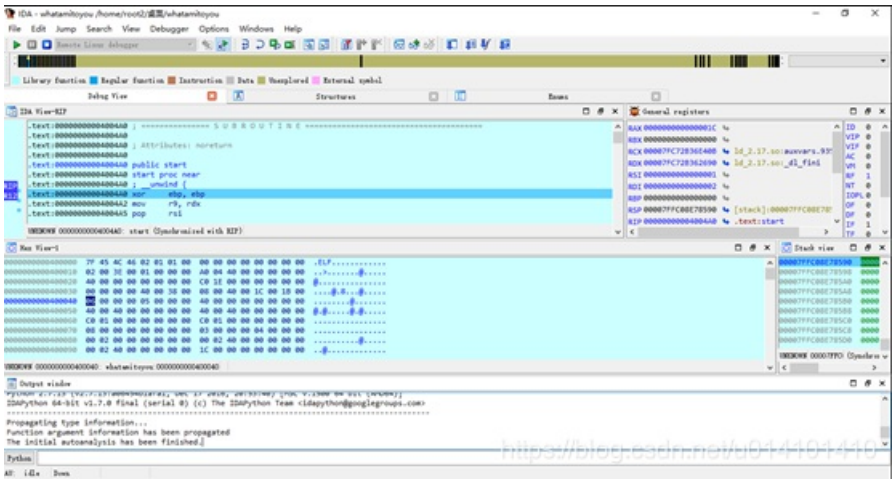
ELF信息

百度得知ELF是linux下的文件头，运行readelf –h whatamitoyou可以查看文件头信息

可以看到该文件是AMDx86-64生成的64位可执行文件，我们环境支持运行，chmod 744给权限后运行一下，发现出错

初步猜测是内存溢出或者内部代码计算出问题，用gdb试探一下，发现在40191C处执行出错，但是没有捕捉到函数名之类的额外提示，意料之中

调试还是要专业工具来，安装好IDA后将linux_server64拷贝到linux系统中，给权限运行如果程序是32位就拷贝linux_server（看雪：IDA-Linux教程https://bbs.pediy.com/thread-247830-1.htm）



几个指令段转伪代码看看，前面gdb提示错误在40191C处，转到，F5查看伪代码

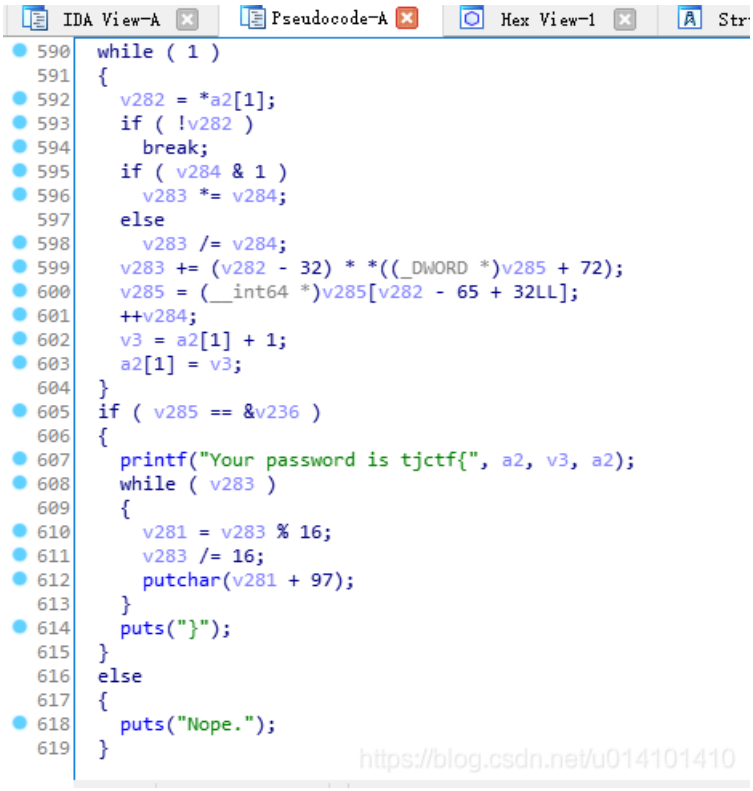
正好是main函数

关键逻辑如下

v282、v283、v285与生成KEY有关，变量定义v283=0，前后代码分析，该变量是统计字符长度；v285=&v140
那么我们去看v140是什么，变量定义为8字节，变量赋值转为char字符后发现是一个单词，依次转到
v144（140~144代码中是个char数组？）

```
memset(&v158, 0, 0xC8uLL);
v163 = 7;
memset(&v140, 0, 0x128uLL);
v140 = 'enoyrevE';
v141 = 'bbuB ...';
v142 = '...mugel';
v143 = ' os m\'I ';
v144 = 'bmud';
memset(&v145, 0, 0xD8uLL);
v150 = 724;
memset(&v128, 0, 0x128uLL);
v128 = 7451608144132923724LL;
v129 = 8367814970102019176LL;
v130 = 2338621028921533800LL;
v131 = 7311074240112784742LL;
v132 = 7018895694957536544LL;
v133 = 11364LL;
memset(&v134, 0, 0xD0uLL);
```

循环中v282是我们输入的参数，用于计算指针偏移量，拼接字符串，而他的定义为char



根据题目和前期出现的字符串，可以搜索到一首歌My Best Friends InThe World(What Am I To You?)。对比这首歌可以看出hex内的歌词顺序是乱，且Key的长度应该是歌词的句数（n=35），成功整理歌词顺序则可获取KEY。

断点到while下第一句，然后运行程序，打开视图->Strings；查看内存中各句歌词的地址（如果没有，就断点在v283处，重新打开Strings）

歌词顺序可知：

歌词第一句为”Everyone, Bubblegum I’m so dumb”,下一句是”I should have just told you”

我们搜索这两句

第一句7FFF718A4E10

第二句7FFF718A46F0

计算偏移量为6F0-E10=-720；偏移量居然出现负数，这明显不对，按理说第二句应该是第一句指针后移一定量的，IDA内存读取不行啊；（这里小白研究了近5个小时，没研究出结果，后面上网搜索视频解答，发现是IDA的锅）

EDB

kali打开edb debugger调试，执行命令“edb --run '/root/whatamitoyou' aaaaaaaaaaaaaa”，根据IDA中获取到的地址，在edb中断点到“00000000004018ED”

运行程序到断点处，然后在内存中转到rax地址处查看，是第一句歌词（0x00007ffebe03fce0），接下来搜索第二句歌词位置

有两处cb8和df0，很明显应该是df0处，不过这里我将两个地址在内存中的数据都查看了，分析一下为啥IDA找的是错误的，明显IDA找到的是cb8位置出的，没有找到df0处的，这里两处其实都是指向5c0处，不知道为什么IDA没找到后面哪个，毕竟用的绿色版，唉

KEY获取计算

汇总上面分析结果：

第一句歌词地址0x00007ffebe03fce0；

第二句歌词地址0x00007ffebe03fdf0；

指针汇编语句mov　　rax, [rax+rdx*8]；

伪代码v285 = (__int64 *)v285[v282 - 65 + 32LL];

可得计算公式：

v282 - 65 + 32=rdx；//rdx由前面的汇编计算出来，然后传给指针汇编语句

0x00007ffebe03fce0+rdx*8=0x00007ffebe03fdf0；//第一句歌词偏移后指针指向第二句歌词

计算出

rdx*8=0x110，rdx=0x22

v282-33=34，v282=67 //ASCII “C”

因此第一个字符为ASCII码67，即字符C；

依次类推，得到剩下的输入。

完整的输入为CBDABCADBCCABBABBABACBCCABDADBABABB

加上这个参数运行程序得到“Your password is tjctf{pblgjd}”