攻防世界逆向——key

原创

jesus_H 于 2019-05-26 17:48:05 发布 1363 收藏 1

分类专栏： web安全 reversing

本文链接：https://blog.csdn.net/qq_41667282/article/details/90578431


web安全 同时被 2 个专栏收录

2 篇文章 2 订阅

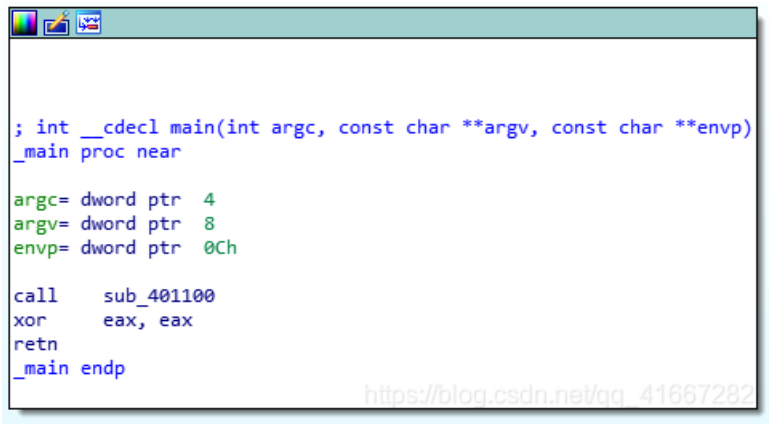
订阅专栏


reversing

4 篇文章 0 订阅

订阅专栏

用ida打开程序，可以看到main函数，调用了一个子函数



```
; int __cdecl main(int argc, const char **argv, const char **envp)
_main proc near

argc= dword ptr  4
argv= dword ptr  8
envp= dword ptr  0Ch

call    sub_401100
xor     eax, eax
retn
_main endp
```


点进去子函数，然后按F5，可以看到有100多行的代码。看到这里无从下手。我们先定位关键的字符串。把代码往下拉可以看到有"Congrats you got it!"的字符串。

```
v14 = sub_402A00(std::cout, "|------------------------------|", sub_402C50);
std::basic_ostream<char,std::char_traits<char>>::operator<<(v14, v15);
v16 = sub_402A00(std::cout, "|==============================|", sub_402C50);
std::basic_ostream<char,std::char_traits<char>>::operator<<(v16, v17);
v18 = sub_402A00(std::cout, "|==============================|", sub_402C50);
std::basic_ostream<char,std::char_traits<char>>::operator<<(v18, v19);
v20 = sub_402A00(std::cout, "|==============================|", sub_402C50);
std::basic_ostream<char,std::char_traits<char>>::operator<<(v20, v21);
v22 = sub_402A00(std::cout, "\\  /\\  /\\  /\\  /\\==============|", sub_402C50);
std::basic_ostream<char,std::char_traits<char>>::operator<<(v22, v23);
v24 = sub_402A00(std::cout, " \\/  \\/  \\/  \\/  \\==============|", sub_402C50);
std::basic_ostream<char,std::char_traits<char>>::operator<<(v24, v25);
v26 = sub_402A00(std::cout, "                  |--------------|", sub_402C50);
std::basic_ostream<char,std::char_traits<char>>::operator<<(v26, v27);
std::basic_ostream<char,std::char_traits<char>>::operator<<(std::cout, sub_402C50);
v28 = sub_402A00(std::cout, "Congrats You got it!", sub_402C50);
```


从这里往会看，可以看到要进入这里有一个分支语句

```
if ( sub_4020C0(&v44, v12, v45, (int)v13, v48) )
{
  v28 = sub_402A00(std::cout, "=W=r=o=n=g=K=e=y=", sub_402C50);
}
else
{
  v14 = sub_402A00(std::cout, "|------------------------------|", sub_402C50);
  std::basic_ostream<char,std::char_traits<char>>::operator<<(v14, v15);
  v16 = sub_402A00(std::cout, "|==============================|", sub_402C50);
```

sub_4020C0是一个关键函数，我们来看看函数的每个参数分别是什么。

1. 点击v44，亮黄的地方可以看到

   ```
   sub_402E90(&Dst, &v44);
   ```

   我们进入这个函数，会发现根本无法下手，我们可以先放弃
2. v12,v45,v48可以发现没有什么实际的意义
3. 然后来看一下v13，memory将值赋给了v13，所以v13就相当于memory，所以来看一下memory，有两个地方用到了memory

   ```
   do
   {
     sub_4021E0(&v41, 1u, (*((_BYTE *)Memory + v0) ^ *((_BYTE *)&v50 + v0)) + 22);
     ++v0;
   }
   while ( v0 < 18 );
   ```

```
v1 = 0;
v49 = 15;
v48 = 0;
LOBYTE(Memory[0]) = 0;
LOBYTE(v53) = 2;
v2 = v43;
v3 = (void **)v41;
do
{
  v4 = &v41;
  if ( v2 >= 0x10 )
    v4 = v3;
  sub_4021E0(Memory, 1u, *((_BYTE *)v4 + v1++) + 9);
}
while ( v1 < 18 );
```


现在看一下memory里面存着什么

```
*(_OWORD *)Memory = xmmword_40528C;
```

```
xmmword_40528C  xmmword 'htadimehtadimeht'
```

v50里面存的内容

```
xmmword_4052A4  xmmword '<<<....++++---->'
```

进去看一下sub_4021E0这个函数是做什么的。

```
if ( Size )
{
  v5 = v4 + Size;
  if ( (unsigned __int8)sub_402690(v4 + Size, v4) )
  {
    v6 = v3[4];
    if ( Size == 1 )
    {
      if ( v3[5] < 0x10u )
        *((_BYTE *)v3 + v6) = a3;
      else
        *(_BYTE *)(*v3 + v6) = a3;
    }
    else
    {
      if ( v3[5] < 0x10u )
        v7 = v3;
      else
        v7 = (_DWORD *)*v3;
      memset((char *)v7 + v6, a3, Size);
    }
    v3[4] = v5;
    if ( v3[5] >= 0x10u )
    {
      *(_BYTE *)(*v3 + v5) = 0;
      return v3;
    }
    *((_BYTE *)v3 + v5) = 0;
  }
}
```


如果继续看每个子函数会发现很难分析得清楚，所以我们现在换种思路用动态调试的方法观察这个会发生什么，直接用Ida动态调试。在sub_4021E0前面设置断点。可以发现只是将值赋给v41，这只是一个赋值函数。
现在我们还有一个问题没有解决v44是什么？
我们可以打开Strings window看到

```
.rdata:00… 00000009     C     bad cast
.rdata:00… 00000029     C     C:\\Users\\CSAW2016\\haha\\flag_dir\\flag.txt
.rdata:00… 00000016     C     ?W?h?a?t h?a?p?p?e?n?
.rdata:00… 00000021     C     |————————————————————|
```

这个路径我们好像从来都没有用到，点击交叉引用看一下。

他是在sub_402550中被用到，在结合sub_401620的代码可以看到它里面调用了sub_402550，所以v44就是一个从文件夹读出的内容。

最后看一下sub_4020C0

```
  v5 = a3;
  if ( this[4] < a3 )
    v5 = this[4];
  if ( this[5] >= 0x10u )
    this = (_DWORD *)*this;
  v6 = a5;
  if ( v5 < a5 )
    v6 = v5;
  if ( v6 )
  {
    v7 = a4;
    v9 = v6 < 4;
    v8 = v6 - 4;
    if ( v9 )
    {
LABEL_11:
      if ( v8 == -4 )
        goto LABEL_20;
    }
    else
    {
      while ( *this == *(_DWORD *)v7 )
      {
        ++this;
        v7 += 4;
        v9 = v8 < 4;
        v8 -= 4;
        if ( v9 )
          goto LABEL_11;
      }
    }
    v9 = *(_BYTE *)this < *(_BYTE *)v7;
    if ( *(_BYTE *)this != *(_BYTE *)v7
      || v8 != -3
      && ((v10 = *((_BYTE *)this + 1), v9 = v10 < *(_BYTE *)(v7 + 1), v10 != *(_BYTE *)(v7 + 1))
       || v8 != -2
       && ((v11 = *((_BYTE *)this + 2), v9 = v11 < *(_BYTE *)(v7 + 2), v11 != *(_BYTE *)(v7 + 2))
        || v8 != -1 && (v12 = *((_BYTE *)this + 3), v9 = v12 < *(_BYTE *)(v7 + 3), v12 != *(_BYTE *)(v7 + 3))))
)
    {
      result = -v9 | 1;
      goto LABEL_21;
    }
LABEL_20:
    result = 0;
LABEL_21:
    if ( result )
      return result;
  }
  if ( v5 >= a5 )
    result = v5 != a5;
  else
    result = -1;
  return result;
```

可以分析出是判断相不相等的问题。
综上就可以写出脚本得到flag

```
str1 = "themidathemidathemida"
str2 = ">----++++....<<<<."

key =""
flag=""
for i in range(18):
 key += chr((ord(str1[i]) ^ ord(str2[i]))+22)
for i in key:
 flag+=chr(ord(i)+9)


print(flag)
```