bugkuCTF平台逆向题第五道love题解



Megrez_ 于 2019-03-26 14:15:31 发布 470 个 收藏

分类专栏: ctf

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/gg 43445167/article/details/88818579

版权



ctf 专栏收录该内容

3篇文章0订阅

订阅专栏

看了看网上的writeup 这尼玛都不是人的思维,自己写个writeup

- 1. 首先 第一件事 看标题 标题一般有提示 对吧。 然而看不出个卵。
- 2. 运行一波 嗯哼 盲猜调strcmp进行比对,flag就会入栈,打开od,开始疯狂寻找

```
8895 54FFFFFF
888C15 6CFFFFFF
EB C0
8D85 6CFFFFFF

BOY edx,dword ptr ss:[ebp-8xAC]

jmp short reverse_.002F57D0
lea eax,dword ptr ss:[ebp-8x94].e1
002F5801
902F5807
02F580E
02F5810
302F5816
002F5817
             F8 ACR8FFFF
                                      11 reverse_.002F10C8
                                   add esp,0x4
mov esi,esp
002F581C
             83C4 04
102F581F
             8BF4
02F5821
02F5822
              68 34A02F00
                                         reverse_.002FA034
                                                                                          ASCII "e3nifIH9b_C@n@dH"
102F5827
             8D8D 6CFFFFF
                                   lea ecx, dword ptr ss:[ebp-0x94]
302F582D
              FF15 84B12F00
                                                                                          ucrtbase.strncmp
                                   add esp,0xC
                                                                                                                                                https://blog.csdn.net/qq_43445167
102F5837
```

- 3. 卧槽 上面那串奇异的字符一定是flag。提交 你懂的
- 4. 开始思考 我发现我输入的字符串没有入栈,这其中一定有蹊跷!

```
002F5810
           8D85 6CFFFFFF
                           lea eax, dword ptr ss:[ebp-0x94]
002F5816
           50
002F5817
           E8 ACB8FFFF
                                reverse_.002F10C8
002F581C
           83C4 04
                            add esp,0x4
002F581F
           8BF4
                            mov esi,esp
002F5821
           50
002F5822
           68 <u>34A02F00</u>
                            push reverse_.002FA034
                                                                      ASCII "e3nifIH9b_C@n@dH"
002F5827
            8D8D 6CFFFFFF
                            lea ecx, dword ptr ss:[ebp-0x94]
002F582D
           FF15 84B12F00
                                dword ptr ds:[<&ucrtbased.strncmp>
                                                                      ucrtbase.strncmp
                            add esp,0xC
002F5834
           83C4 OC
002F5837
           3BF4
                            cmp esi,esp
           E8 E9B8FFFF
                                reverse_.002F1127
002F583E
           85C0
002F5840
           74 OF
                            je short reverse__002F5851
                                                                      这个要跳
Stack address=00F9FD28, (ASCII "MUIzNDU2")
ecx=00000<u>0</u>55
                                                                                                 https://blog.csdn.net/qq_43445167
```

5. 然后我选择去找输入函数,跟踪我输入的字符串。

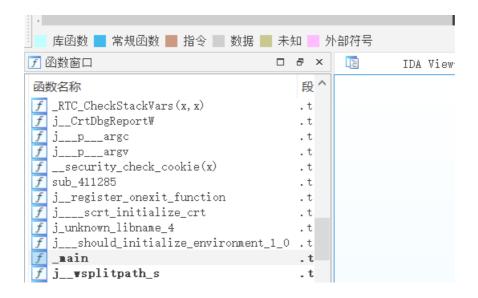
```
E8 02BCFFFF
                              reverse_.002F1375
                                                                     输入函数
                         add esp,0x8
2F5773
        83C4 08
2F5776
        8BF4
                         mov esi,esp
        6A 28
2F5778
                         nush 8x2
        8D45 F4
2F577A
                         lea eax, dword ptr ss:[ebp-0xC]
2F577D
                         <u>oush</u> eax
        50
2F577E
                         lea ecx,dword ptr ss:[ebp-0x28]
        8D4D D8
2F5781
2F5782
        E8 41B9FFFF
                             reverse_.002F10C8
                                                                     极可能为加密函数
2F5787
        8304 04
                         add esp,0x4
2F578A
        50
                              eax
        8D55 D8
2F578B
                         lea edx, dword ptr ss:[ebp-0x28]
2F578E
        52
                         push edx
        E8 2AB9FFFF
2F578F
                              reverse_.002F10BE
                         add esp,0x0
        83C4 OC
2F5794
F5797
        50
                         push eax
        8D85 6CFFFFFF
2F5798
                         lea eax, dword ptr ss:[ebp-0x94]
F579E
        50
                         push eax
 =002BF8BC, <mark>(ASCII "12314560")</mark>
```

6. 如图 我发现我的字符串出现了 我就觉得它对我的字符串动了手脚 事实证明...我错了 是它的下一个函数对我的字符串动了手脚...(我就是f8看出来的)

```
002F5778
002F577A
          8D45 F4
                         lea eax,dword ptr ss:[ebp-0xC]
002F577D
          50
                          ush eax
002F577E
          8D4D D8
                         lea ecx,dword ptr ss:[ebp-0x28]
002F5781
          51
                                                               极可能为加密函数
          E8 41B9FFFF
                         all reverse_.002F10C8
002F5782
002F5787
          83C4 04
                         add esp,0x4
002F578A
          50
002F578B
                         iea edx,dword ptr ss:[ebp-0x28]
          8D55 D8
002F578E
                             edx
002F578F
          E8 2AB9FFFF
                          all reverse_.002F10BE
002F5794
          83C4 OC
                         add esp, 0x0
002F5797
          50
002F5798
          8D85 6CFFFFFF
                         lea eax, dword ptr ss:[ebp-0x94]
002F579E
          50
                         002F579F
          FF15 88812E NO
          83<u>64</u> 0C
                         add esp, 0xC
002F57A5
eax=027EAFA0 (ASCII "MTIzMTQ1NjA=")
Address Hex dump
                                                     ASCII
```

7.

8. 这个字符串与图2中的字符串像的一匹,尼玛脉络出来了。题目将输入的flag加密后与另一个字符串比较。开IDA! 逆向算



```
f sub_4112CB
                                        .t
   _guard_check_icall(x)
                                        .t
   j___report_gsfailure
                                        .t
f j_exit
                                        .t
f j__initterm
                                        .t
f j__configure_narrow_argv
                                        .t
f j___setusermatherr
                                        .t
f j___vcrt_GetModuleFileNameW
                                        .t
f j_unknown_libname_3
                                        .t
f j_IsProcessorFeaturePresent
                                        .t
f sub_411302
                                        .t
f j__sprintf_s
                                        .t
f j___scrt_release_startup_lock
                                        .t
f j___scrt_acquire_startup_lock
                                       .t
f j__crt_atexit
                                        .t
  _RTC_GetSrcLine(uchar *, wchar_t *, u…
                                       .t 🗸
                                        >
<
行 94/274
ౣ 图表概览
                                    □ & ×
```

法! (先找主函数)

9. 直接f5看伪码,往里跟

```
救据 ■ 未知 ■ 外部符号
     □ & × [3
                                                     伪代码
                      IDA View-A
                                 ×
                                                                  ×
                                                                      0
                                                                             十六进制视图-1 🛛 🛕
                                                                                                              结
                 2{
         段 ^
                 3 int v0; // eax
        .t
                 4 const char *v1; // eax
        .t
                 5
                   size_t v2; // eax
        .t
                    int v3; // edx
        .t
                     __int64 v4; // ST08_8
        .t
                 8 signed int j; // [esp+DCh] [ebp-ACh]
        .t
                   signed int i; // [esp+E8h] [ebp-A0h]
        .t
                10 signed int v8; // [esp+E8h] [ebp-A0h]
        .t
                   char Dest[108]; // [esp+F4h] [ebp-94h]
        .t
                11
ment_1_0 .t
                   char Str; // [esp+160h] [ebp-28h]
char v11; // [esp+17Ch] [ebp-Ch]
                12
        .t
                13
        .t
                14
        .t
              15
                    for (i = 0; i < 100; ++i)
        .t
                16
                    {
        .t
              17
                     if (i >= 0x64)
        . t
              18
                            _report_rangecheckfailure();
        .t
              19
                      Dest[i] = 0;
        .t
               20
        .t
              21
                    sub_41132F("please enter the flag:");
        .t
              22
                    sub_411375("%20s", &Str);
        .t
              23  v0 = j_strlen(&Str);
24  v1 = sub_4110BE(&Str, v0, &v11);
nt
        .t
              25
                    strnepy(Dest, v1, 0x28u);
        .t
              26
                    v8 = j_strlen(Dest);
        .t
              27
                    for ( j = 0; j < v8; ++j )
οk
        .t
              28
                     Dest[j] += j;
        .t
:_t *,u… .t ✔ | • 29
                    v2 = j_strlen(Dest);
              30
                   if (!strncmp(Dest, Str2, v2))
        >
              31
                     sub_41132F("rigth flag!\n");
                32
                    else
     □ & × | • 33
                      sub_41132F("wrong flag!\n");
              34
                   HIDWORD(v4) = v3;
              35
                   LODWORD(\vee 4) = 0;
              36
                   return v4;
              37 }
                  00004AE0 _main_0:2 (4156E0)
```

10.自己分析一边就知道这个函数有问题了,往里跟就能看到加密函数。 然后 算法逆向不出来。。。。。。。。。

去看writeup了,base64,气得我吐血。就当熟悉base64的加密过程了。

