攻防世界 Reverse高手进阶区 2分题 Replace

原创

思源湖的鱼 于 2020-12-21 15:27:10 发布 248 收藏

分类专栏： ctf 文章标签： ctf 攻防世界 reverse upx

本文链接：https://blog.csdn.net/weixin_44604541/article/details/111474552

ctf 专栏收录该内容

200 篇文章 23 订阅

订阅专栏

前言

继续ctf的旅程

攻防世界Reverse高手进阶区的2分题

本篇是Replace的writeup

发现攻防世界的题目分数是动态的

就仅以做题时的分数为准了

解题过程

PE查壳

有个upx壳

upxshell脱壳

扔进IDA

```
1  int __cdecl main(int argc, const char **argv, const char **envp)
2  {
3    char Buf; // [esp+4h] [ebp-2Ch]
4    char Dst; // [esp+5h] [ebp-2Bh]
5
6    Buf = 0;
7    memset(&Dst, 0, 0x27u);
8    printf("Welcome The System\nPlease Input Key:");
9    gets_s(&Buf, 0x28u);
10   if ( strlen(&Buf) - 35 <= 2 )
11   {
12     if ( sub_401090(&Buf) == 1 )
13       printf("Well Done!\n");
14     else
15       printf("Your Wrong!\n");
16   }
17   return 0;
18 }
```



```
1  signed int __fastcall sub_401090(int a1, int a2)
2  {
3    int v2; // ebx
4    int v4; // edx
5    char v5; // al
6    int v6; // esi
7    int v7; // edi
8    char v8; // al
9    int v9; // eax
10   char v10; // cl
11   int v11; // eax
12   int v12; // ecx
13
14   v2 = a1;
15   if ( a2 != 35 )
16     return -1;
17   v4 = 0;
18   while ( 1 )
19   {
20     v5 = *(_BYTE *)(v4 + v2);
21     v6 = (v5 >> 4) % 16;
22     v7 = (16 * v5 >> 4) % 16;
23     v8 = byte_402150[2 * v4];
24     if ( v8 < 48 || v8 > 57 )
25       v9 = v8 - 87;
26     else
27       v9 = v8 - 48;
28     v10 = byte_402151[2 * v4];
29     v11 = 16 * v9;
30     if ( v10 < 48 || v10 > 57 )
31       v12 = v10 - 87;
32     else
33       v12 = v10 - 48;
34     if ( (unsigned __int8)byte_4021A0[16 * v6 + v7] != ((v11 + v12) ^ 0x19) )
35       break;
36     if ( ++v4 >= 35 )
37       return 1;
38   }
39   return -1;
40 }
```



```
.rdata:00402150 ; char byte_402150[]
.rdata:00402150 byte_402150     db 32h                  ; DATA XREF: sub_401090:loc_4010CC↑r
.rdata:00402151 ; char byte_402151[]
.rdata:00402151 byte_402151     db 61h                  ; DATA XREF: sub_401090:loc_4010E9↑r
.rdata:00402152 a49f69c38395cde db '49f69c38395cde96d6de96d6f4e025484954d6195448def6e2dad67786e21d5ad'
.rdata:00402152                 db 'ae6',0
.rdata:00402197                 align 10h
.rdata:004021A0 ; char byte_4021A0[256]
```

```
.rdata:004021A0 byte_4021A0     db 63h                  ; DATA XREF: sub_401090+82↑r
.rdata:004021A1                 db  7Ch ; |
.rdata:004021A2                 db  77h ; w
.rdata:004021A3                 db  7Bh ; {
.rdata:004021A4                 db 0F2h
.rdata:004021A5                 db  6Bh ; k
.rdata:004021A6                 db  6Fh ; o
```

```
.rdata:004021A8                 db  30h ; 0
.rdata:004021A9                 db    1
.rdata:004021AA                 db  67h ; g
.rdata:004021AB                 db  2Bh ; +
.rdata:004021AC                 db 0FEh
.rdata:004021AD                 db 0D7h
.rdata:004021AE                 db 0ABh
.rdata:004021AF                 db  76h ; v
.rdata:004021B0                 db 0CAh
.rdata:004021B1                 db  82h
.rdata:004021B2                 db 0C9h
.rdata:004021B3                 db  7Dh ; }
.rdata:004021B4                 db 0FAh
.rdata:004021B5                 db  59h ; Y
.rdata:004021B6                 db  47h ; G
.rdata:004021B7                 db 0F0h
.rdata:004021B8                 db 0ADh
.rdata:004021B9                 db 0D4h
.rdata:004021BA                 db 0A2h
.rdata:004021BB                 db 0AFh
.rdata:004021BC                 db  9Ch
.rdata:004021BD                 db 0A4h
.rdata:004021BE                 db  72h ; r
.rdata:004021BF                 db 0C0h
.rdata:004021C0                 db 0B7h
.rdata:004021C1                 db 0FDh
.rdata:004021C2                 db  93h
.rdata:004021C3                 db  26h ; &
.rdata:004021C4                 db  36h ; 6
.rdata:004021C5                 db  3Fh ; ?
.rdata:004021C6                 db 0F7h
.rdata:004021C7                 db 0CCh
.rdata:004021C8                 db  34h ; 4
.rdata:004021C9                 db 0A5h
.rdata:004021CA                 db 0E5h
```


根据逻辑得到脚本

```
a="2a49f69c38395cde96d6de96d6f4e025484954d6195448def6e2dad67786e21d5adae6"
b="a49f69c38395cde96d6de96d6f4e025484954d6195448def6e2dad67786e21d5adae6"
c=[0x63 ,0x7c ,0x77 ,0x7b ,0xf2 ,0x6b ,0x6f ,0xc5 ,0x30 ,0x1 ,0x67 ,0x2b ,0xfe ,0xd7 ,0xab ,0x76 ,0xca ,0x82 ,0x
c9 ,0x7d ,0xfa ,0x59 ,0x47 ,0xf0 ,0xad ,0xd4 ,0xa2 ,0xaf ,0x9c ,0xa4 ,0x72 ,0xc0 ,0xb7 ,0xfd ,0x93 ,0x26 ,0x36 ,
0x3f ,0xf7 ,0xcc ,0x34 ,0xa5 ,0xe5 ,0xf1 ,0x71 ,0xd8 ,0x31 ,0x15 ,0x4 ,0xc7 ,0x23 ,0xc3 ,0x18 ,0x96 ,0x5 ,0x9a ,
0x7 ,0x12 ,0x80 ,0xe2 ,0xeb ,0x27 ,0xb2 ,0x75 ,0x9 ,0x83 ,0x2c ,0x1a ,0x1b ,0x6e ,0x5a ,0xa0 ,0x52 ,0x3b ,0xd6 ,
0xb3 ,0x29 ,0xe3 ,0x2f ,0x84 ,0x53 ,0xd1 ,0x0 ,0xed ,0x20 ,0xfc ,0xb1 ,0x5b ,0x6a ,0xcb ,0xbe ,0x39 ,0x4a ,0x4c
,0x58 ,0xcf ,0xd0 ,0xef ,0xaa ,0xfb ,0x43 ,0x4d ,0x33 ,0x85 ,0x45 ,0xf9 ,0x2 ,0x7f ,0x50 ,0x3c ,0x9f ,0xa8 ,0x51
 ,0xa3 ,0x40 ,0x8f ,0x92 ,0x9d ,0x38 ,0xf5 ,0xbc ,0xb6 ,0xda ,0x21 ,0x10 ,0xff ,0xf3 ,0xd2 ,0xcd ,0xc ,0x13 ,0xe
c ,0x5f ,0x97 ,0x44 ,0x17 ,0xc4 ,0xa7 ,0x7e ,0x3d ,0x64 ,0x5d ,0x19 ,0x73 ,0x60 ,0x81 ,0x4f ,0xdc ,0x22 ,0x2a ,0
x90 ,0x88 ,0x46 ,0xee ,0xb8 ,0x14 ,0xde ,0x5e ,0xb ,0xdb ,0xe0 ,0x32 ,0x3a ,0xa ,0x49 ,0x6 ,0x24 ,0x5c ,0xc2 ,0x
d3 ,0xac ,0x62 ,0x91 ,0x95 ,0xe4 ,0x79 ,0xe7 ,0xc8 ,0x37 ,0x6d ,0x8d ,0xd5 ,0x4e ,0xa9 ,0x6c ,0x56 ,0xf4 ,0xea ,
0x65 ,0x7a ,0xae ,0x8 ,0xba ,0x78 ,0x25 ,0x2e ,0x1c ,0xa6 ,0xb4 ,0xc6 ,0xe8 ,0xdd ,0x74 ,0x1f ,0x4b ,0xbd ,0x8b
,0x8a ,0x70 ,0x3e ,0xb5 ,0x66 ,0x48 ,0x3 ,0xf6 ,0xe ,0x61 ,0x35 ,0x57 ,0xb9 ,0x86 ,0xc1 ,0x1d ,0x9e ,0xe1 ,0xf8
,0x98 ,0x11 ,0x69 ,0xd9 ,0x8e ,0x94 ,0x9b ,0x1e ,0x87 ,0xe9 ,0xce ,0x55 ,0x28 ,0xdf ,0x8c ,0xa1 ,0x89 ,0xd ,0xbf
 ,0xe6 ,0x42 ,0x68 ,0x41 ,0x99 ,0x2d ,0xf ,0xb0 ,0x54 ,0xbb ,0x16]
flag = []

v4 = 0
while True:
 v8 = ord(a[2 * v4])
 if v8 < 48 or v8 > 57:
  v9 = v8 - 87
 else:
  v9 = v8 - 48
 v10 = ord(b[2 * v4])
 v11 = 16 * v9
 if v10 < 48 or v10 > 57:
  v12 = v10 - 87
 else:
  v12 = v10 - 48
 for v5 in range(32, 127):
  v6 = (v5 >> 4) % 16
  v7 = (16 * v5 >> 4) % 16
  if c[16 * v6 + v7] == ((v11 + v12) ^ 0x19):
   flag.append(chr(v5))
   break
 v4 += 1
 if v4 >= 35: break
print("".join(flag))
```

```
    v8 = ord(a[2 * v4])
    if v8 < 48 or v8 > 57:
        v9 = v8 - 87
    else:
        v9 = v8 - 48
    v10 = ord(b[2 * v4])
    v11 = 16 * v9
    if v10 < 48 or v10 > 57:
        v12 = v10 - 87
    else:
        v12 = v10 - 48
    for v5 in range(32, 127):
        v6 = (v5 >> 4) % 16
        v7 = (16 * v5 >> 4) % 16
        if c[16 * v6 + v7] == ((v11 + v12) ^ 0x19):
            flag.append(chr(v5))
            break
    v4 += 1
    if v4 >= 35: break
print("".join(flag))
```

```
flag{Th1s_1s_Simple_Rep1ac3_Enc0d3}
```


得到flag

结语

脱壳

简单逻辑