

buuctf 逆向 xor

原创

菜逼的ctf之路 于 2020-10-05 21:58:34 发布 2011 收藏 5

版权声明：本文为博主原创文章，遵循 [CC 4.0 BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) 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/weixin_45701079/article/details/108933456

版权

buuctf 逆向 xor

题逻辑比较简单，但是中间有点小插曲，等会再说，先打开软件，没壳(我查过了),然后打开ida，如图

```
IDA View-A | Pseudocode-A | Hex View-1 | Structures | Enums
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     char *v3; // rsi
4     int result; // eax
5     signed int i; // [rsp+2Ch] [rbp-124h]
6     char v6[264]; // [rsp+40h] [rbp-110h]
7     __int64 v7; // [rsp+148h] [rbp-8h]
8
9     memset(v6, 0, 0x100uLL);
10    v3 = (char *)256;
11    printf("Input your flag:\n", 0LL);
12    get_line(v6, 256LL);
13    if ( strlen(v6) != 33 )
14        goto LABEL_12;
15    for ( i = 1; i < 33; ++i )
16        v6[i] ^= v6[i - 1];
17    v3 = global;
18    if ( !strncmp(v6, global, 0x21uLL) )
19        printf("Success", v3);
20    else
21 LABEL_12:
22    printf("Failed", v3);
23    result = __stack_chk_guard;
24    if ( __stack_chk_guard == v7 )
25        result = 0;
26    return result;
27 }
```

https://blog.csdn.net/weixin_45701079

逻辑很简单，就是把下一个和之前的一个异或，然后保存。

然后比较坑的地方就来了。

点开global，出现如图所示

```
__cstring:00000000100000F6E ;org 100000F6Eh
__cstring:00000000100000F6E byte_100000F6E db 66h
__cstring:00000000100000F6F db 0Ah
__cstring:00000000100000F70 db 6Bh ; k
__cstring:00000000100000F71 db 0Ch
__cstring:00000000100000F72 db 77h ; w
__cstring:00000000100000F73 db 26h ; &
__cstring:00000000100000F74 db 4Fh ; 0
__cstring:00000000100000F75 db 2Eh ; .
__cstring:00000000100000F76 db 40h ; @
__cstring:00000000100000F77 db 11h
__cstring:00000000100000F78 db 78h ; x
__cstring:00000000100000F79 db 0Dh
__cstring:00000000100000F7A db 5Ah ; Z
__cstring:00000000100000F7B db 3Bh ; ;
```

```

cstring:0000000100000F7C      db  55h ; Ú
cstring:0000000100000F7D      db  11h
cstring:0000000100000F7E      db  70h ; p
cstring:0000000100000F7F      db  19h
cstring:0000000100000F80      db  46h ; F
cstring:0000000100000F81      db  1Fh
cstring:0000000100000F82      db  76h ; v
cstring:0000000100000F83      db  22h ; "
cstring:0000000100000F84      db  4Dh ; M
cstring:0000000100000F85      db  23h ; #
cstring:0000000100000F86      db  44h ; D
cstring:0000000100000F87      db  0Eh
cstring:0000000100000F88      db  67h ; g
cstring:0000000100000F89      db   6
cstring:0000000100000F8A      db  68h ; h
cstring:0000000100000F8B      db  0Fh
cstring:0000000100000F8C      db  47h ; G
cstring:0000000100000F8D      db  32h ; 2

```

这里的数据比较奇怪了，既有数据，又有字符。所以不要单纯的复制下来，要转换为ascii码，然后进行异或，最后贴上脚本。

```

a= [0x66, 0x0A, 0x6B, 0x0C, 0x77, 0x26,
    0x4F, 0x2E, 0x40, 0x11, 0x78, 0x0D,
    0x5A, 0x3B, 0x55, 0x11,
    0x70, 0x19, 0x46, 0x1F, 0x76, 0x22,
    0x4D, 0x23, 0x44, 0x0E, 0x67,
    0x06, 0x68, 0x0F, 0x47, 0x32,0x4F]

s='f'
s+='f'
for i in range(1,len(a)):
    s+=chr(a[i]^a[i-1])
print(s)

```

因为第一个是f，所以要提前加上去，之后运行就可以得到flag

最后，想更新一些《逆向工程核心原理》这本书的学习笔记，因为最近课有点多，所以今天就先划水了，就做了这一道题。之后几天再补上关于《逆向工程核心原理》这本书上的学习笔记