

第七届山东省大学生网络安全技能大赛(科来杯)逆向WP

原创

北风~ 于 2021-03-14 21:42:39 发布 628 收藏

分类专栏: [CTF 逆向与保护](#)

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/weixin_45055269/article/details/114576865

版权



[CTF 同时被 2 个专栏收录](#)

31 篇文章 3 订阅

订阅专栏



[逆向与保护](#)

65 篇文章 4 订阅

订阅专栏

file

出题人有个不好的习惯，总喜欢清除回收站，这次在出题的过程中又把文件给删了，你能帮他还原回来吗？flag无标准格式,提交答案请加上flag{}

elf 静态分析

```

12
13 v12 = __readfsqword(0x28u);
14 v8 = fopen(argv[1], "r");
15 if ( !v8 )
16     __assert_fail("file!=NULL", "file.c", 0x8Fu, "main");
17 for ( i = 0; (unsigned int)__isoc99_fscanf(v8, "%c", (char *)&v11[64] + i) != -1; ++i )
18     ;
19 v5 = 0;
20 v9 = (_BYTE *)encode(flllag);
21 v10 = (_BYTE *)encode(sttr_home);
22 for ( j = 0; j <= 63; j += 2 )
23     v11[v5++] = sub_123887656((unsigned int)sttr_home[j], (unsigned int)sttr_home[j + 1]);
24 *v9 = encode(flllag);
25 for ( k = 0; k < v5; ++k )
26     {
27         if ( *flllag != (k ^ *((char *)&v11[64] + k) ^ v11[k]) )
28             {
29                 printf("Your file is wrong!! try again");
30                 return 0;
31             }
32         ++flllag;
33     }
34 *v10 = encode(sttr_home);

```

核心是异或操作，flllag虽然看着被encode加密了，但*flllag地址里的值还是没变，这点小坑，接着就是写脚本，将文件作为参数带入就可以了，v11动调可拿

```

v11=[0x66,0x4e,0x6,0x22,0x66,0x25,0x42,0x5d,0x56,0x2e,0x76,0x6e,0x4,0x2d,0x42,0x2c,0x7,0x2c,0x45,0x69,0x2d,0x12,
0x5c,0x7e,0x65,0x52,0x60,0x69,0x54,0x64,0x66,0x43]
flllag=bytearray(b'flag{hello_player_come_on_hahah}')
flag=[0]*32
for i in range(0x20):
    flag[i]=hex(i^flllag[i]^v11[i])
print(' '.join(flag))

```

winhex打开，将hex值一个一个写入就可了

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	ANSI ASCII
00	23	65	46	19	48	21	36	32	48	23	15	64	41	35	46	#eF H!62H# dA5F
65	62	34	15	54	62	15	06	13	14	12	13	20	18	10	21	eb4 Tb !

flag是将文件的md5

914a7b9df69eab5b74b9edb7070e53e8

flag{914a7b9df69eab5b74b9edb7070e53e8}

not_only_smc

所见非所得

smc+花指令

DIE查壳后发现UPX壳，esp手脱，修复导入表后，还是打不开，只能带壳分析，在VirtualAlloc下断，F9运行，直到停到输入，点击EIP找到输入点的位置下断,输入字符串，这个字符串是随便输入，目的是找到输入字符串在内存的位置

773C7FA1	90	nop
773C7FA2	90	nop
773C7FA3	90	nop

773C7FA3	B8 01200000	mov eax,0x2001
773C7FA8	B9 00000000	mov ecx,0x0
773C7FAD	8D5424 04	lea edx,dword ptr ss:[esp+0x4]
773C7FB1	64:FF15 C0000000	call dword ptr fs:[0xC0]
EIP → 773C7FB8	83C4 04	add esp,0x4
773C7FBB	C2 2000	ret 0x20
773C7FBE	90	nop
773C7FBF	90	nop
773C7FC0	90	nop
773C7FC1	90	nop
773C7FC2	90	nop
773C7FC3 <kernel32.C	B8 03200000	mov eax,0x2003
773C7FC8	B9 00000000	mov ecx,0x0
773C7FCD	8D5424 04	lea edx,dword ptr ss:[esp+0x4]
773C7FD1	64:FF15 C0000000	call dword ptr fs:[0xC0]
773C7FD8	83C4 04	add esp,0x4

esp=0030F1C4

.text:773C7FB8 kernel32.dll:\$B7FB8 #B7FB8

地址	十六进制	ASCII
10011BC0	66 6C 61 67 7B 72 65 31 6A 55 6E 6B 5F 77 6E 6E	lag{re1jUnk_wnn
10011BD0	65 65 64 61 67 69 72 6C 66 72 69 65 6E 64 7D 0D	eedagirlfriend}.
10011BE0	0A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10011BF0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

接着F9，程序会在VirtualAlloc停下，然后单步F8就能走到程序领空。

这还是一个smc，取输入的第8位的后五位为key进行异或解密，这怎么搞？key不知道，连真正的结果也异或不出。

10002DD0	8B85 1CFFFFFF	mov eax,dword ptr ss:[ebp-0xE4]
10002DD6	99	cdq
10002DD7	BE 05000000	mov esi,0x5
10002DDC	F7FE	idiv esi
10002DDE	0FBE5415 D8	movsx edx,byte ptr ss:[ebp+edx-0x28]
EIP → 10002DE3	33CA	xor ecx,edx
10002DE5	8B45 FC	mov eax,dword ptr ss:[ebp-0x4]
10002DE8	0385 1CFFFFFF	add eax,dword ptr ss:[ebp-0xE4]
10002DEE	8808	mov byte ptr ds:[eax],cl
10002DF0	EB B8	jmp 0x10002DAA
10002DF2	8D8D 70FFFFFF	lea ecx,dword ptr ss:[ebp-0x90]
10002DF8	894D EC	mov dword ptr ss:[ebp-0x14],ecx
10002DFB	6A 20	push 0x20
10002DFD	FF75 EC	push dword ptr ss:[ebp-0x14]

ecx=3F '?'

edx=6A 'j'

10002DE3

地址	十六进制	ASCII
0030F6F0	6A 55 6E 6B 5F 00 00 00 2B 9F 18 A3 D1 01 00 00	jUnk_...+..eÑ...
0030F700	01 00 00 00 18 F7 30 00 01 00 00 00 88 DF 18 00÷0.....ß..
0030F710	20 00 00 00 00 00 1C 00 58 F7 30 00 DF 34 00 10X÷0..ß4..
0030F720	00 00 00 10 01 00 00 00 00 00 00 00 6B 9E 18 A3k..f

于是在下面看到了，他去call这个位置

10002DEE	8808	mov byte ptr ds:[eax],cl
10002DF0	EB B8	jmp 0x10002DAA
10002DF2	8D8D 70FFFFFF	lea ecx,dword ptr ss:[ebp-0x90]
10002DF8	894D EC	mov dword ptr ss:[ebp-0x14],ecx
10002DFB	6A 20	push 0x20
10002DFD	FF75 EC	push dword ptr ss:[ebp-0x14]
10002E00	6A 38	push 0x38
10002E02	8B45 FC	mov eax,dword ptr ss:[ebp-0x4]
10002E05	FFD0	call eax
10002E07	8B95 3CFFFFFF	mov edx,dword ptr ss:[ebp-0xC4]

10002E0D	3B95 34FFFFFF	cmp edx,dword ptr ss:[ebp-0xCC]
10002E13	76 13	jbe 0x10002E28
10002E15	8D85 20FFFFFF	lea eax,dword ptr ss:[ebp-0xE0]
10002E1B	05 1BEDFFFF	add eax,0xFFFFED1B
10002E20	8985 F0FEFFFF	mov dword ptr ss:[ebp-0x110],eax
10002E26	EB 1C	jmp 0x10002E44
10002E28	C785 48FFFFFF 45520000	mov dword ptr ss:[ebp-0xB8],0x5245
10002E32	8B45 F4	mov eax,dword ptr ss:[ebp-0xC]

由于这种函数的头部的硬编码固定，所以可以异或出真正的key

```

tmp=[0x55,0x8B,0xEC,0x81,0xEC]
tmp0=[0x3f,0xde,0x82,0xea,0xb3]
flag=[0]*5
for i in range(5):
    flag[i]=tmp[i]^tmp0[i]
print(''.join(map(chr,flag)))
#jUnk_

```

所以接着拼凑测试flag，flag{re1jUnk_wnneedagirlfriend}，接着输入，遇到了大片的花指令

The screenshot shows a debugger interface with two main windows. The top window displays assembly code with addresses, hex values, and mnemonics. The bottom window shows a memory dump with columns for address, hexadecimal values, and ASCII characters. The ASCII column shows the string 'flag{re1jUnk_wnneedagirlfriend}.'

可以去花，或者这么调试也可，在字符串的内存位置下硬件断点，F9结合F2下断动调即可，是个体力活

001C0182	EE	out dx,al
001C0183	EA 4C588B4D FC0F	jmp far 0xFFC:0x4D8B584C
001C018A	B6 54	mov dh,0x54
001C018C	0D C88B450C	or eax,0xC458BC8
001C0191	0345 C0	add eax,dword ptr ss:[ebp-0x40]
001C0194	0FB608	movzx ecx,byte ptr ds:[eax]
001C0197	33CA	xor ecx,edx
001C0199	8B55 0C	mov edx,dword ptr ss:[ebp+0xC]
001C019C	0355 C0	add edx,dword ptr ss:[ebp-0x40]
001C019F	880A	mov byte ptr ds:[edx],cl
001C01A1	EB 14	jmp 0x1C01B7
001C01A3	EA 50EB0BEA 8BC4	jmp far 0xC48B:0xEA0BE850

001C01A9	EA 90E00EA 00C1	jmp far 0x2444:0xD60F66A2
001C01AA	A8 01	test al,0x1
001C01AC	74 06	je 0x1C01B4
001C01AE	EB 0B	jmp 0x1C01BB
001C01B0	EA A2660FD6 4424	jmp far 0x2444:0xD60F66A2
001C01B7	EB EB	jmp 0x1C01A4
001C01B9	EE	out dx,al

以及换位异或，好难的

1000114E	7D 26	jge 0x10001176
10001150	8B55 0C	mov edx,dword ptr ss:[ebp+0xC]
10001153	0355 9C	add edx,dword ptr ss:[ebp-0x64]
10001156	8B45 08	mov eax,dword ptr ss:[ebp+0x8]
10001159	0345 9C	add eax,dword ptr ss:[ebp-0x64]
1000115C	0FB608	movzx ecx,byte ptr ds:[eax]
1000115F	8B45 08	mov eax,dword ptr ss:[ebp+0x8]
10001162	0FB61410	movzx edx,byte ptr ds:[eax+edx]
EIP → 10001166	33D1	xor edx,ecx
10001168	8B45 0C	mov eax,dword ptr ss:[ebp+0xC]
1000116B	0345 9C	add eax,dword ptr ss:[ebp-0x64]
1000116E	8B4D 08	mov ecx,dword ptr ss:[ebp+0x8]
10001171	881401	mov byte ptr ds:[ecx+eax],dl
10001174	EB C9	jmp 0x1000113F
10001176	8D55 1C	lea edx,dword ptr ss:[ebp+0x1C]
10001179	0FB7C2	movzx eax,dx
1000117C	8D4D A4	lea ecx,dword ptr ss:[ebp-0x5C]
1000117F	3BC1	cmp eax,ecx

```

ans = [0xE8, 0x90, 0x24, 0xE6, 0x0A, 0xE3, 0xF7, 0xA8, 0x09, 0xC0, 0x35, 0x74, 0x26, 0x4D, 0xA0, 0x2D,
       0xD6, 0x1A, 0x5A, 0x5C, 0x16, 0x2D, 0xF0, 0x46, 0x44, 0x10, 0x5F, 0x83, 0x5B, 0xBE, 0x86, 0xAC]
key = [0xBB, 0x35, 0xAF, 0x29, 0xA3, 0x1D, 0x97, 0x11, 0x8B, 0x05, 0x7F, 0xF9, 0x73, 0xED, 0x67, 0xE1,
       0x5B, 0xD5, 0x4F, 0xC9, 0x43, 0xBD, 0x37, 0xB1, 0x2B, 0xA5, 0x1F, 0x99, 0x13, 0x8D, 0x07, 0x81]
for i in range(65):
    j = 0x10
    while j > 0:
        for k in range(j):
            ans[j+k] = ans[j+k] ^ ans[k]
        j = j >> 1
for i in range(32):
    ans[i]^=key[i]
    print(chr(ans[i]),end='')
#SMc_AnD_jUnk_C0de_1s_s0_fuunn~!

```

flag{SMc_AnD_jUnk_C0de_1s_s0_fuunn~!}

[babyLoginPlus](#)

新品发布会。『babyLogin Plus』哪一面，都是亮(汇)点(编)

vm类的题还是没有有什么好的办法，找到输入，下硬件断点，跟就可

004011ED 90 nop
004011EE 90 nop
004011EF 90 nop
004011F0 <babyloginp . 8B4424 04 mov eax,dword ptr ss:[esp+0x4]
004011F4 . 8B48 04 mov ecx,dword ptr ds:[eax+0x4]
004011F7 . 8B10 mov edx,dword ptr ds:[eax]
EIP → 004011F9 . 2BD1 sub edx,ecx
004011FB . 8B48 14 mov ecx,dword ptr ds:[eax+0x14]
004011FE . 41 inc ecx
004011FF . 8910 mov dword ptr ds:[eax],edx
00401201 . 8948 14 mov dword ptr ds:[eax+0x14],ecx
00401204 . C3 ret
00401205 90 nop
00401206 90 nop
00401207 90 nop
00401208 90 nop
00401209 90 nop
0040120A 90 nop

edx=66 'f'
ecx=9 '\t'

.text:004011F9 babyloginplus.exe:\$11F9 #5F9 <sub_4011F0+9>

0040117A 90 nop
0040117B 90 nop
0040117C 90 nop
0040117D 90 nop
0040117E 90 nop
0040117F 90 nop
00401180 <babyloginp . 8B4424 04 mov eax,dword ptr ss:[esp+0x4]
00401184 . 8B48 04 mov ecx,dword ptr ds:[eax+0x4]
00401187 . 8B10 mov edx,dword ptr ds:[eax]
EIP → 00401189 . 33D1 xor edx,ecx
0040118B . 8B48 14 mov ecx,dword ptr ds:[eax+0x14]
0040118E . 41 inc ecx
0040118F . 8910 mov dword ptr ds:[eax],edx
00401191 . 8948 14 mov dword ptr ds:[eax+0x14],ecx
00401194 . C3 ret
00401195 90 nop
00401196 90 nop
00401197 90 nop

edx=5D ']'
ecx=26 '&'

.text:00401189 babyloginplus.exe:\$1189 #589 <sub_401180+9>

内存 1 内存 2 内存 3 内存 4 内存 5 监视 1 |x=| 局部变量 结构体

地址	十六进制	ASCII
0040B0B0	5D 00 00 00 26 00 00 00 00 00 00 00 00 00 00 00]...&.....
0040B0C0	00 00 00 00 68 85 40 00 D0 00 00 00 00 10 40 00	...h.@.@....@.

00401180 <babyloginp	. 8B4424 04	mov eax,dword ptr ss:[esp+0x4]
00401184	. 8B48 04	mov ecx,dword ptr ds:[eax+0x4]
00401187	. 8B10	mov edx,dword ptr ds:[eax]
EIP → 00401189	. 33D1	xor edx,ecx
0040118B	. 8B48 14	mov ecx,dword ptr ds:[eax+0x14]
0040118E	. 41	inc ecx
0040118F	. 8910	mov dword ptr ds:[eax],edx
00401191	. 8948 14	mov dword ptr ds:[eax+0x14],ecx
00401194	. C3	ret
00401195	. 90	nop
00401196	. 90	nop
00401197	. 90	nop
00401198	. 90	nop
00401199	. 90	nop
0040119A	. 90	nop
0040119B	. 90	nop
0040119C	. 90	nop
0040119D	. 90	nop

edx=7B '{'
ecx=57 'w'

.text:00401189 babyloginplus.exe:\$1189 #589 <sub_401180+9>

最后有个+0x6的操作

写脚本逆向

```

src = [0x32, 0x26, 0x18, 0x21, 0x41, 0x23, 0x2A, 0x57, 0x44, 0x29, 0x35, 0x12, 0x20, 0x17, 0x45, 0x1C,
       0x68, 0x2D, 0x7A, 0x79, 0x47, 0x7F, 0x44, 0x09, 0x1E, 0x75, 0x41, 0x2A, 0x19, 0x34, 0x76, 0x47,
       0x14, 0x50, 0x52, 0x76, 0x58]
key = [0x57, 0x65, 0x6C, 0x63, 0x6F, 0x6D, 0x65, 0x5F, 0x74, 0x6F, 0x5F, 0x73, 0x64, 0x6E, 0x69, 0x73,
       0x63, 0x5F, 0x32, 0x30, 0x31, 0x38, 0x5F, 0x42, 0x79, 0x2E, 0x5A, 0x65, 0x72, 0x6F, 0x00, 0x00,
       0x00, 0x00, 0x00, 0x00, 0x00]
flag = ''
for i in range(len(src)):
    flag += chr(((src[i]-0x6) ^ 0x26 ^ key[i])+9)
print(flag)
#flag{_p1us_babyL0gin_pPpPpPpPp_p1us_}

```