

【reversing.kr逆向之旅】Twist1的writeup

原创

iqiqiya 于 2019-01-19 16:24:54 发布 386 收藏

分类专栏: [我的逆向之路](#) [我的CTF之路](#) [-----reversing.kr](#) [我的CTF进阶之路](#) 文章标签: [【reversing.kr逆向之旅】Twist1的writeup](#) [逆向题](#)

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/xiangshangbashaonian/article/details/86553766>

版权



[我的逆向之路](#) 同时被 3 个专栏收录

108 篇文章 10 订阅

订阅专栏



[我的CTF之路](#)

92 篇文章 5 订阅

订阅专栏



[-----reversing.kr](#)

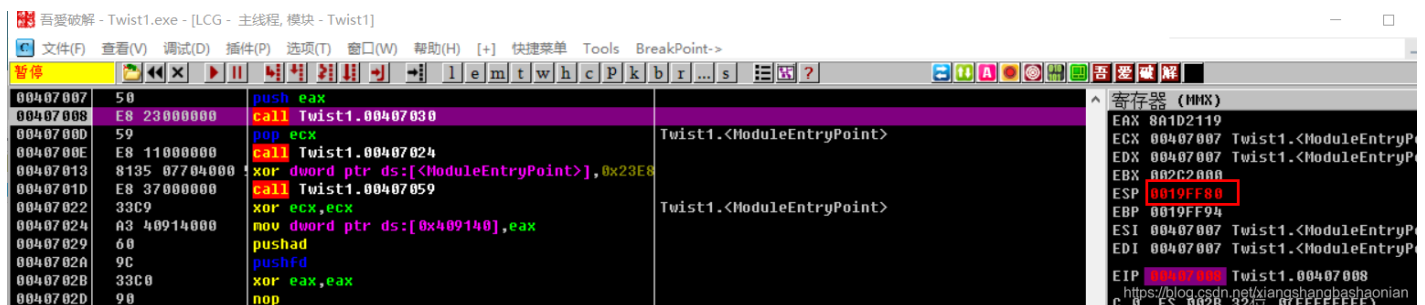
11 篇文章 0 订阅

订阅专栏

将程序载入OllyDbg



直接F8单步的话会直接停止运行 我们先可以单步一次使用ESP定律



接着单步一步一步走就可以 遇到向上的跳转 在他的下一行按F4就可以走到这里

如下图所示 将要跳向OEP

```
00407185 0000 add byte ptr ds:[eax],al
00407187 3D C0904000 cmp eax, Twist1.004090C0
0040718C ^ 75 F5 jnz short Twist1.00407183
0040718E 90 nop
0040718F B8 00004000 mov eax, Twist1.00400000
00407194 05 00100000 add eax, 0x1000
00407199 C008 03 ror byte ptr ds:[eax], 0x3
0040719C 40 inc eax
0040719D 3D 00704000 cmp eax, Twist1.00407000
004071A2 ^ 75 F5 jnz short Twist1.00407199
004071A4 90 nop
004071A5 C705 83714000 mov dword ptr ds:[0x407183], 0x0
004071AF C605 BD714000 mov byte ptr ds:[0x4071BD], 0xE9
004071B6 C605 BF714000 mov byte ptr ds:[0x4071BF], 0xA3
004071BD ^ E9 BAA3FFFF jmp Twist1.0040157C
004071C2 0000 add byte ptr ds:[eax], al
004071C4 0000 add byte ptr ds:[eax], al
004071C6 0000 add byte ptr ds:[eax], al
004071C8 0000 add byte ptr ds:[eax], al
```

```
吾爱破解 - Twist1.exe - [LCG - 主线程, 模块 - Twist1]
文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H) [+] 快捷菜单 Tools BreakPoint-> BP P VB Not
暂停
0040157C 55 push ebp
0040157D 8BEC mov ebp, esp
0040157F 6A FF push -0x1
00401581 68 B8804000 push Twist1.004080B8
00401586 68 30394000 push Twist1.00403930
0040158B 64:A1 00000000 mov eax, dword ptr fs:[0]
00401591 50 push eax
00401592 64:8925 00000000 mov dword ptr fs:[0], esp
00401599 83EC 10 sub esp, 0x10
0040159C 53 push ebx
0040159D 56 push esi
0040159E 57 push edi
0040159F 8965 E8 mov dword ptr ss:[ebp-0x18], esp
004015A2 FF15 0C804000 call dword ptr ds:[&KERNEL32.GetVersion] kernel32.GetVersion
004015A8 33D2 xor edx, edx
004015AA 8AD4 mov dl, ah
004015AC 8915 A0B94000 mov dword ptr ds:[0x40B9A0], edx
004015B2 8BC8 mov ecx, eax
004015B4 81E1 FF000000 and ecx, 0xFF
004015BA 890D 9CB94000 mov dword ptr ds:[0x40B99C], ecx
```

来到OEP后

直接右键OllyDump脱壳 两种方式都保存下 发现方式二可以成功运行

空白区域右键-->中文搜索引擎 我们可以看到关键字串 那个je跳转貌似很不简单

```
00401067 push Twist1.00409030
004011DD push Twist1.0040904C
00401205 push Twist1.00409064
0040122E push Twist1.0040906C
0040127F push Twist1.00409084
004012CF push Twist1.00409078
004012DF push Twist1.00409070
0040156E push Twist1.004090C0
0040157F push -0x1
00401B80 mov eax, Twist1.004090C0
00401D97 mov ecx, dword ptr ds:[0x4094A8]
00401E52 mov ecx, dword ptr ds:[0x409344]
00401F5D mov eax, dword ptr ds:[0x409340]
0040206C mov ecx, dword ptr ds:[0x409340]
ntdll.dll
Reversing.Kr.CrackMe\n
Input:
%s
SetUnhandledExceptionFilter
Correct!\n
Wrong\n
调@
(Initial CPU selection)
调@
(null)
(null)
(null)
```

004012C5	90	nop	
004012C6	E8 75FFFFFF	call Twist1.00401240	
004012CB	85C0	test eax, eax	Twist1.00407000
004012CD	74 10	je short Twist1.004012DF	
004012CF	68 78904000	push Twist1.00409070	Correct!\n
004012D4	E8 5B020000	call Twist1.00401534	
004012D9	83C4 04	add esp, 0x4	
004012DC	33C0	xor eax, eax	Twist1.00407000
004012DE	C3	retn	
004012DF	68 70904000	push Twist1.00409070	Wrong!\n
004012E4	E8 4B020000	call Twist1.00401534	
004012E9	83C4 04	add esp, 0x4	

<https://blog.csdn.net/xiangshangbashaonian>

我们先不用管

接着单步

发现每次走到这里 F8的话 程序就会停止运行

吾爱破解 - Twist1.exe - [LCG - m主线程, 模块 - Twist1]

文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H) [+] 快捷菜单 Tools Breakf

暂停

0040160A	E8 711A0000	call Twist1.00403080	
0040160F	E8 90000000	call Twist1.004016A4	
00401614	A1 B0B94000	mov eax, dword ptr ds:[0x40B9B0]	
00401619	A3 B4B94000	mov dword ptr ds:[0x40B9B4], eax	
0040161E	50	push eax	
0040161F	FF35 A8B94000	push dword ptr ds:[0x40B9A8]	
00401625	FF35 A4B94000	push dword ptr ds:[0x40B9A4]	
0040162B	E8 40FCFFFF	call Twist1.00401270	
00401630	83C4 0C	add esp, 0xC	https://blog.csdn.net/xiangshangbashaonian
00401633	8945 E4	mov dword ptr ss:[ebp-0x1C], eax	

重新载入 运行到这里的时候 我们右键跟随 将着只要一单步就崩溃

百度了一下SetUnhandledExceptionFilter

说是用来设置异常捕获函数 猜测可能就是这个的原因了

吾爱破解 - bbb.exe - [LCG - 主线程, 模块 - bbb]

文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H) [+] 快捷菜单 Tools BreakPoint->

暂停

00401268	90	nop	
00401269	33C0	xor eax, eax	
0040126B	8BE5	mov esp, ebp	
0040126D	5D	pop ebp	
0040126E	C3	retn	
0040126F	90	nop	
00401270	E8 EBFEFFFF	call bbb.00401160	
00401275	E8 86FDFFFF	call bbb.00401000	
0040127A	A1 64B94000	mov eax, dword ptr ds:[0x40B964]	
0040127F	68 84904000	push bbb.00409084	ASCII "SetUnhandledExceptionFilter"
00401284	50	push eax	
00401285	E8 06FEFFFF	call bbb.00401090	
0040128A	83C4 08	add esp, 0x8	
0040128D	A3 60B94000	mov dword ptr ds:[0x40B960], eax	
00401292	68 40114000	push bbb.00401140	
00401297	FFD0	call eax	
00401299	33C0	xor eax, eax	
0040129B	1212	adc dl, byte ptr ds:[edx]	
0040129D	90	nop	
0040129E	90	nop	
0040129F	90	nop	
004012A0	90	nop	
004012A1	90	nop	

<https://blog.csdn.net/xiangshangbashaonian>

解决办法 我是直接在下边设置EIP来跳过这部分代码的执行 从而避免遇到异常

0040129B	. 1212	adc dl,byte ptr ds:[edx]	
0040129D	. 90	nop	
0040129E	. 90	nop	
0040129F	. 90	nop	

接着F8走过好几个nop 还有三个call 第三个call是用来获取我们的输入

输入iqiqiya回车 在单步几个nop 来到这里

004012C5	. 90	nop	
004012C6	. E8 75FFFFFF	call bbb.00401240	关键call
004012CB	. 85C0	test eax,eax	
004012CD	. 74 10	je short bbb.004012DF	
004012CF	. 68 78904000	push bbb.00409078	ASCII "Correct!\n"
004012D4	. E8 58020000	call bbb.00401534	
004012D9	. 83C4 04	add esp,0x4	
004012DC	. 33C0	xor eax,eax	
004012DE	. C3	retn	
004012DF	. 68 70904000	push bbb.00409070	ASCII "Wrong!\n"
004012E4	. E8 48020000	call bbb.00401534	
004012E9	. 83C4 04	add esp,0x4	
004012EC	. 33C0	xor eax,eax	
004012EE	. C3	retn	
004012EF	. 90	nop	

<https://blog.csdn.net/xiangshangbashaonian>

可以看到关键就是401240这个call

我们还是右键跟随

文件(F) 查看(V) 调试(D) 插件(P) 选项(T) 窗口(W) 帮助(H) [+] 快捷菜单 Tools BreakPoint->

暂停

0040123C	. 90	nop	
0040123D	. 90	nop	
0040123E	. 90	nop	
0040123F	. 90	nop	
00401240	. \$ 55	push ebp	
00401241	. 8BEC	mov ebp,esp	
00401243	. 83EC 0C	sub esp,0xC	esp的值减去0xC
00401246	. 33C0	xor eax,eax	eax:清0
00401248	. > 8A88 70894000	mov cl,byte ptr ds:[eax+0x408970]	将我们的输入iqiqiya 每次去一个字节赋值给cl
0040124E	. 84C9	test cl,cl	
00401250	. 74 0A	je short bbb.0040125C	
00401252	. 884C05 F4	mov byte ptr ss:[ebp+eax-0xC],cl	
00401256	. 40	inc eax	
00401257	. 83F8 0A	cmp eax,0xA	
0040125A	. 7C EC	jle short bbb.00401248	
0040125C	. > E9 AC5F0000	jmp bbb.0040720D	
00401261	. 90	nop	
00401262	. 90	nop	
00401263	. 90	nop	
00401264	. 90	nop	
00401265	. 90	nop	
00401266	. 90	nop	
00401267	. 90	nop	
00401268	. 90	nop	
00401269	. 33C0	xor eax,eax	
0040126B	. 8BE5	mov esp,ebp	
0040126D	. 5D	pop ebp	
0040126E	. C3	retn	

ds:[00408970]=69 ('i')

cl=C0

跳转来自 0040125A

地址	HEX 数据	ASCII
00408970	69 71 69 71 69 79 61 00 00 00 00 00 00 00 00 00	iqiqiya..
00408980	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00408990	00 00 00 00 F0 23 00 00 02 06 00 00 06 00 00 00?..

0019FF34 0019FF44 返回到 bbb

0019FF38 00401238

0019FF3C 0040906C ASCII "%s"

0019FF40 0019FF50

<https://blog.csdn.net/xiangshangbashaonian>

接下来走到jmp

00407208	. 00	db 00	
0040720C	. 00	db 00	
0040720D	. > 0000	add byte ptr ds:[eax],al	将eax的值 也就是我们输入的字符串的长度作为DS段的偏移
0040720F	. 00	db 00	

明天接着写