170923 逆向-Reversing.kr（MusicPlayer）

原创

奈沙夜影 于 2017-09-24 01:44:16 发布 1007 收藏

分类专栏： CrackMe

本文链接：https://blog.csdn.net/whklhhhh/article/details/78074712

CrackMe 专栏收录该内容

83 篇文章 2 订阅

订阅专栏

1625-5 王子昂 总结《2017年9月23日》【连续第356天总结】

A. Reversing.kr-Music Player

B.

ReadMe显示，现有程序只能播放1分钟，我们需要绕过这个显示从而得到flag，程序有多次检测。

从图标和提供的msvbvm60.dll可以知道是VB写的
OD加载发现从ntdll以后F9就直接飞了
用IDA加载也空空如也，只有三个子函数call dll，应该是反调机制？

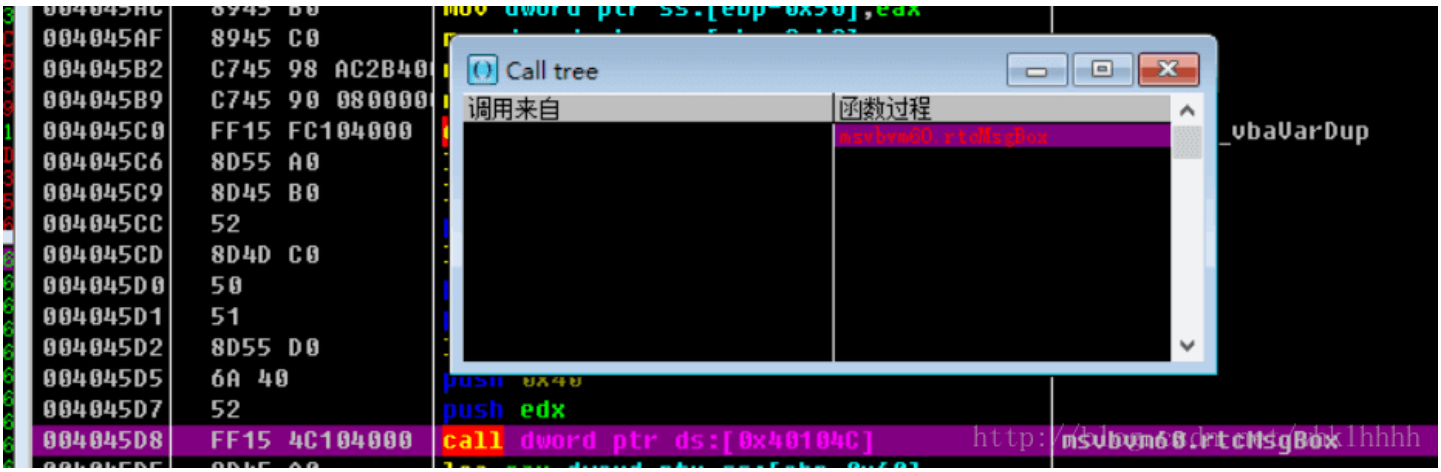
```
.text:00402890 sub_402890      proc near               ; CODE XREF: .text:00404EE2↓p
.text:00402890                                         ; .text:loc_405040↓p ...
.text:00402890                 mov     eax, dword_4073AC
.text:00402895                 or      eax, eax
.text:00402897                 jz      short loc_40289B
.text:00402899                 jmp     eax
.text:0040289B ; ---------------------------------------------------------------------------
.text:0040289B
.text:0040289B loc_40289B:                             ; CODE XREF: sub_402890+7↑j
.text:0040289B                 push    offset off_402878
.text:004028A0                 mov     eax, offset DllFunctionCall
.text:004028A5                 call    eax ; DllFunctionCall
.text:004028A7                 jmp     eax
.text:004028A7 sub_402890      endp
.text:004028A7
.text:004028A7 ; ---------------------------------------------------------------------------
.text:004028A9                 align 4
.text:004028AC                 dd 13h
.text:004028B0 aMcigeterrorstr db 'mciGetErrorStringA',0 ; DATA XREF: .text:004028C8↓o
.text:004028C3                 align 4
.text:004028C4 off_4028C4      dd offset aWinmm_dll    ; DATA XREF: sub_4028DC:loc_4028E7↓o
.text:004028C4                                         ; "winmm.dll"
.text:004028C8                 dd offset aMcigeterrorstr ; "mciGetErrorStringA"
.text:004028CC                 dd 40000h, 4073B0h, 2 dup(0)
.text:004028DC
.text:004028DC ; =============== S U B R O U T I N E =======================================
.text:004028DC
.text:004028DC
.text:004028DC sub_4028DC      proc near               ; CODE XREF: .text:004058E8↓p
.text:004028DC                 mov     eax, dword_4073B8
.text:004028E1                 or      eax, eax
.text:004028E3                 jz      short loc_4028E7
.text:004028E5                 jmp     eax
.text:004028E7 ; ---------------------------------------------------------------------------
.text:004028E7
.text:004028E7 loc_4028E7:                             ; CODE XREF: sub_4028DC+7↑j
.text:004028E7                 push    offset off_4028C4
.text:004028EC                 mov     eax, offset DllFunctionCall
.text:004028F1                 call    eax ; DllFunctionCall
.text:004028F3                 jmp     eax
.text:004028F3 sub_4028DC      endp
.text:004028F3
```


遂打开VB Decompiler，这次清晰地得到了结构和事件
我本来以为程序是OPEN文件以后，点击PLAY时间轴出现0:00/1:00
是直接截取前一分钟的内容加载播放
在PLAY的代码里找了半天，也只发现一个HS_POS.MAX=CINT(600)
前后搜索了一下都没找到相关内容，按地址去OD中下断可以断下来
但是没有找到600

无奈，去翻了下大佬的WriteUp，发现实际上是播放到1:00时弹窗，然后停止
VB弹窗调用的API是rtcMsgBox

在模块中查找未果
估计是因为根目录下提供了msvbvm60.dll，把它改了个名字，Ctrl+G就可以看到加载的dll换成c盘的系统msvbvm60.dll了，这个时候再查看所有调用可以找到rtcMsgBox，不过调用树中仍然是空的

在函数内部下断，运行，断住，ALT+F9返回程序领空才终于找到调用：



向上翻几句，发现时间对比和关键跳转：

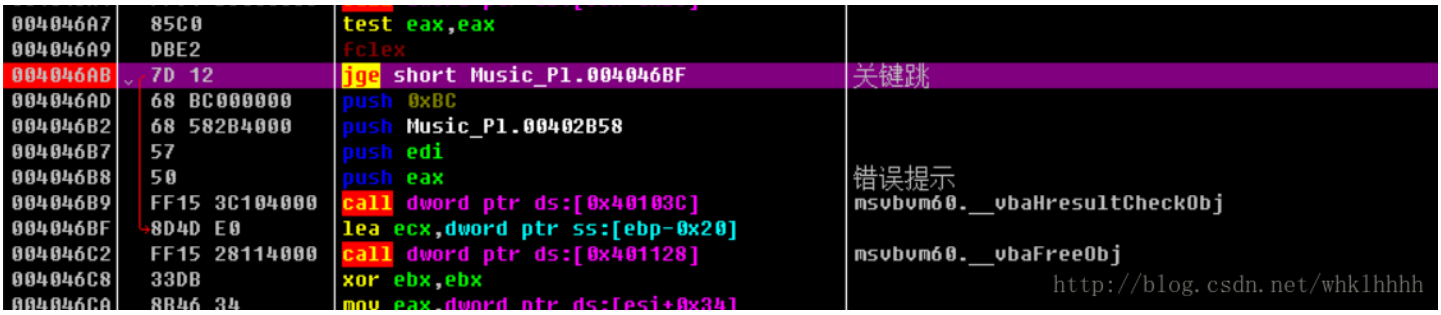
```
0040455D    8B85 5CFFFFFF   mov eax,dword ptr ss:[ebp-0xA4]
00404563    3D 60EA0000     cmp eax,0xEA60                  ; 时间比较
00404568    8945 E8         mov dword ptr ss:[ebp-0x18],eax
0040456B    0F8C 8D000000   jl Music_Pl.004045FE            ; 关键跳
```

将关键跳改为jmp，运行发现报错：

运行时错误：380

参照WriteUp，断Kernel32.RaiseException（同样调用树无结果，在函数内部下断）
似乎是OD的分析问题，断点位置的代码分析不整齐，导致再运行的时候理应被断住的地方报错了
不过没关系，虽然提示不太一样，在OD中F12暂停下来仍然可以找到调用
理论上来说ALT+K应该就能找到调用，不知道为什么仍然是空的
但是堆栈中还是可以看到，向上翻直到出现用户模块Music_P1的调用：

找到调用后向上看一眼就能发现关键跳转，重新运行jmp掉即可：



最后password出现在标题中，完成。

最后复盘的时候想起，这个事件应该来自于计时器
每秒检查一次时间是否超过60000ms
正好在VB Decompiler中有TMR_POS_Timer事件，检查发现：

```
loc_0040456B: If var_A4 < 60000 Then GoTo loc_004045FE
loc_00404574: var_eax = Call FrmMain.CMD_STOP_Click
loc_004045F9: GoTo loc_00404795
loc_004045FE: 'Referenced from: 0040456B
loc_00404601: If var_30 = -1 Then GoTo loc_004046CA
loc_0040460B: var_eax = FrmMain.Proc_0_10_403370(Me, var_30)
loc_00404627: If var_407000 <> 0 Then GoTo loc_00404631
loc_0040462F: GoTo loc_00404642
loc_00404631: 'Referenced from: 00404627
loc_00404642: 'Referenced from: 0040462F
loc_0040465A: If CLng((var_BC / 100)) <= 0 Then GoTo loc_0040466A
loc_00404664: ecx = "LI"
loc_0040466A: 'Referenced from: 0040465A
loc_0040466C: If CLng((var_BC / 100)) <> 0 Then GoTo loc_00404673
loc_00404673: 'Referenced from: 0040466C
loc_0040468D: var_C0 = var_20
loc_004046A1: HS_POS.Value = var_ret_1
loc_004046CA: 'Referenced from: 00404601
loc_004046EC: ClsMCI = var_A4
loc_0040470E: If var_A4 <= 60010 Then GoTo loc_00404795
loc_0040476F: ecx = &H402BDC & Chr(114) & &H402BE4
loc_00404795: 'Referenced from: 004045F9
```

```
  loc_0040479E: GoTo loc_004047CE
  loc_004047CD: Exit Sub
  loc_004047CE: 'Referenced from: 0040479E
  loc_004047CE: Exit Sub
End Sub
```


确实反编译出来了，未通过则调用STOP事件
不过之后的弹窗理论上来说应该在goto loc_00404795之前
但是这里的反编译却毫无显示

```
  loc_004045C0: call [004010FCh] ; %ecx = %S_edx_S '__vbaVarDup
  loc_004045C6: lea edx, var_60
  loc_004045C9: lea eax, var_50
  loc_004045CC: push edx
  loc_004045CD: lea ecx, var_40
  loc_004045D0: push eax
  loc_004045D1: push ecx
  loc_004045D2: lea edx, var_30
  loc_004045D5: push 00000040h
  loc_004045D7: push edx
  loc_004045D8: call [0040104Ch] ; @MsgBox(%StkVar1, %StkVar2, %StkVar3, %StkVar4, %StkVar5)
  loc_004045DE: lea eax, var_60
  loc_004045E1: lea ecx, var_50
  loc_004045E4: push eax
  loc_004045E5: lea edx, var_40
  loc_004045E8: push ecx
  loc_004045E9: lea eax, var_30
  loc_004045EC: push edx
  loc_004045ED: push eax
  loc_004045EE: push 00000004h
  loc_004045F0: call [00401018h] ; undef 'Ignore this '__vbaFreeVarList
  loc_004045F6: add esp, 00000014h
  loc_004045F9: jmp 00404795h
  loc_004045FE: cmp eax, FFFFFFFFh
```


学长说应该是反调或者花指令……之后得研究一下OD的去花插件了*(:3」∠)*

C. 明日计划
本来以为Reversing.kr的难度不大，就把最前面三个看起来很简单的题跳过了
现在看来还是很有意思的，回头把那三个也做了吧
EasyCrack