CPU Disasm

地址 十六进制数据 指令 注释 标签

0C1E350E 8D7E 58 lea edi,[esi+58]

0C1E3511 8D4424 10 lea eax,[esp+10]

0C1E3515 50 push eax

0C1E3516 8BCF mov ecx,edi

0C1E3518 C74424 50 00000 mov dword ptr ss:[esp+50],0

0C1E3520 E8 FFD30200 call <jmp.&MFC42.#3874> 跳转至 MFC42.#3874

0C1E3525 8B4C24 10 mov ecx,dword ptr ss:[esp+10]

0C1E3529 51 push ecx

0C1E352A 68 0815220C push offset CUQG\_ocx.0C221508 ASCII "CChatWnd::OnSend %s"

0C1E352F E8 6C500200 **call** CUQG\_ocx.**0C2085A0**

0C1E3534 83C4 08 add esp,8

0C1E3537 8D4C24 10 lea ecx,[esp+10]

0C1E353B E8 DED30200 call <jmp.&MFC42.#6282> 跳转至 MFC42.#6282

***特征码***：8D 7E 58 8D 44 24 10 50 8B CF C7 44 24 50 00 00 00 00

向下第二个call

CALL测试：

push ecx ;喊话的内容

push 0C221508

**call 0C2085A0**

push 081D1F48

push 122B1508

call 122985A0

发现并不好使！

-------------

扩大查找范围：（我是以经找好了，这个游戏喊话下send断点不好使，但是他程序哥哥开发时候不太认真打印了好多调试信息，所以，，，，你们懂的如何查找了吧）

CPU Disasm

地址 十六进制转储 命令 注释 Label

120734E0 /$ 6A FF PUSH -1 CUQG\_ocx.120734E0(guessed Arg1,Arg2)

120734E2 |. 68 601E0A12 PUSH 120A1E60 入口点

120734E7 |. 64:A1 00000000 MOV EAX,DWORD PTR FS:[0]

120734ED |. 50 PUSH EAX

120734EE |. 64:8925 00000000 MOV DWORD PTR FS:[0],ESP Installs SE handler 120A1E60

120734F5 |. 83EC 34 SUB ESP,34

120734F8 |. 53 PUSH EBX

120734F9 |. 55 PUSH EBP

120734FA |. 56 PUSH ESI

120734FB |. 8BF1 MOV ESI,ECX

120734FD |. 57 PUSH EDI

120734FE |. 8D4C24 10 LEA ECX,[ESP+10]

12073502 |. C686 C0070000 01 MOV BYTE PTR DS:[ESI+7C0],1

12073509 |. E8 5AD20200 CALL <JMP.&MFC42.#540> [MFC42.#540

1207350E |. 8D7E 58 LEA EDI,[ESI+58]

12073511 |. 8D4424 10 LEA EAX,[ESP+10]

12073515 |. 50 PUSH EAX /Arg1

12073516 |. 8BCF MOV ECX,EDI |

12073518 |. C74424 50 00000000 MOV DWORD PTR SS:[ESP+50],0 |

12073520 |. E8 FFD30200 CALL <JMP.&MFC42.#3874> \MFC42.#3874

12073525 |. 8B4C24 10 MOV ECX,DWORD PTR SS:[ESP+10]

12073529 |. 51 PUSH ECX /<%s> ***怀疑的游戏喊话CALL***

1207352A |. 68 08150B12 PUSH OFFSET 120B1508 |Format = "CChatWnd::OnSend %s"

1207352F |. E8 6C500200 CALL 120985A0 \CUQG\_ocx.120985A0, 游戏喊话call

12073534 |. 83C4 08 ADD ESP,8

12073537 |. 8D4C24 10 LEA ECX,[ESP+10]

1207353B |. E8 DED30200 CALL <JMP.&MFC42.#6282> 跳转到 MFC42.#6282

12073540 |. 8D4C24 10 LEA ECX,[ESP+10]

12073544 |. E8 CFD30200 CALL <JMP.&MFC42.#6283> 跳转到 MFC42.#6283

12073549 |. 8B86 E8070000 MOV EAX,DWORD PTR DS:[ESI+7E8]

1207354F |. 8B1D D46A0A12 MOV EBX,DWORD PTR DS:[<&USER32.GetWindowRect>] ASCII j

12073555 |. 8D5424 24 LEA EDX,[ESP+24]

12073559 |. 52 PUSH EDX /Rect

1207355A |. 50 PUSH EAX |hWnd

1207355B |. FFD3 CALL EBX \USER32.GetWindowRect

1207355D |. 8B4C24 24 MOV ECX,DWORD PTR SS:[ESP+24]

12073561 |. 8B5424 28 MOV EDX,DWORD PTR SS:[ESP+28]

12073565 |. 8B4424 10 MOV EAX,DWORD PTR SS:[ESP+10]

12073569 |. 83C1 0A ADD ECX,0A

1207356C |. 83C2 05 ADD EDX,5

1207356F |. 894C24 1C MOV DWORD PTR SS:[ESP+1C],ECX

12073573 |. 895424 20 MOV DWORD PTR SS:[ESP+20],EDX

12073577 |. 8B48 F8 MOV ECX,DWORD PTR DS:[EAX-8]

1207357A |. 85C9 TEST ECX,ECX

1207357C |.- 75 5D JNZ SHORT 120735DB

1207357E |. F646 44 04 TEST BYTE PTR DS:[ESI+44],04

12073582 |.- 74 3F JZ SHORT 120735C3

12073584 |. 8D4C24 1C LEA ECX,[ESP+1C]

12073588 |. 51 PUSH ECX /Arg2

12073589 |. 68 F8140B12 PUSH OFFSET 120B14F8 |/Arg1 = ASCII "ChatBlankNotice"

1207358E |. E8 8D540200 CALL 12098A20 |\CUQG\_ocx.12098A20

12073593 |. 8BCC MOV ECX,ESP |

12073595 |. 896424 20 MOV DWORD PTR SS:[ESP+20],ESP |

12073599 |. 50 PUSH EAX |/Arg1

1207359A |. E8 E1D10200 CALL <JMP.&MFC42.#537> |\MFC42.#537

1207359F |. 8D8E C40E0000 LEA ECX,[ESI+0EC4] |

120735A5 |. E8 2C800200 CALL 1209B5D6 \CUQG\_ocx.1209B5D6

120735AA |. 8B96 E40E0000 MOV EDX,DWORD PTR DS:[ESI+0EE4]

120735B0 |. 6A 00 PUSH 0 /lParam = 0

120735B2 |. 68 D0070000 PUSH 7D0 |wParam = 7D0

120735B7 |. 68 080B0000 PUSH 0B08 |Msg = 0B08

120735BC |. 52 PUSH EDX |hWnd

120735BD |. FF15 D06A0A12 CALL DWORD PTR DS:[<&USER32.PostMessageA>] \USER32.PostMessageA

120735C3 |> 68 FC2F0B12 PUSH OFFSET 120B2FFC /Arg1 = CUQG\_ocx.120B2FFC, ASCII

120735C8 |. 8BCF MOV ECX,EDI |

120735CA |. E8 B14C0000 CALL 12078280 \CUQG\_ocx.12078280

120735CF |. 8BCF MOV ECX,EDI

120735D1 |. E8 3CD30200 CALL <JMP.&MFC42.#5981> [MFC42.#5981

120735D6 |.- E9 19010000 JMP 120736F4

120735DB |> 8D4424 10 LEA EAX,[ESP+10]

120735DF |. 8BCE MOV ECX,ESI

120735E1 |. 50 PUSH EAX /Arg1

120735E2 |. E8 F9330000 CALL 120769E0 \CUQG\_ocx.120769E0

120735E7 |. 8D4C24 10 LEA ECX,[ESP+10]

120735EB |. E8 28D30200 CALL <JMP.&MFC42.#6283> 跳转到 MFC42.#6283

120735F0 |. 8B4C24 10 MOV ECX,DWORD PTR SS:[ESP+10]

120735F4 |. 8B41 F8 MOV EAX,DWORD PTR DS:[ECX-8]

120735F7 |. 85C0 TEST EAX,EAX

120735F9 |.- 75 4E JNZ SHORT 12073649

120735FB |. F646 44 04 TEST BYTE PTR DS:[ESI+44],04

120735FF |.- 0F84 EF000000 JZ 120736F4

12073605 |. 8D5424 1C LEA EDX,[ESP+1C]

12073609 |. 52 PUSH EDX /Arg2

1207360A |. 68 F8140B12 PUSH OFFSET 120B14F8 |/Arg1 = ASCII "ChatBlankNotice"

1207360F |. E8 0C540200 CALL 12098A20 |\CUQG\_ocx.12098A20

12073614 |. 8BCC MOV ECX,ESP |

12073616 |. 896424 20 MOV DWORD PTR SS:[ESP+20],ESP |

1207361A |. 50 PUSH EAX |/Arg1

1207361B |. E8 60D10200 CALL <JMP.&MFC42.#537> |\MFC42.#537

12073620 |. 8D8E C40E0000 LEA ECX,[ESI+0EC4] |

12073626 |. E8 AB7F0200 CALL 1209B5D6 \CUQG\_ocx.1209B5D6

1207362B |. 8B86 E40E0000 MOV EAX,DWORD PTR DS:[ESI+0EE4]

12073631 |. 6A 00 PUSH 0 /lParam = 0

12073633 |. 68 D0070000 PUSH 7D0 |wParam = 7D0

12073638 |. 68 080B0000 PUSH 0B08 |Msg = 0B08

1207363D |. 50 PUSH EAX |hWnd

1207363E |. FF15 D06A0A12 CALL DWORD PTR DS:[<&USER32.PostMessageA>] \USER32.PostMessageA

12073644 |.- E9 AB000000 JMP 120736F4

12073649 |> 8D4C24 10 LEA ECX,[ESP+10]

1207364D |. 51 PUSH ECX /Arg1

1207364E |. 8D4C24 18 LEA ECX,[ESP+18] |

12073652 |. E8 35D10200 CALL <JMP.&MFC42.#535> \MFC42.#535

12073657 |. 8B5424 10 MOV EDX,DWORD PTR SS:[ESP+10]

1207365B |. 6A 00 PUSH 0 /Arg1 = 0

1207365D |. 8D4C24 14 LEA ECX,[ESP+14] |

12073661 |. C64424 50 01 MOV BYTE PTR SS:[ESP+50],1 |

12073666 |. 8B6A F8 MOV EBP,DWORD PTR DS:[EDX-8] |

12073669 |. E8 9ED20200 CALL <JMP.&MFC42.#2915> \MFC42.#2915

**1207366E |. 45 INC EBP**

**1207366F |. 50 PUSH EAX /Arg2**

**12073670 |. 55 PUSH EBP |Arg1**

**12073671 |. 8BCE MOV ECX,ESI |**

**12073673 |. E8 A8000000 CALL 12073720 \CUQG\_ocx.12073720, 真正的游戏喊话call**

12073678 |. 85C0 TEST EAX,EAX

1207367A |.- 74 1A JZ SHORT 12073696

1207367C |. 8D4424 14 LEA EAX,[ESP+14]

12073680 |. 8BCE MOV ECX,ESI

12073682 |. 50 PUSH EAX /Arg1

12073683 |. E8 38060000 CALL 12073CC0 \CUQG\_ocx.12073CC0

12073688 |. 68 FC2F0B12 PUSH OFFSET 120B2FFC /Arg1 = CUQG\_ocx.120B2FFC, ASCII

1207368D |. 8BCF MOV ECX,EDI |

1207368F |. E8 EC4B0000 CALL 12078280 \CUQG\_ocx.12078280

12073694 |.- EB 50 JMP SHORT 120736E6

12073696 |> 8B56 78 MOV EDX,DWORD PTR DS:[ESI+78]

12073699 |. 8D4C24 34 LEA ECX,[ESP+34]

1207369D |. 51 PUSH ECX

1207369E |. 52 PUSH EDX

1207369F |. FFD3 CALL EBX

120736A1 |. F646 44 04 TEST BYTE PTR DS:[ESI+44],04

120736A5 |.- 74 3F JZ SHORT 120736E6

120736A7 |. 8D4424 1C LEA EAX,[ESP+1C]

120736AB |. 50 PUSH EAX /Arg2

120736AC |. 68 F8140B12 PUSH OFFSET 120B14F8 |/Arg1 = ASCII "ChatBlankNotice"

120736B1 |. E8 6A530200 CALL 12098A20 |\CUQG\_ocx.12098A20

120736B6 |. 8BCC MOV ECX,ESP |

120736B8 |. 896424 20 MOV DWORD PTR SS:[ESP+20],ESP |

120736BC |. 50 PUSH EAX |/Arg1

120736BD |. E8 BED00200 CALL <JMP.&MFC42.#537> |\MFC42.#537

120736C2 |. 8D8E C40E0000 LEA ECX,[ESI+0EC4] |

120736C8 |. E8 097F0200 CALL 1209B5D6 \CUQG\_ocx.1209B5D6

120736CD |. 8B8E E40E0000 MOV ECX,DWORD PTR DS:[ESI+0EE4]

120736D3 |. 6A 00 PUSH 0 /lParam = 0

120736D5 |. 68 D0070000 PUSH 7D0 |wParam = 7D0

120736DA |. 68 080B0000 PUSH 0B08 |Msg = 0B08

120736DF |. 51 PUSH ECX |hWnd

120736E0 |. FF15 D06A0A12 CALL DWORD PTR DS:[<&USER32.PostMessageA>] \USER32.PostMessageA

120736E6 |> 8D4C24 14 LEA ECX,[ESP+14]

120736EA |. C64424 4C 00 MOV BYTE PTR SS:[ESP+4C],0

120736EF |. E8 6ED00200 CALL <JMP.&MFC42.#800> [MFC42.#800

120736F4 |> 8D4C24 10 LEA ECX,[ESP+10]

120736F8 |. C74424 4C FFFFFFFF MOV DWORD PTR SS:[ESP+4C],-1

12073700 |. E8 5DD00200 CALL <JMP.&MFC42.#800> [MFC42.#800

12073705 |. 8B4C24 44 MOV ECX,DWORD PTR SS:[ESP+44]

12073709 |. 5F POP EDI

1207370A |. 5E POP ESI

1207370B |. 5D POP EBP

1207370C |. 33C0 XOR EAX,EAX

1207370E |. 64:890D 00000000 MOV DWORD PTR FS:[0],ECX

12073715 |. 5B POP EBX

12073716 |. 83C4 40 ADD ESP,40

12073719 \. C2 0800 RETN 8

我们找到的真正喊话call，其实是这个，如下：

mov eax,0AF20000 ; 申请一块内存地址，放入"123"的喊话内容

mov ebp,4 ; 喊话的长度是4

mov esi,105DCDD4 ; 这次游戏的平衡值

**PUSH EAX ; 喊话内容**

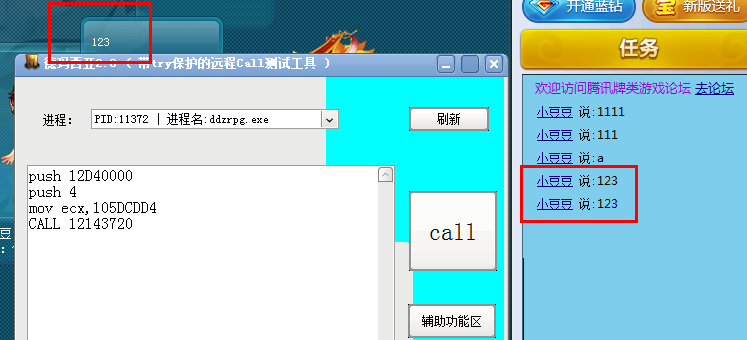
**PUSH EBP ; 喊话内容长度+1**

**MOV ECX,ESI ; ecx平衡地址**

**CALL 12073720 ; 这次游戏的call地址值（动态变化的，因为这个方法位于dll中，基址被重定位）**

**特征码：8B 54 24 10 6A 00 8D 4C 24 14 C6 44 24 50 01 8B 6A F8 找下面第2个call**

然后测试一下，发现可以啦！



但是你也看到了，call的地址两次游戏的启动，明显变化啦

=============================

动态call地址变化，如何解决？

思路：这个游戏主进程是：ddzrpg.exe，然后喊话的call位于模块：CUQG.ocx中，

所以可以先找到 喊话call，单独位于CUQG.ocx的位置:addr1；然后再在游戏进程中去获取模块CUQG.dll的加载地址:addr2 ；

那么，最后这个call的地址就是 addr = addr1+addr2;

**解决：**

思路有了就开始动手解决，先找单独CUQG.ocx中的 call地址；其次找取他位于游戏中的地址；

1.找CUQG.ocx中的call地址，方法：

OD直接附件CUQG.ocx

先把这个文件从游戏目录中拿出来： 

最后得到的地址是：10003720

2.找模块CUQG.dll加载到游戏里面的地址，方法：

这个地址是因为首先游戏主程序加载到 0x04000000，然后加载CUQG.ocx只能被重定位了，这里就需要取编程取模块地址了，

用到的一个函数是：GetModuleHandle()

函数的声明

HMODULE GetModuleHandle(LPCTSTR lpModuleName);

功能说明

获取一个特定的应用程序或动态链接库的模块句柄，且这个模块必须已经被加载到调用者的进程空间中。

参数说明

lpModuleName 模块名称

返回值

如执行成功成功，则返回模块句柄。零表示失败。通过GetLastError获得错误信息

注意

前提是：只有欲获取的模块已映射到调用该函数的进程内，才会正确得到模块句柄。常用模块映射函数：LoadLibrary（..）。

以上两步就可以找到喊话call的地址了！！！！

现在剩下的就是确定ecx平衡值了

[105DCDD4]=1212 6F48

[105DBBBC]=1212 6F48

[105DBEB4]=1212 6F48

这是一个很头疼的问题，目前没有什么好的方法，可以发现这个地址对应的一个值是固定的：1212 6F48(后四位固定)

根据类比联系一下call的地址，很容易发现一个规律 12120000 - 00030000 = 120F0000 正好是找call时后的找的CUQG.ocx模块的载入地址（句柄）

这里注意有个 00030000h 的 差值，发现这个规律也就好搜索了

那么就直接内存搜索：12126F48，就会得到 ecx平衡值；

我能力有限，解决的方法就是整个搜索内存了，方法很笨！希望大神指点一二。

=====================

最后的效果如下：

