2021/05/20_x86逆向C++_第7课_异常

笔记本： x86逆向-C++
创建时间： 2021/5/20 星期四 15:16
作者： ileemi

- 课前会议
- C++
- 动态分析定位异常相关代码

网络过滤驱动
minifilter

腾讯反外挂

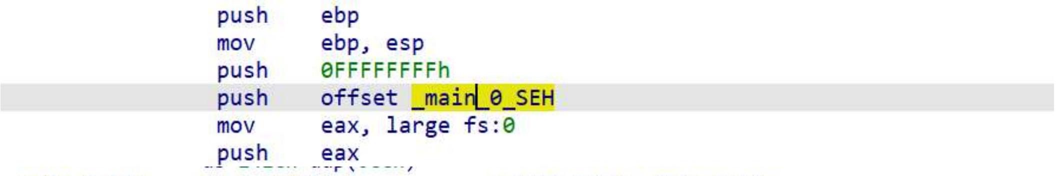
课前会议

基类个数判断：根据派生类覆盖虚表的次数课判定其基类的个数。未使用的类成员函数，Release版会对其进行不同层次的优化，可能会将其优化没。通过 IDA 定位虚表进行上下文分析，也可通过字符串进行分析。

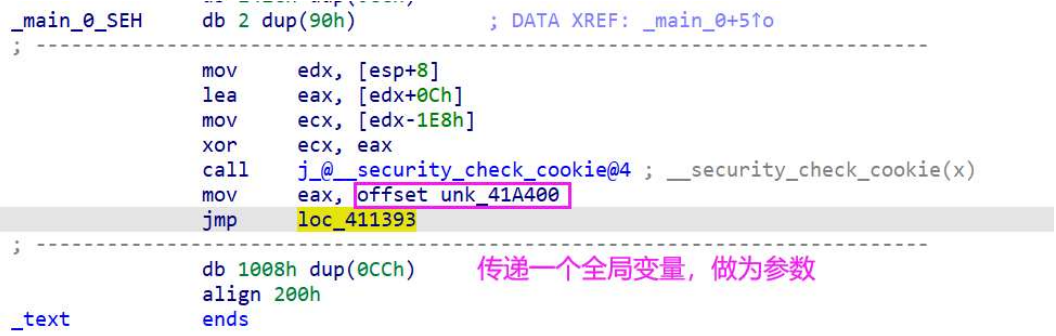
C++

try catch（接收异常）：基本数据类型会按照类型进行强制匹配，对象会根据继承层次匹配。

函数入口注册SEH fs:[0]
函数出口注销SEH fs:[0]
注册SEH：



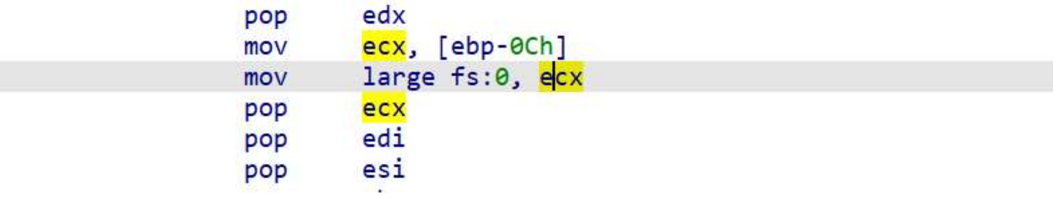
```
push    ebp
mov     ebp, esp
push    0FFFFFFFFh
push    offset _main_0_SEH
mov     eax, large fs:0
push    eax
```



```
_main_0_SEH     db 2 dup(90h)           ; DATA XREF: _main_0+5↑o
; ---------------------------------------------------------------------------
                mov     edx, [esp+8]
                lea     eax, [edx+0Ch]
                mov     ecx, [edx-1E8h]
                xor     ecx, eax
                call    j_@__security_check_cookie@4 ; __security_check_cookie(x)
                mov     eax, offset unk_41A400
                jmp     loc_411393
; ---------------------------------------------------------------------------
                db 1008h dup(0CCh)
                align 200h
_text           ends
```

传递一个全局变量，做为参数

注销SEH：

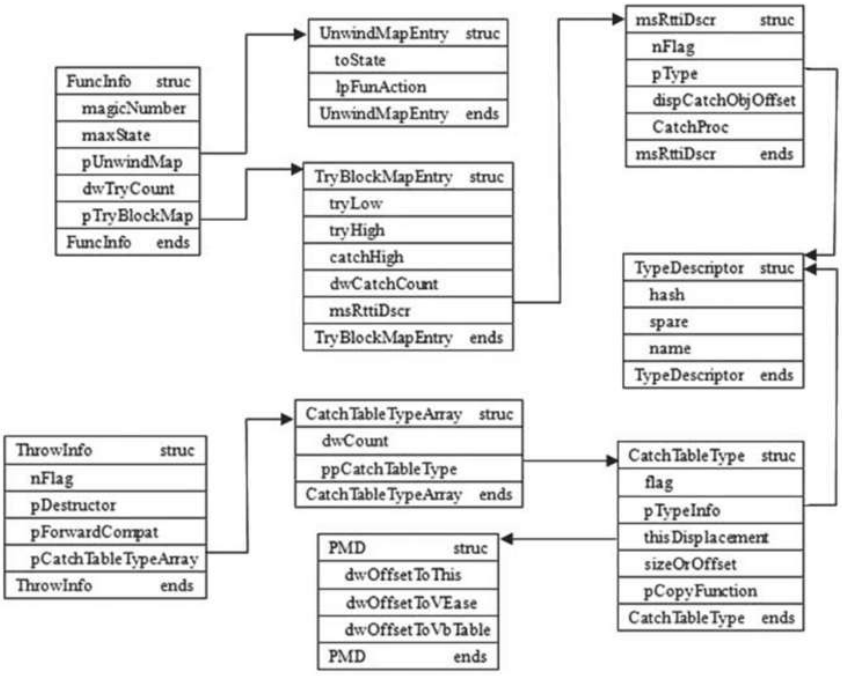


```
pop     edx
mov     ecx, [ebp-0Ch]
mov     large fs:0, ecx
pop     ecx
pop     edi
pop     esi
```

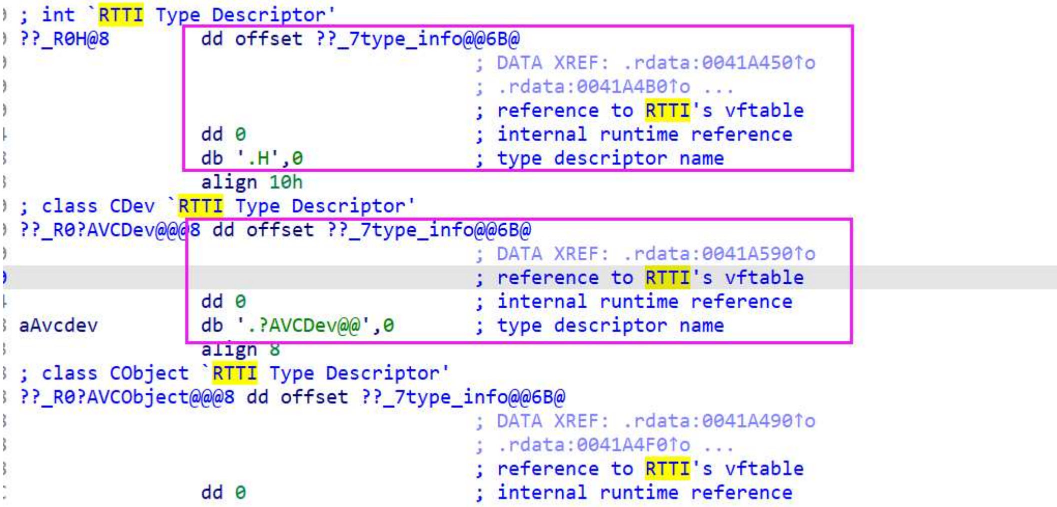
> Throw表、Catch表（Debug、Release都会建表）：

Throw表 ==> 记录Throw(类型 类型的值)
Catch表 ==> 记录所有catch的信息（类型，处理代码位置) RVA

> Catch表结构：在Visual Studio源码中可查看 "**ehdata.h**"

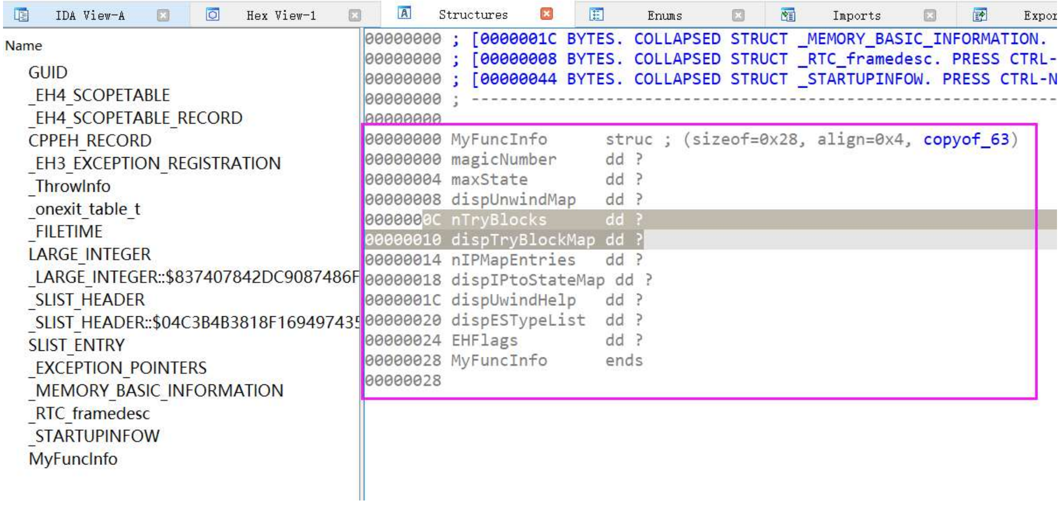


> RTTI：



```
; int `RTTI Type Descriptor'
??_R0H@8        dd offset ??_7type_info@@6B@
                                        ; DATA XREF: .rdata:0041A450↑o
                                        ; .rdata:0041A4B0↑o ...
                                        ; reference to RTTI's vftable
                dd 0                    ; internal runtime reference
                db '.H',0               ; type descriptor name
                align 10h
; class CDev `RTTI Type Descriptor'
??_R0?AVCDev@@@8 dd offset ??_7type_info@@6B@
                                        ; DATA XREF: .rdata:0041A590↑o
                                        ; reference to RTTI's vftable
                dd 0                    ; internal runtime reference
aAvcdev         db '.?AVCDev@@',0       ; type descriptor name
                align 8
; class CObject `RTTI Type Descriptor'
??_R0?AVCObject@@@8 dd offset ??_7type_info@@6B@
                                        ; DATA XREF: .rdata:0041A490↑o
                                        ; .rdata:0041A4F0↑o ...
                                        ; reference to RTTI's vftable
                dd 0                    ; internal runtime reference
```

在IDA中可以添加自定义的结构体头文件（使用C语法）。可直接将该结构添加到IDA中，解析对应程序的对应的结构体：

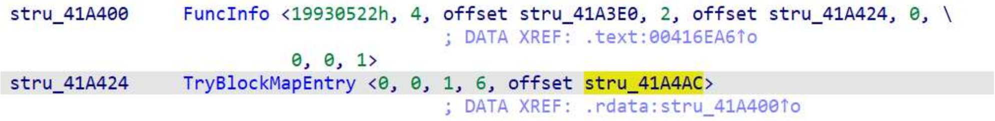


```
Name
  GUID
  _EH4_SCOPETABLE
  _EH4_SCOPETABLE_RECORD
  CPPEH_RECORD
  _EH3_EXCEPTION_REGISTRATION
  _ThrowInfo
  _onexit_table_t
  _FILETIME
  LARGE_INTEGER
  _LARGE_INTEGER::$837407842DC9087486F
  _SLIST_HEADER
  _SLIST_HEADER::$04C3B4B3818F16949743
  SLIST_ENTRY
  _EXCEPTION_POINTERS
  _MEMORY_BASIC_INFORMATION
  _RTC_framedesc
  _STARTUPINFOW
  MyFuncInfo
```

```
00000000 ; [0000001C BYTES. COLLAPSED STRUCT _MEMORY_BASIC_INFORMATION.
00000000 ; [00000008 BYTES. COLLAPSED STRUCT _RTC_framedesc. PRESS CTRL-
00000000 ; [00000044 BYTES. COLLAPSED STRUCT _STARTUPINFOW. PRESS CTRL-N
00000000 ; ---------------------------------------------------------------
00000000
00000000 MyFuncInfo      struc ; (sizeof=0x28, align=0x4, copyof_63)
00000000 magicNumber     dd ?
00000004 maxState        dd ?
00000008 dispUnwindMap   dd ?
0000000C nTryBlocks      dd ?
00000010 dispTryBlockMap dd ?
00000014 nIPMapEntries   dd ?
00000018 dispIPtoStateMap dd ?
0000001C dispUwindHelp   dd ?
00000020 dispESTypeList  dd ?
00000024 EHFlags         dd ?
00000028 MyFuncInfo      ends
00000028
```

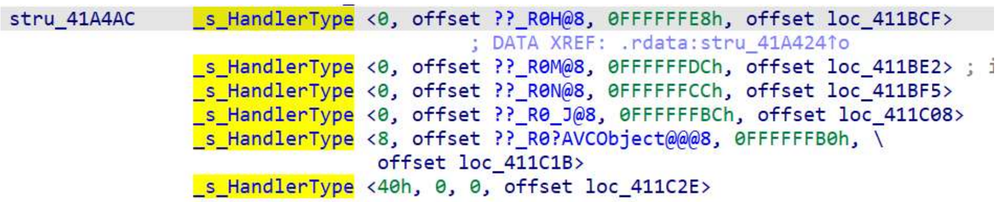
通过FunvInfo、UnwindMapEntry（异常展开）、TryBlockMapEntry（try的结构体数组地址，数量由nTryBlockMap决定）、_s_HandlerType（catch信息）、ThreadInfo等表信息，解析 "try catch"： "**alt+q**" 设置类型

1.



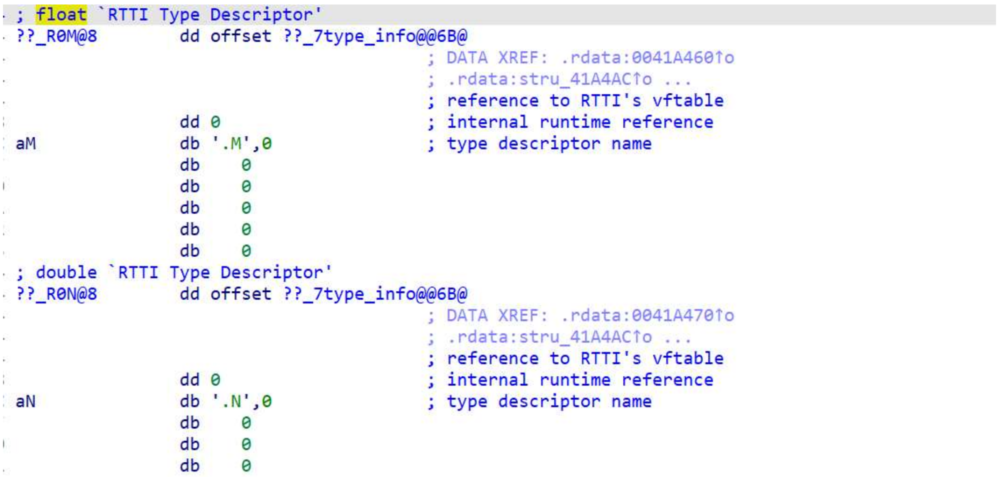
```
stru_41A400     FuncInfo <19930522h, 4, offset stru_41A3E0, 2, offset stru_41A424, 0, \
                                        ; DATA XREF: .text:00416EA6↑o
                          0, 0, 1>
stru_41A424     TryBlockMapEntry <0, 0, 1, 6, offset stru_41A4AC>
                                        ; DATA XREF: .rdata:stru_41A400↑o
```

2.



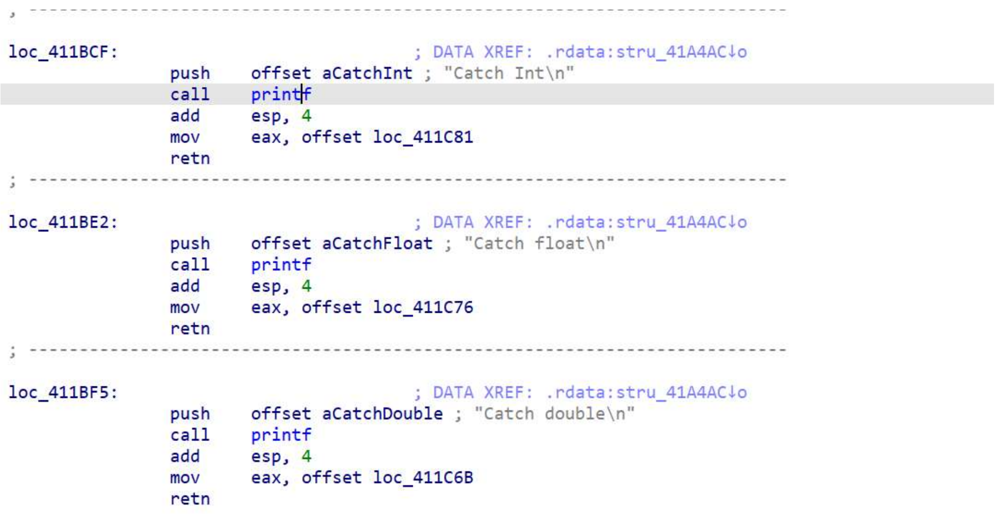
```
stru_41A4AC     _s_HandlerType <0, offset ??_R0H@8, 0FFFFFFE8h, offset loc_411BCF>
                                         ; DATA XREF: .rdata:stru_41A424↑o
                _s_HandlerType <0, offset ??_R0M@8, 0FFFFFFDCh, offset loc_411BE2> ; i
                _s_HandlerType <0, offset ??_R0N@8, 0FFFFFFCCh, offset loc_411BF5>
                _s_HandlerType <0, offset ??_R0_J@8, 0FFFFFFBCh, offset loc_411C08>
                _s_HandlerType <8, offset ??_R0?AVCObject@@@8, 0FFFFFFB0h, \
                                offset loc_411C1B>
                _s_HandlerType <40h, 0, 0, offset loc_411C2E>
```

3.



```
; float `RTTI Type Descriptor'
??_R0M@8        dd offset ??_7type_info@@6B@
                                        ; DATA XREF: .rdata:0041A460↑o
                                        ; .rdata:stru_41A4AC↑o ...
                                        ; reference to RTTI's vftable
                dd 0                    ; internal runtime reference
aM              db '.M',0               ; type descriptor name
                db    0
                db    0
                db    0
                db    0
                db    0
; double `RTTI Type Descriptor'
??_R0N@8        dd offset ??_7type_info@@6B@
                                        ; DATA XREF: .rdata:0041A470↑o
                                        ; .rdata:stru_41A4AC↑o ...
                                        ; reference to RTTI's vftable
                dd 0                    ; internal runtime reference
aN              db '.N',0               ; type descriptor name
                db    0
                db    0
                db    0
```

4.



```
; ---------------------------------------------------------------------------

loc_411BCF:                             ; DATA XREF: .rdata:stru_41A4AC↓o
                push    offset aCatchInt ; "Catch Int\n"
                call    printf
                add     esp, 4
                mov     eax, offset loc_411C81
                retn
; ---------------------------------------------------------------------------

loc_411BE2:                             ; DATA XREF: .rdata:stru_41A4AC↓o
                push    offset aCatchFloat ; "Catch float\n"
                call    printf
                add     esp, 4
                mov     eax, offset loc_411C76
                retn
; ---------------------------------------------------------------------------

loc_411BF5:                             ; DATA XREF: .rdata:stru_41A4AC↓o
                push    offset aCatchDouble ; "Catch double\n"
                call    printf
                add     esp, 4
                mov     eax, offset loc_411C6B
                retn
```

动态分析定位异常相关代码

在注册SEH代码中下断点（CxxFrameHandler3），等待编译器查表，在调用函数参数较多的函数处下断点。在程序抛异常之前定位catch的代码。

__try{}__except{} 也使用 SEH异常，表只有一个。没有数据类型一说，所有异常都会接收。