



HexRaysCodeXplorer: object oriented RE for fun and profit

Alexander Matrosov
@matrosov

Eugene Rodionov
@vxradius

Agenda

- ✓ C++ Code Reconstruction Problems
- ✓ Show problems on real examples (Flamer)
- ✓ HexRaysCodeXplorer v1.5 [H2HC Edition]

C++ Code Reconstruction Problems

- Object identification

- ✓ Type reconstruction

- Class layout reconstruction

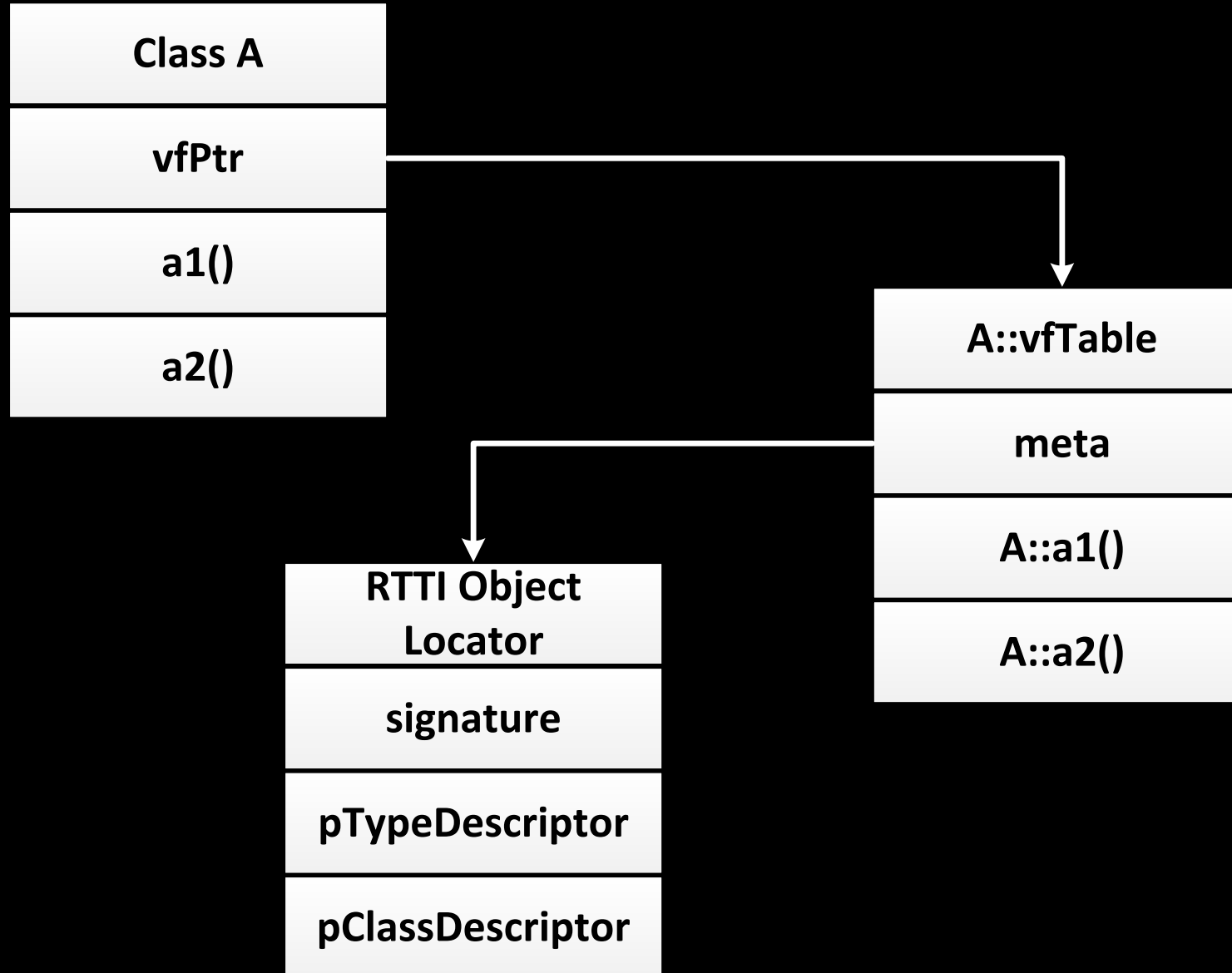
- ✓ Identify constructors/destructors
- ✓ Identify class members
- ✓ Local/global type reconstruction
- ✓ Associate object with exact method calls

- RTTI reconstruction

- ✓ Vtable reconstruction
- Associate vtable object with exact object
- ✓ Class hierarchy reconstruction

Depend on compiler design

C++ Code Reconstruction Problems



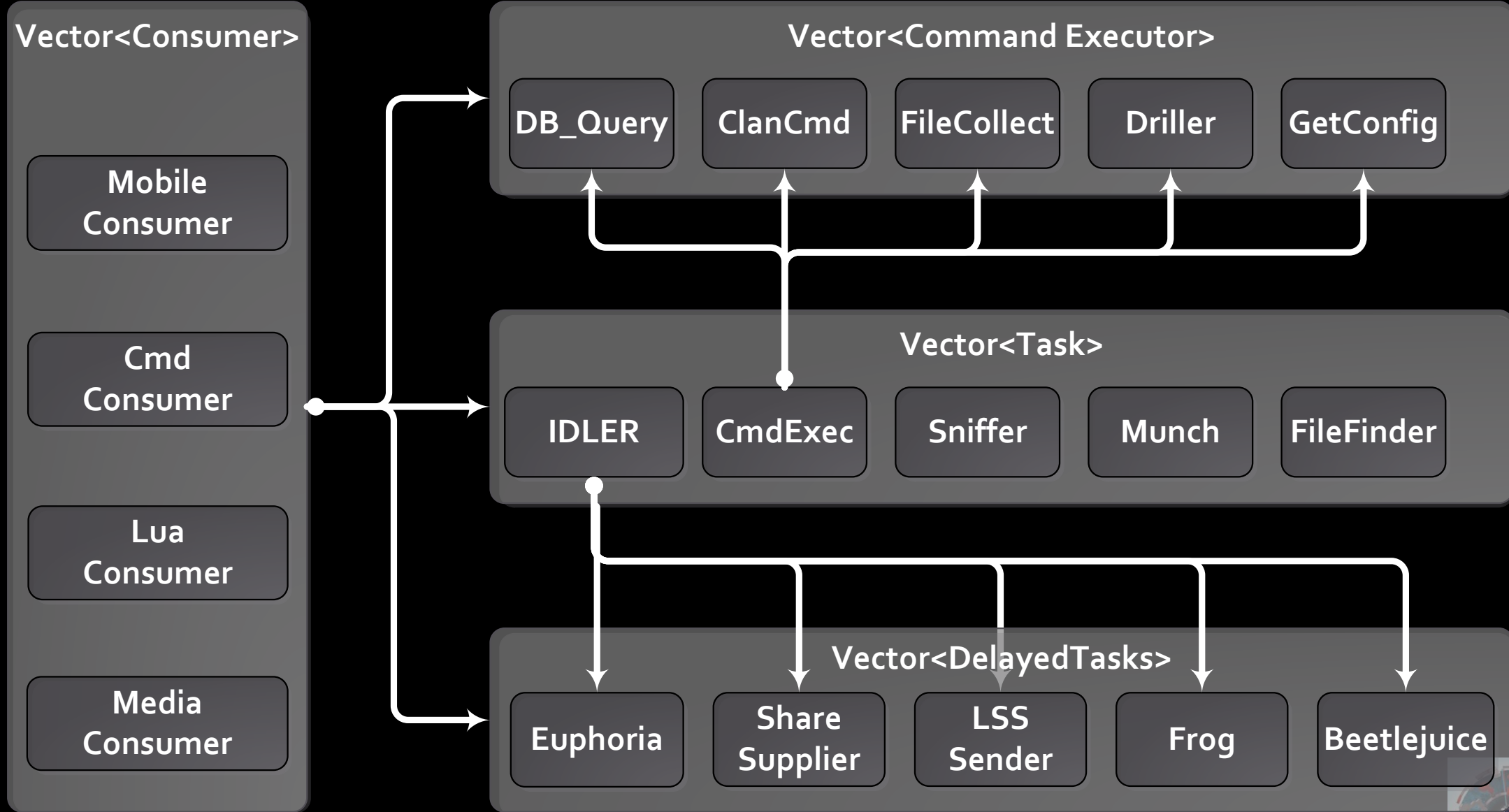
C++ Code Reconstruction Problems



REconstructing Flamer Framework



An overview of the Flamer Framework



An overview of the Flamer Framework

Vector<Consum

Mobile
Consum

Cmd
Consum

Lua
Consum

Media
Consum

```
0 0x10256aa0 - 0x10256afc: VECTOR_DATA_2_VTABLE method count: 23
1 0x10256bb0 - 0x10256bd8: FILE_MAPPING_1_VTABLE method count: 10
2 0x10256bd8 - 0x10256bf0: GLOBAL_EVENT_1_VTABLE method count: 6
3 0x102679a0 - 0x102679f0: PROCESS_HANDLE_VTABLE method count: 20
4 0x10267a90 - 0x10267acc: THREAD_HANDLE_VTABLE method count: 15
5 0x10267b08 - 0x10267b7c: FILE_VTABLE_0 method count: 29
6 0x10267bc0 - 0x10267bd8: EVENT_VTABLE method count: 6
7 0x10267df0 - 0x10267e40: PROCESS_HANDLE_VTABLE_0 method count: 20
8 0x10267e40 - 0x10267e80: EVENTGLOBAL_HZ_VTABLE method count: 16
9 0x10267e90 - 0x10267eb0: KASPER_EVENT_ENTRY_VTABLE method count: 8
10 0x10267f10 - 0x10267f34: TOKEN_HANDLE_VTABLE method count: 9
11 0x10268118 - 0x10268120: USTRING_REG_PATH_VTABLE method count: 2
12 0x10268128 - 0x102681a4: FILE_1_vTable method count: 31
13 0x10268260 - 0x10268298: ENC_2_VTABLE method count: 14
14 0x10268478 - 0x102684a8: ZLIB_HLPR_VTABLE method count: 12
15 0x102684e0 - 0x1026853c: ENC_3_VTABLE method count: 23
16 0x1026856c - 0x10268590: SYSTEM_HANDLE_INFO_VTABLE method count: 9
17 0x10268688 - 0x102686bc: DICT_1_VTABLE method count: 13
18 0x10268d78 - 0x10268dd4: MAIN_VECT_3_VTABLE method count: 23
19 0x10268f80 - 0x10268fe8: CONCOL_HANDLER_VTABLE method count: 26
20 0x102693c0 - 0x102693d0: CMD_EXECUTER_VIPER_VTABLE method count: 4
21 0x10269490 - 0x102694ec: MAIN_VECT_1_VTABLE method count: 23
22 0x102694f0 - 0x1026954c: MAIN_VECT_2_VTABLE method count: 23
23 0x10269550 - 0x102695ac: MAIN_VECT_4_VTABLE method count: 23
24 0x10269768 - 0x102697dc: MAIN_VECT_2_IDLER_VTABLE method count: 29
25 0x102697dc - 0x10269818: _MAIN_VECT_2_IDLER_VTABLE method count: 15
26 0x10269818 - 0x10269874: VECT_VTABLE method count: 23
27 0x10269874 - 0x10269884: MAIN_VECT_4_TIME_UPDATER_VTABLE method count: 4
28 0x10269a2c - 0x10269a68: MAIN_3_VECT_1_VTABLE method count: 15
29 0x10269b48 - 0x10269bbc: MAIN_VECT_2_HNT_VTABLE method count: 29
30 0x10269bc8 - 0x10269c3c: MAIN_VECT_2_VOLUME_SUPPLIER_VTABLE method count: 29
31 0x10269c40 - 0x10269cb4: MAIN_VECT_2_VIRTUAL_VOLUME_SUPPLIER_VTABLE method count: 29
32 0x10269e10 - 0x10269e84: MAIN_VECT_2_HeadacheConsumer_VTABLE method count: 29
```

nfig

der

lejuice

<http://www.welivesecurity.com/2012/08/02/flamer-analysis-framework-reconstruction/>



An overview of the Flamer Framework

Vector<Consum

Mobile
Consumer

Cmd
Consumer

Lua
Consumer

Media
Consumer

```
0 0x10256aa0 - 0x10256afc: VECTOR_DATA_2_VTABLE method count: 23
1 0x10256bb0 - 0x10256bd8: FILE_MAPPING_1_VTABLE method count: 10
2 0x10256bd8 - 0x10256bf0: GLOBAL_EVENT_1_VTABLE method count: 6
3 0x102679a0 - 0x102679f0: PROCESS_HANDLE_VTABLE method count: 20
4 rdata:10267F38 off_10267F38 dd offset sub_10014D09 ; DATA XREF: Vector1_Copy+18to
5 rdata:10267F38 ; Vector1_Init+1Cto ...
6 rdata:10267F38 ; action
7 rdata:10267F3C dd offset File_GetHandle ; toState
8 rdata:10267F40 dd offset sub_10054E04 ; action
9 rdata:10267F44 dd offset sub_10054E04 ; toState
10 rdata:10267F48 dd offset sub_1001E652 ; action
11 rdata:10267F4C dd offset sub_1001E652 ; toState
12 rdata:10267F50 dd offset sub_10035BCA ; action
13 rdata:10267F54 dd offset sub_1019373F ; toState
14 rdata:10267F58 dd offset sub_1001448A ; action
15 rdata:10267F5C dd offset Data1_Vector_Insert ; toState
16 rdata:10267F60 dd offset sub_10014522 ; action
17 rdata:10267F64 dd offset sub_10014580 ; toState
18 rdata:10267F68 dd offset sub_100145A1 ; action
19 rdata:10267F6C dd offset sub_100036D0 ; toState
20 rdata:10267F70 dd offset sub_100EDD41 ; action
21 rdata:10267F74 dd offset sub_10003C05 ; toState
22 rdata:10267F78 dd offset sub_10028089 ; action
23 rdata:10267F7C dd offset sub_100145C2 ; toState
24 rdata:10267F80 dd offset sub_1001460E ; action
25 rdata:10267F84 dd offset VectData1_CheckLimits ; toState
26 rdata:10267F88 dd offset get_less_power ; action
27 rdata:10267F8C dd offset sub_10014680 ; toState
28 rdata:10267F90 dd offset sub_10014732 ; action
29 rdata:10267F94 dd 0 ; toState
29 0x10269b48 - 0x10269bbc: MAIN_VECT_2_HNT_VTABLE method count: 29
30 0x10269bc8 - 0x10269c3c: MAIN_VECT_2_VOLUME_SUPPLIER_VTABLE method count: 29
31 0x10269c40 - 0x10269cb4: MAIN_VECT_2_VIRTUAL_VOLUME_SUPPLIER_VTABLE method count: 29
32 0x10269e10 - 0x10269e84: MAIN_VECT_2_HeadacheConsumer_VTABLE method count: 29
```

nfig

der

lejuice

<http://www.welivesecurity.com/2012/08/02/flamer-analysis-framework-reconstruction/>



Identify Smart Pointer Structure

- Smart pointers
- Strings
- Vectors to maintain the objects
- Custom data types:
 - ✓ wrappers
 - ✓ tasks,
 - ✓ triggers
 - ✓ and etc.



Data Types Being Used: Smart pointers

```
typedef struct SMART_PTR
```

```
{
```

```
    void          *pObject;    // pointer to the object
```

```
    int          *RefNo;      // reference counter
```

```
};
```

```
SMART_PTR_STRUCT *__userpurge SmartPtr_InitalizeByObject<eax>(SMART_PTR_STRUCT *a1<esi>, void *pObject)
{
    int *v2; // eax@1

    LOBYTE(v2) = new(4);
    if ( v2 )
        *v2 = 1;
    else
        v2 = 0;
    a1->RefNo = v2;
    a1->Object = pObject;
    return a1;
}
```

Identify Smart Pointer Structure

```
SmartPtr_InializeByObject proc near      ; CODE XR
                                      ; sub_100
```

```
var_10      = dword ptr -10h
var_C       = dword ptr -0Ch
var_4       = dword ptr -4
arg_0       = dword ptr  8
```

```
mov     eax, offset sub_101C690A
call    __EH_prolog
push    ecx
push    4
call    alloc_mem
pop     ecx
mov     [ebp+var_10], eax
and     [ebp+var_4], 0
test    eax, eax
jz      short loc_100041F5
mov     dword ptr [eax], 1
jmp     short loc_100041F7
```

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

```
loc_100041F5:                          ; CODE XR
xor     eax, eax
```

```
loc_100041F7:                          ; CODE XR
or      [ebp+var_4], 0FFFFFFFFh
mov     ecx, [ebp+var_C]
mov     [esi+4], eax
mov     eax, [ebp+arg_0]
mov     [esi], eax
mov     eax, esi
mov     large fs:0, ecx
leave
retn    4
```

```
SmartPtr_InializeByObject endp
```



```
SMART_PTR_STRUCT *__userpurge SmartPtr
{
    int *v2; // eax@1

    v2 = alloc_mem(4);
    if ( v2 )
        *v2 = 1;
    else
        v2 = 0;
    a1->RefNo = v2;
    a1->Object = a2;
    return a1;
}
```


Data Types Being Used: Vectors

```
struct VECTOR
{
    void *vTable;           // pointer to the table
    int NumberOfItems;     // self-explanatory
    int MaxSize;           // self-explanatory
    void *vector;          // pointer to buffer with elements
};
```

- **Used to handle the objects:**

- ✓ tasks
- ✓ triggers
- ✓ etc.

Identify Exact Virtual Function Call in Vtable

VECTOR_DATA_1_VTABLE dd offset sub_10048202

```
STRUCT_4_3 *__thiscall CSocket_Cto
{
    STRUCT_4_3 *v1; // esi@1

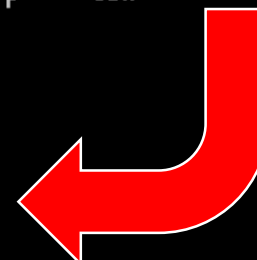
    v1 = this;
    this->vTable = &csocket_v_table;
    this->struct44 = 0;
    this->DeviceTcp = 0;
    this->DeviceUdp = 0;
    sub_19664(&this->struct41);
    sub_13E22(&v1->struct2);
    v1->SocketNumber = 0;
    v1->RefNo = 0;
    return v1;
}
```

```
dd offset File_GetHandle
dd offset sub_10054E04
dd offset sub_10054E04
dd offset sub_1001E652
dd offset sub_1001E652
dd offset sub_10035BCA
dd offset sub_1019373F
dd offset sub_10047E6C
dd offset Data1_Vector_Insert
dd offset sub_10047F09
dd offset VectData1_GetEntry
dd offset sub_10047F84
dd offset sub_10047FCE
dd offset sub_10047DC6
dd offset sub_10047FA5
dd offset sub_10028089
dd offset sub_10047FF5
dd offset sub_1004803A
dd offset VectData1_CheckLimits
dd offset get_less_power
dd offset sub_10038440
dd offset sub_100480AC
dd offset sub_100476F7
```

```
; DATA X sh 4609h
; Data1 p eax
v edx, eax
v ecx, [edi+0Ch]
v eax, [ecx]
a edx, [ebp-8]
sh edx
sh dword ptr [edi+10h]
ll dword ptr [eax+0Ch]
st al, al
z short loc_1BCA9
v ecx, edi ; this
ll CloseTcpSocket
v eax, 44CFh
sh eax
p ebx
```

Load pointer to table
of virtual methods

Call virtual method

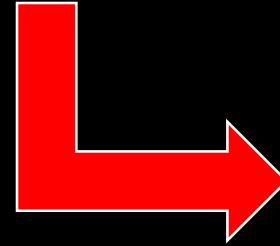


off_1026A064

```
; DATA X
; sub_100476F7
```

Identify Exact Virtual Function Call in Vtable

```
int __thiscall Rc4_GetBufferSize(_RC4_STRUCT *this)
{
    return (this->Reader->vTable->GetResBufSize());
}
```



```
; int __thiscall Rc4_GetBufferSize(_RC4_STRUCT *this)
Rc4_GetBufferSize proc near                ; DATA XREF:
    mov     ecx, [ecx+4]
    mov     eax, [ecx]
    jmp     dword ptr [eax+10h]
Rc4_GetBufferSize endp
```



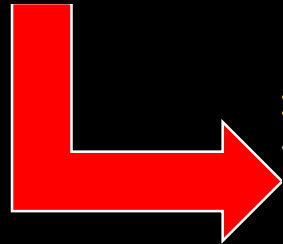
```
RC4_UTABLE    dd offset Rc4_GetReader ; DATA XREF: sub_1011E919+1E10
               dd offset Rc4_GetWriter
               dd offset ?Destroy@EventWaitNode@details@Concurrency@@QAEXXZ
               dd offset ?Sweep@EventWaitNode@details@Concurrency@@QAE_NXZ
               dd offset Rc4_GetBufferSize
               dd offset Rc4_IncreaseSize
               dd offset Rc4_Check
               dd offset Rc4_InitEmpty
               dd offset Rc4_Release
               dd offset Rc4_GetMuxName
```

Identify Custom Type Operations

```
; int __thiscall DataVector1_GetEntry(VECTOR_DATA_1_STRUCT *this, int a2)
DataVector1_GetEntry proc near          ; DATA XREF: .rdata:10256ACC↓o
```

```
arg_0= dword ptr 8
```

```
push    ebp
mov     ebp, esp
push    esi
push    edi
mov     edi, [ebp+arg_0]
mov     esi, ecx
mov     eax, [esi]
push    offset unk_103313A6
push    edi
call    dword ptr [eax+4Ch]
mov     eax, [esi+0Ch]
lea     eax, [eax+edi*4]
pop     edi
pop     esi
pop     ebp
retn    4
DataVector1_GetEntry endp
```



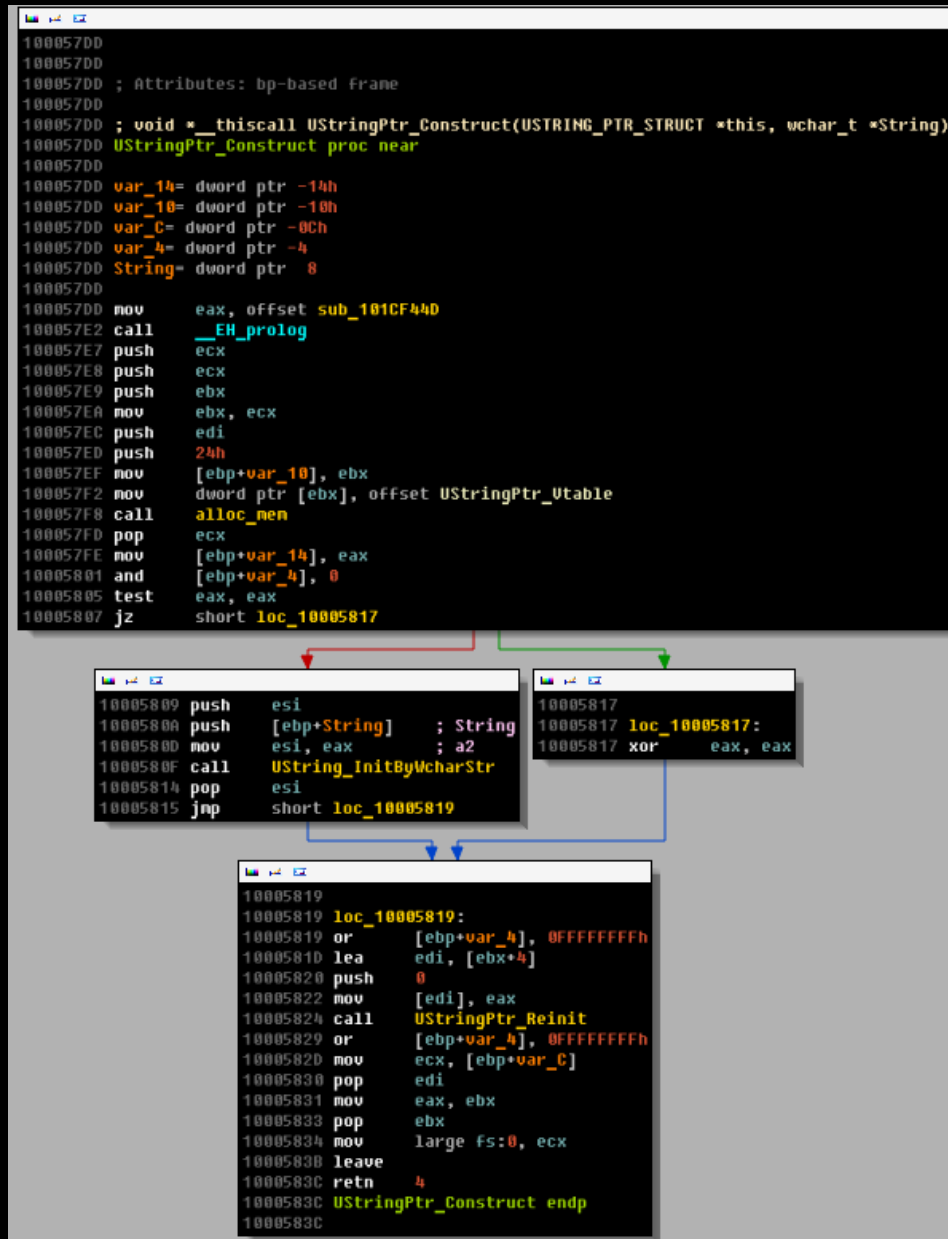
```
int __thiscall DataVector1_GetEntry(VECTOR_DATA_1_STRUCT *this, int a2)
{
    VECTOR_DATA_1_STRUCT *v2; // esi@1

    v2 = this;
    (this->vTable->CheckVectLimits)(a2, &unk_103313A6);
    return (v2->vector + 4 * a2);
}
```


Data Types Being Used: Strings

```
struct USTRING_STRUCT
{
    void *vTable;           // pointer to the table
    int RefNo;              // reference counter
    int Initialized;
    wchar_t *UnicodeBuffer; // pointer to unicode string
    char *AsciiBuffer;      // pointer to ASCII string
    int AsciiLength;        // length of the ASCII string
    int Reserved;
    int Length;             // Length of unicode string
    int LengthMax;          // Size of UnicodeBuffer
};
```

Identify Objects Constructors



Identify Objects Constructors

```
10005700
10005700
10005700 ; Attributes: bp-based frame
10005700
10005700 ; void __thiscall UStringPtr_Construct(USTRING_PTR_STRUCT *this, wchar_t *String)
10005700 UStringPtr_Construct proc near
10005700
10005700     var_14= dword ptr -14h
10005700     var_10= dword ptr -10h
10005700     var_C= dword ptr -0Ch
10005700     use [ebp+var_14]
```

```
USTRING_PTR_STRUCT *__thiscall UStringPtr_Construct(USTRING_PTR_STRUCT *this, wchar_t *String)
{
    USTRING_PTR_STRUCT *v2; // ebx@1
    USTRING_STRUCT *v3; // eax@1
    USTRING_STRUCT *v4; // eax@2

    v2 = this;
    this->vTable = UStringPtr_Vtable;
    v3 = alloc_mem(36);
    if ( v3 )
        v4 = UString_InitByWcharStr(v3, String);
    else
        v4 = 0;
    v2->String = v4;
    UStringPtr_Reinit(&v2->String, 0);
    return v2;
}
```

```
10005820 push    0
10005822 mov     [edi], eax
10005824 call    UStringPtr_Reinit
10005829 or     [ebp+var_4], 0FFFFFFFh
1000582D mov     ecx, [ebp+var_C]
10005830 pop     edi
10005831 mov     eax, ebx
10005833 pop     ebx
10005834 mov     large fs:0, ecx
10005838 leave
1000583C retn    4
1000583C UStringPtr_Construct endp
1000583C
```

REconstructing Object's Attributes

141	MAIN_VECT_3_OBJ_X_1_STRUCT	00000010	Auto	struct {int vTable;int field1;int field2;int field3;}
159	MAIN_VECT_3_ENTRY	00000044		struct {int vTable;int field4;int field5;char field6_0;char field6_1;char field6_2;cha...
143	MAIN_VECT_3_1_STRUCT	0000004C	Auto	struct {int vTable;MAIN_VECT_3_1_1_STRUCT Main31;GLOBAL_EVENT_STRUCT_...
145	MAIN_VECT_3_1_PTR_STRUCT	00000008	Auto	struct {MAIN_VECT_3_1_STRUCT *pObject;int *RefNo;}
160	MAIN_VECT_3_1_ENTRY	00000010		struct {int vTable;int field0;int field1;int field2;}
144	MAIN_VECT_3_1_1_STRUCT	00000024	Auto	struct {int vTale;VECTOR_DATA_1_STRUCT Vect1;VECTOR_DATA_1_STRUCT Vec...
132	MAIN_VECT_2_STRUCT	00000080	Auto	struct {int field0;VECTOR_DATA_1_STRUCT VectorOfObjects;GLOBAL_EVENT_ST...
148	MAIN_VECT_2_OBJ_HEAD_VTABLE	00000074	Auto	struct {int field0;int field1;int field2;int field3;int field4;int field5;int field...
133	MAIN_VECT_2_OBJ_HEAD_STRUCT	00000088	Auto	struct {MAIN_VECT_2_OBJ_HEAD_VTABLE *vTable;HANDLE_INFO_STRUCT *Thr...
135	MAIN_VECT_2_OBJ_HEAD_1_STRUCT	00000028	Auto	struct {int vTable;_MAIN_VECT_2_OBJ_HEAD_1_STRUCT Vect21;}
165	MAIN_VECT_2_NHT_STRUCT	00000098	Auto	struct {MAIN_VECT_2_OBJ_HEAD_STRUCT Header;SMART_PTR_STRUCT EntryP...
136	MAIN_VECT_2_MUNCH_OBJ_STRUCT	000000DC	Auto	struct {MAIN_VECT_2_OBJ_HEAD_STRUCT Header;_MAIN_VECT_2_MUNCH_OB...
157	MAIN_VECT_2_JIMMY_STRUCT	00000188	Auto	struct {MAIN_VECT_2_JIMMY_1_STRUCT Jimmy1;MAIN_VECT_2_JIMMY_2_STRU...
158	MAIN_VECT_2_JIMMY_2_STRUCT	00000150	Auto	struct {MAIN_VECT_2_OBJ_HEAD_STRUCT head;GLOBAL_EVENT_STRUCT_1 Sy...
156	MAIN_VECT_2_JIMMY_1_STRUCT	00000038	Auto	struct {int vTable;int field1;int field2;int field3;int field4;int field5;int field...
146	MAIN_VECT_2_IDLER_STRUCT	000000BC	Auto	struct {MAIN_VECT_2_OBJ_HEAD_STRUCT Header;_MAIN_VECT_2_IDLER_STR...
162	MAIN_VECT_2_GADGET_SUPP_STRUCT	000003DC	Auto	struct {MAIN_VECT_2_OBJ_HEAD_STRUCT Header;EVENT_HANDLE_STRUCT Ev...
163	MAIN_VECT_2_GADGET_SUPP_1_STRUCT	00000010	Auto	struct {int vTable;int field0;int field1;int field2;}
172	MAIN_VECT_2_COMMAND_FILE_FINDER_STRUCT	000000DC	Auto	struct {MAIN_VECT_2_OBJ_HEAD_STRUCT Header;MAIN_VECT_4_OBJ_HEAD_S...
173	MAIN_VECT_2_COMMAND_FILE_FINDER_NOTIF_ENTRY_...	00000014		struct {HANDLE_INFO_PTR_STRUCT HandleInfo;USTRING_PTR_STRUCT FolderN...
164	MAIN_VECT_2_CMD_RUNNER_STRUCT	0000009C	Auto	struct {MAIN_VECT_2_OBJ_HEAD_STRUCT Header;int CmdDispatcher;int Comma...
130	MAIN_VECT_1_STRUCT	000000E4	Auto	struct {int vTable;VECTOR_DATA_1_STRUCT VectorOfCmdDispatchers;GLOBAL_E...
138	MAIN_VECTOR_4_GLOB_STRUCT	00000034	Auto	struct {GLOBAL_EVENT_STRUCT_1 SyncEvent;VECTOR_DATA_1_STRUCT Vector;}

REconstructing Object's Attributes

141	MAIN_VECT_3_OBJ_X_1_STRUCT	00000010	Auto	struct {int vTable;int field1;int field2;int field3;}
159	MAIN_VECT_3_ENTRY	00000044		struct {int vTable;int field4;int field5;char field6_0;char field6_1;char field6_2;cha...
143	MAIN_VECT_3_1_STRUCT			STRUCT Main31;GLOBAL_EVENT_STRUCT_...
145	MAIN_VECT_3_1_PTR_STRUCT			ject;int *RefNo;}
160	MAIN_VECT_3_1_ENTRY			field2;}
144	MAIN_VECT_3_1_1_STRUCT			UCT Vect1;VECTOR_DATA_1_STRUCT Vec...
132	MAIN_VECT_2_STRUCT			UCT VectorOfObjects;GLOBAL_EVENT_ST...
148	MAIN_VECT_2_OBJ_HEAD_VTAB			field3;int field4;int field5;int field6;int field...
133	MAIN_VECT_2_OBJ_HEAD_STRL			BLE *vTable;HANDLE_INFO_STRUCT *Thr...
135	MAIN_VECT_2_OBJ_HEAD_1_ST			HEAD_1_STRUCT Vect21;}
165	MAIN_VECT_2_NHT_STRUCT			UCT Header;SMART_PTR_STRUCT EntryP...
136	MAIN_VECT_2_MUNCH_OBJ_ST			UCT Header;_MAIN_VECT_2_MUNCH_OB...
157	MAIN_VECT_2_JIMMY_STRUCT			CT Jimmy1;MAIN_VECT_2_JIMMY_2_STRU...
158	MAIN_VECT_2_JIMMY_2_STRUC			UCT head;GLOBAL_EVENT_STRUCT_1 Sy...
156	MAIN_VECT_2_JIMMY_1_STRUC			field3;int field4;int field5;int field6;int field...
146	MAIN_VECT_2_IDLER_STRUCT			UCT Header;_MAIN_VECT_2_IDLER_STR...
162	MAIN_VECT_2_GADGET_SUPP_S			UCT Header;EVENT_HANDLE_STRUCT Ev...
163	MAIN_VECT_2_GADGET_SUPP_1			field2;}
172	MAIN_VECT_2_COMMAND_FILE			UCT Header;MAIN_VECT_4_OBJ_HEAD_S...
173	MAIN_VECT_2_COMMAND_FILE			andleInfo;USTRING_PTR_STRUCT FolderN...
164	MAIN_VECT_2_CMD_RUNNER_STRUCT	0000009C	Auto	struct {MAIN_VECT_2_OBJ_HEAD_STRUCT Header;int CmdDispatcher;int Comma...
130	MAIN_VECT_1_STRUCT	000000E4	Auto	struct {int vTable;VECTOR_DATA_1_STRUCT VectorOfCmdDispatchers;GLOBAL_E...
138	MAIN_VECTOR_4_GLOB_STRUCT	00000034	Auto	struct {GLOBAL_EVENT_STRUCT_1 SyncEvent;VECTOR_DATA_1_STRUCT Vector;}

Please enter text

Please edit the type declaration

```
struct MAIN_VECT_2_COMMAND_FILE_FINDER_STRUCT
{
    MAIN_VECT_2_OBJ_HEAD_STRUCT Header;
    MAIN_VECT_4_OBJ_HEAD_STRUCT Main4Head;
    int table;
    VECTOR_DATA_1_STRUCT vect;
    int field0;
    GLOBAL_EVENT_STRUCT_1 sync;
    EVENT_HANDLE_STRUCT EventHandle;
};
```

OK Cancel Help

REconstructing Object's Methods

Please enter text

Please edit the type declaration

```
struct STRUCT_SOCKET_UTABLE
{
    int InitTransport;
    int OpenTransport;
    int CloseTransport;
    int TcpConnect;
    int TcpDisconnect;
    int field5;
    int field6;
    int ReleaseNodeFromList;
    int TcpListen;
    int TcpAccept;
    int TcpSend;
    int TcpReceive;
    int UdpSend;
    int UdpReceive;
    int TcpGetAddrInfo;
    int field15;
    int SetTimeout;
    int SendOverUdp;
    int field18;
    int field19;
    int field20;
};
```

OK Cancel Help

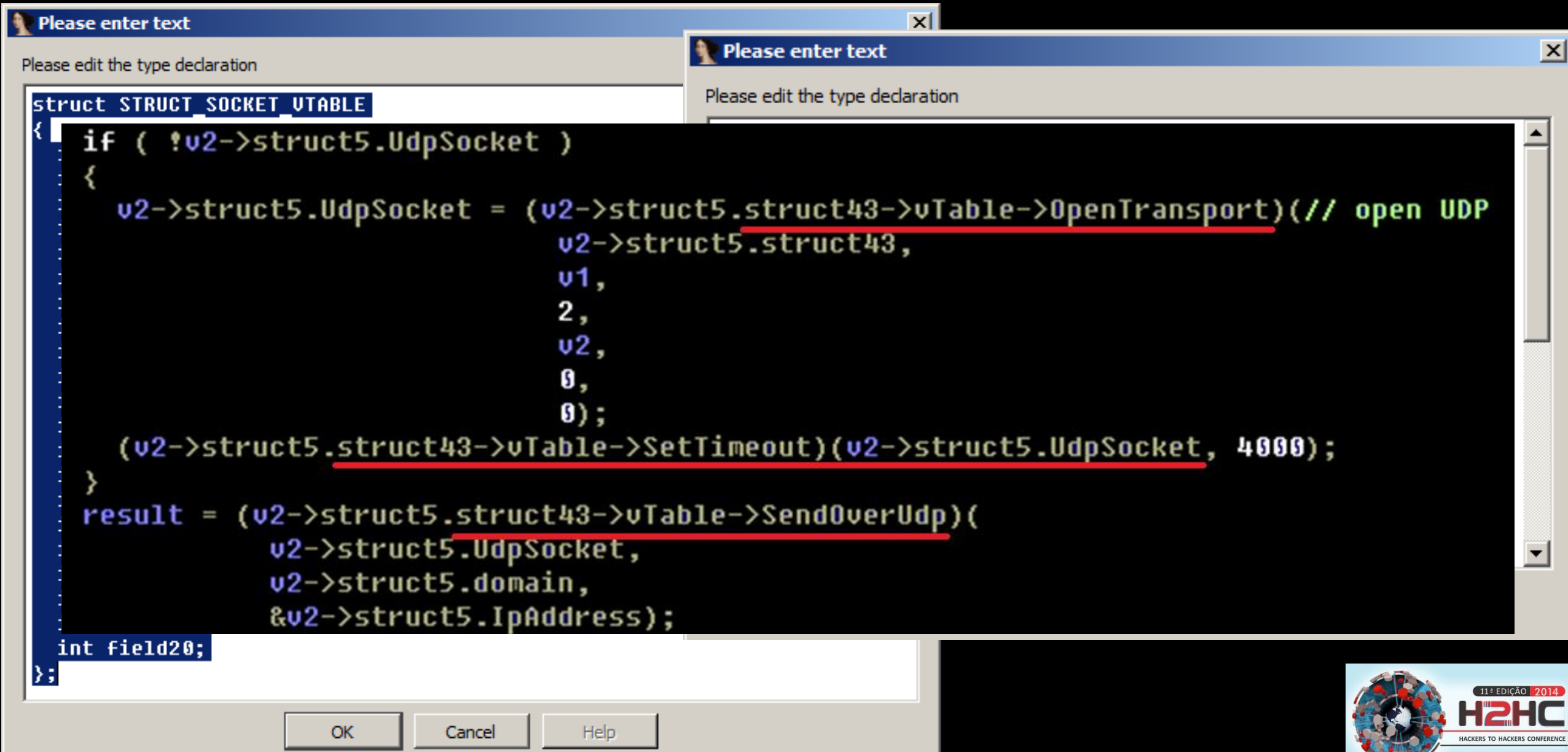
Please enter text

Please edit the type declaration

```
#pragma pack(push, 1)
struct STRUCT_SOCKET
{
    STRUCT_SOCKET_UTABLE *vTable;
    STRUCT_4_4 *struct44;
    int mem1;
    int field5;
    int field6;
    int field7;
    int field8;
    int field9;
    int field10;
    int field11;
    int tcp_drv_ver;
    int field13;
    int field14;
    int field15;
```

OK Cancel Help

Reconstructing Object's Methods

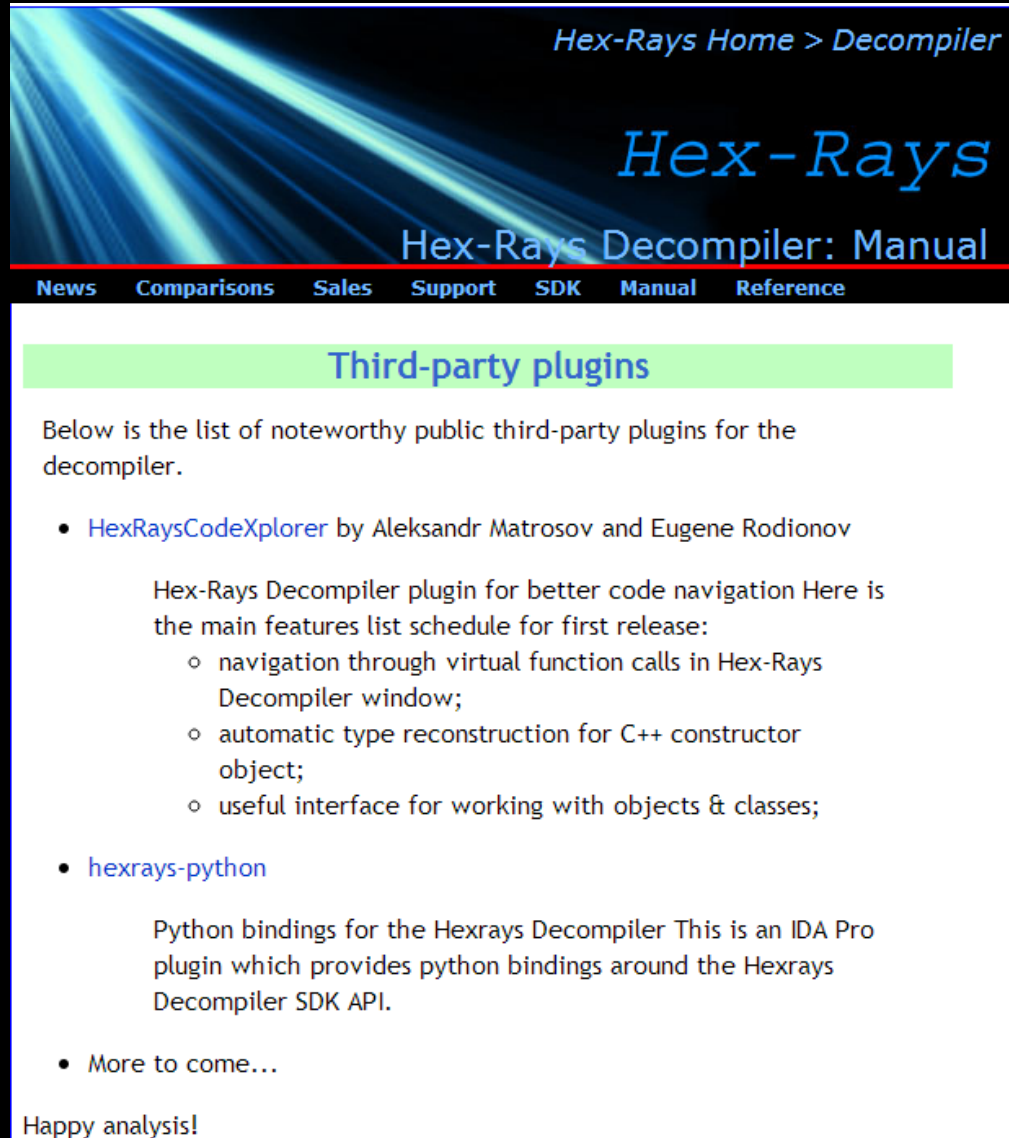


HexRaysCodeXplorer



HexRaysCodeXplorer v1.0:

released in 2013 at REcon



Hex-Rays Home > Decompiler

Hex-Rays

Hex-Rays Decompiler: Manual

[News](#) [Comparisons](#) [Sales](#) [Support](#) [SDK](#) [Manual](#) [Reference](#)

Third-party plugins

Below is the list of noteworthy public third-party plugins for the decompiler.

- [HexRaysCodeXplorer](#) by Aleksandr Matrosov and Eugene Rodionov

Hex-Rays Decompiler plugin for better code navigation Here is the main features list schedule for first release:

- navigation through virtual function calls in Hex-Rays Decompiler window;
- automatic type reconstruction for C++ constructor object;
- useful interface for working with objects & classes;

- [hexrays-python](#)

Python bindings for the Hexrays Decompiler This is an IDA Pro plugin which provides python bindings around the Hexrays Decompiler SDK API.

- More to come...

Happy analysis!

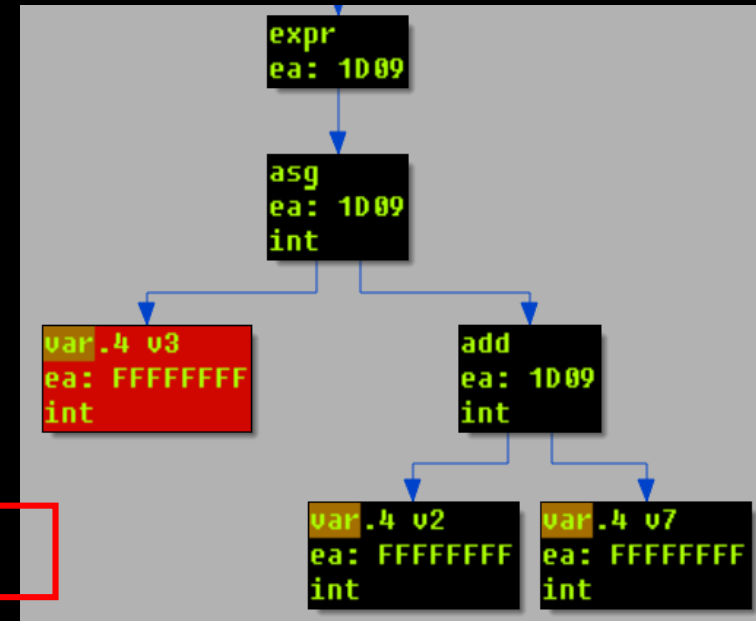
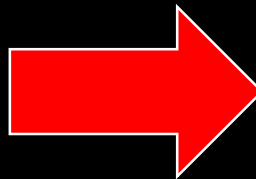
HexRaysCodeXplorer Features

- Hex-Rays decompiler plugin
- The plugin was designed to facilitate static analysis of:
 - ✓ object oriented code
 - ✓ position independent code
- The plugin allows to:
 - ✓ navigate through decompiled virtual methods
 - ✓ partially reconstruct object type

Hex-Rays Decompiler Plugin SDK

- At the heart of the decompiler lies *ctree* structure:
 - ✓ syntax tree structure
 - ✓ consists of *citem_t* objects
 - ✓ there are 9 maturity levels of the *ctree* structure

```
while ( 1 )  
{  
    LOBYTE(v2) = *v4++;  
    if ( !(_BYTE)v2 )  
        break;  
    v7 = ROR4 (v3, 11);  
    v3 = v2 + v7;  
}
```



Hex-Rays Decompiler Plugin SDK

- **At the heart of the decompiler lies *ctree* structure:**

```

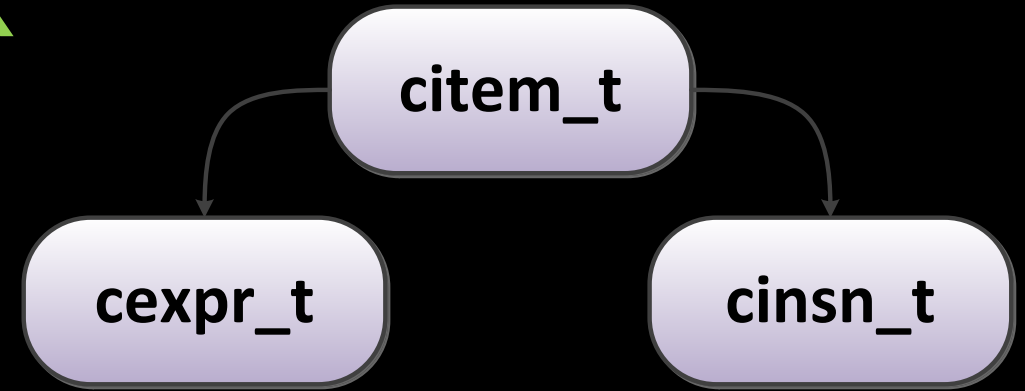
1 /// Ctree maturity level. The level will increase
2 /// as we switch from one phase of ctree generation to the next one
3 enum ctree_maturity_t
4 {
5     CMAT_ZERO,          ///< does not exist
6     CMAT_BUILT,         ///< just generated
7     CMAT_TRANS1,        ///< applied first wave of transformations
8     CMAT_NICE,          ///< nicefied expressions
9     CMAT_TRANS2,        ///< applied second wave of transformations
10    CMAT_CPA,            ///< corrected pointer arithmetic
11    CMAT_TRANS3,         ///< applied third wave of transformations
12    CMAT_CASTED,         ///< added necessary casts
13    CMAT_FINAL,          ///< ready-to-use
14 };

```



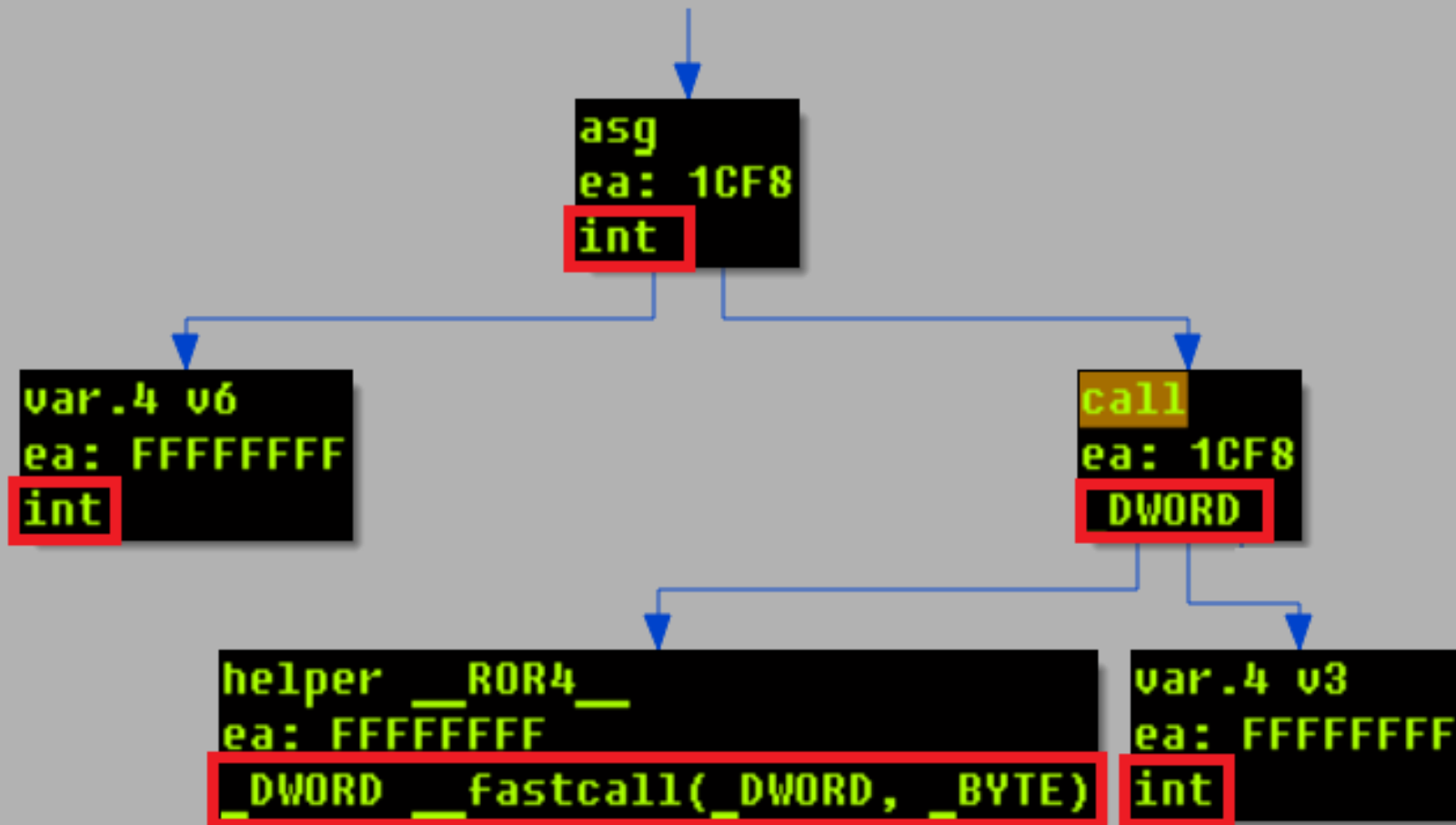
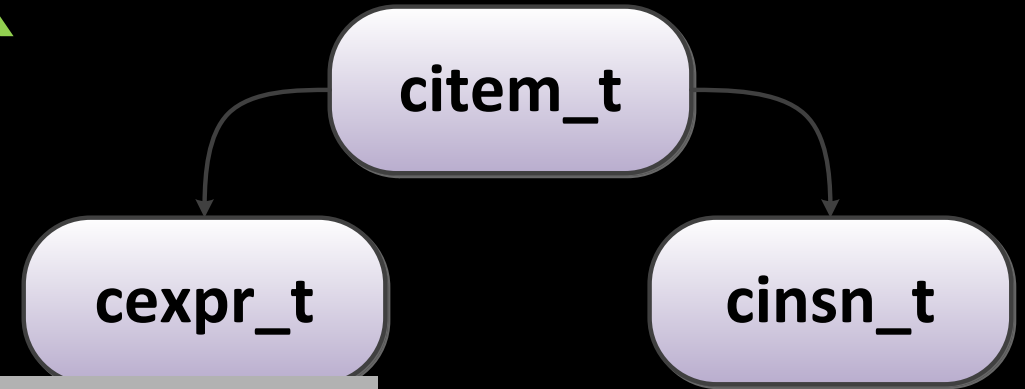
Hex-Rays Decompiler Plugin SDK

- Type *citem_t* is a base class for:
 - ✓ *cexpr_t* – expression type
 - ✓ *cinsn_t* – statement type
- Expressions have attached type information
- Statements include:
 - ✓ *block*, *if*, *for*, *while*, *do*, *switch*, *return*, *goto*, *asm*
- Hex-Rays provides iterators for traversing the *citem_t* objects within *ctree* structure:
 - ✓ *ctree_visitor_t*
 - ✓ *ctree_parentee_t*



Hex-Rays Decompiler Plugin SDK

- Type `citem_t` is a base class for:
 - ✓ `cexpr_t` – expression type
 - ✓ `cinsn_t` – statement type



objects within *ctree*

DEMO time :)



HexRaysCodeXplorer: Gapz Position Independent Code

```
gl_context = (ExAllocatePoolWithTag)(0, 2576, 'ZPAG');  
_gl_context = gl_context;
```



```
v12 = (get_export_by_hash)(kernel_base, hash_ntoskrnl_PsCreateSystemThread, v11);  
v13 = hash_routine;  
_gl_context->PsCreateSystemThread = v12;  
v14 = (get_export_by_hash)(kernel_base, hash_ntoskrnl_PsTerminateSystemThread, v13);  
v15 = hash_routine;  
_gl_context->PsTerminateSystemThread = v14;  
v16 = (get_export_by_hash)(kernel_base, hash_ntoskrnl_KeDelayExecutionThread, v15);  
v17 = hash_routine;  
_gl_context->KeDelayExecutionThread = v16;
```



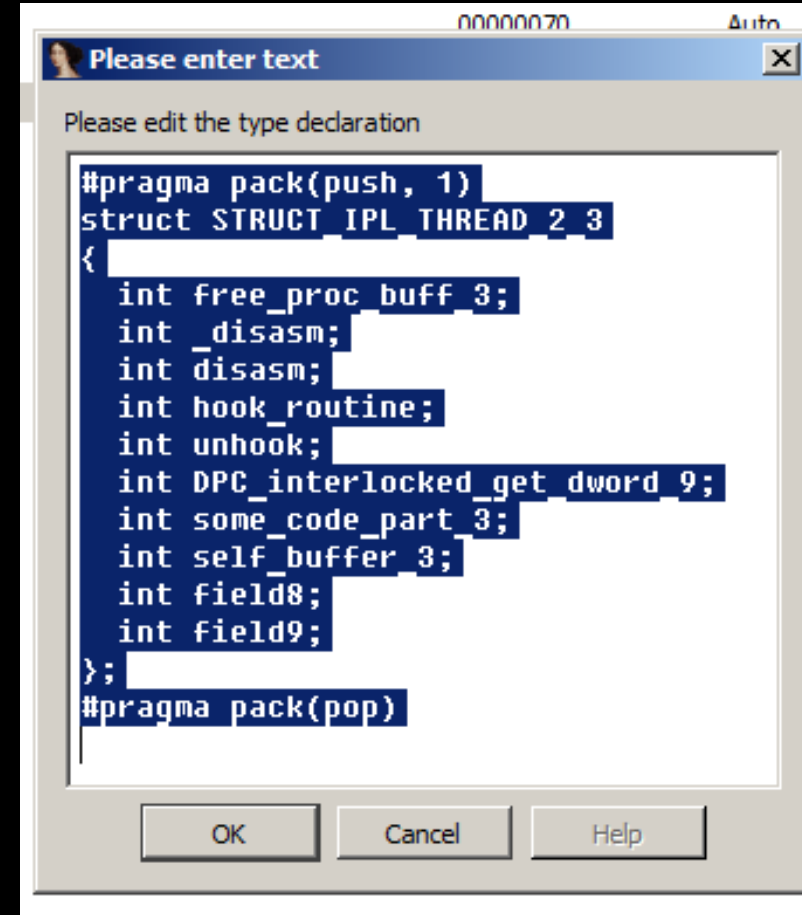
```
_gl_context->ZwOpenSymbolicLinkObject)(&hSymLink, 0x80000000, &v301)
```

HexRaysCodeXplorer: Virtual Methods

- The IDA's 'Local Types' is used to represent object type

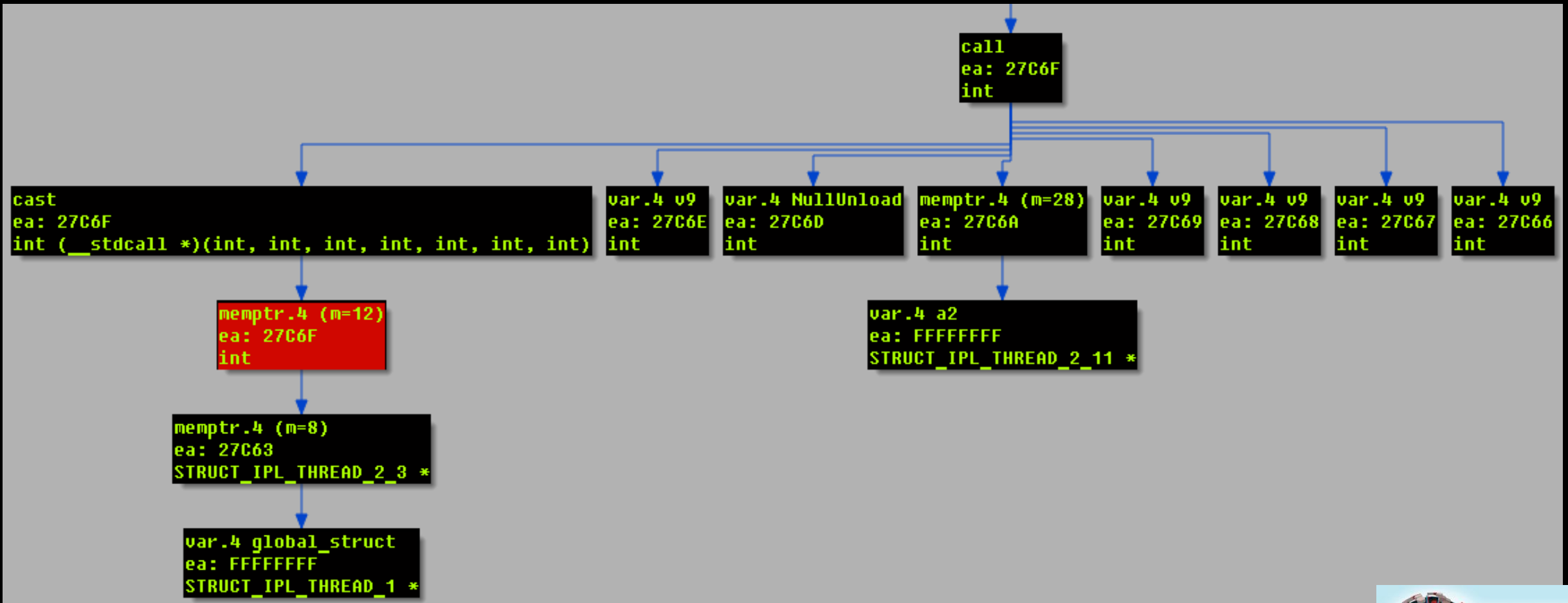
```
int __stdcall block_3_init(STRUCT_IPL_THREAD_2_3 *self_buffer, STRUCT_IPL_THREAD_1 *a2)
{
    STRUCT_IPL_THREAD_2 *v2; // ebx@1
    int _self_buffer; // esi@1
    int (*get_some_code)(void); // edi@1
    STRUCT_IPL_THREAD_2_3 *v5; // eax@1
    int v6; // eax@1
    STRUCT_IPL_THREAD_1 *v7; // ST0C_4@1

    v2 = a2->proc_buffer;
    _self_buffer = self_buffer;
    get_some_code = (&self_buffer[0x36].field8 + -self_buffer->free_proc_buff_3 + 3);
    a2->proc_buffer->alloc_mem(a2->proc_buffer, &self_buffer, 40, 0);
    v5 = self_buffer;
    a2->proc_buff_3 = self_buffer;
    v5->self_buffer_3 = _self_buffer;
    self_buffer->free_proc_buff_3 = _self_buffer - *_self_buffer + 0x112F;
    self_buffer->DPC_interlocked_get_dword_9 = _self_buffer - *_self_buffer + 0xAA7;
    self_buffer->hook_routine = _self_buffer + 0xAF0 - *_self_buffer;
    self_buffer->unhook = _self_buffer + 0xF74 - *_self_buffer;
    self_buffer->_disasm = _self_buffer + 0x388 - *_self_buffer;
    self_buffer->disasm = self_buffer->_disasm;
    v6 = get_some_code();
    v7 = a2;
    self_buffer->some_code_part_3 = v6; // D2B7
    (v2->replace_dword)(_self_buffer + 32, *(_self_buffer + 12), 0xBFFFFFFF, v7);
    return 0;
}
```



HexRaysCodeXplorer: Virtual Methods

- Hex-Rays decompiler plugin is used to navigate through the virtual methods



HexDumpCodeViewer: Virtual Methods

```
a2->bull_unload_hook = (global_struct->proc_buff_3->hook_routine)(  
    v9,  
    NullUnload,  
    a2->Null_unload_hook,  
    v9,  
    v9,  
    v9,  
    v9);
```

cast
ea: 27C6F
int (__stdcall *)(int, int, int, int, int, int, int)

var.4 v9
ea: 27C6E
int

var.4 NullUnload
ea: 27C6D
int

memptr.4 (m=28)
ea: 27C6C
int

var.4 v9
ea: 27C69
int

var.4 v9
ea: 27C68
int

var.4 v9
ea: 27C67
int

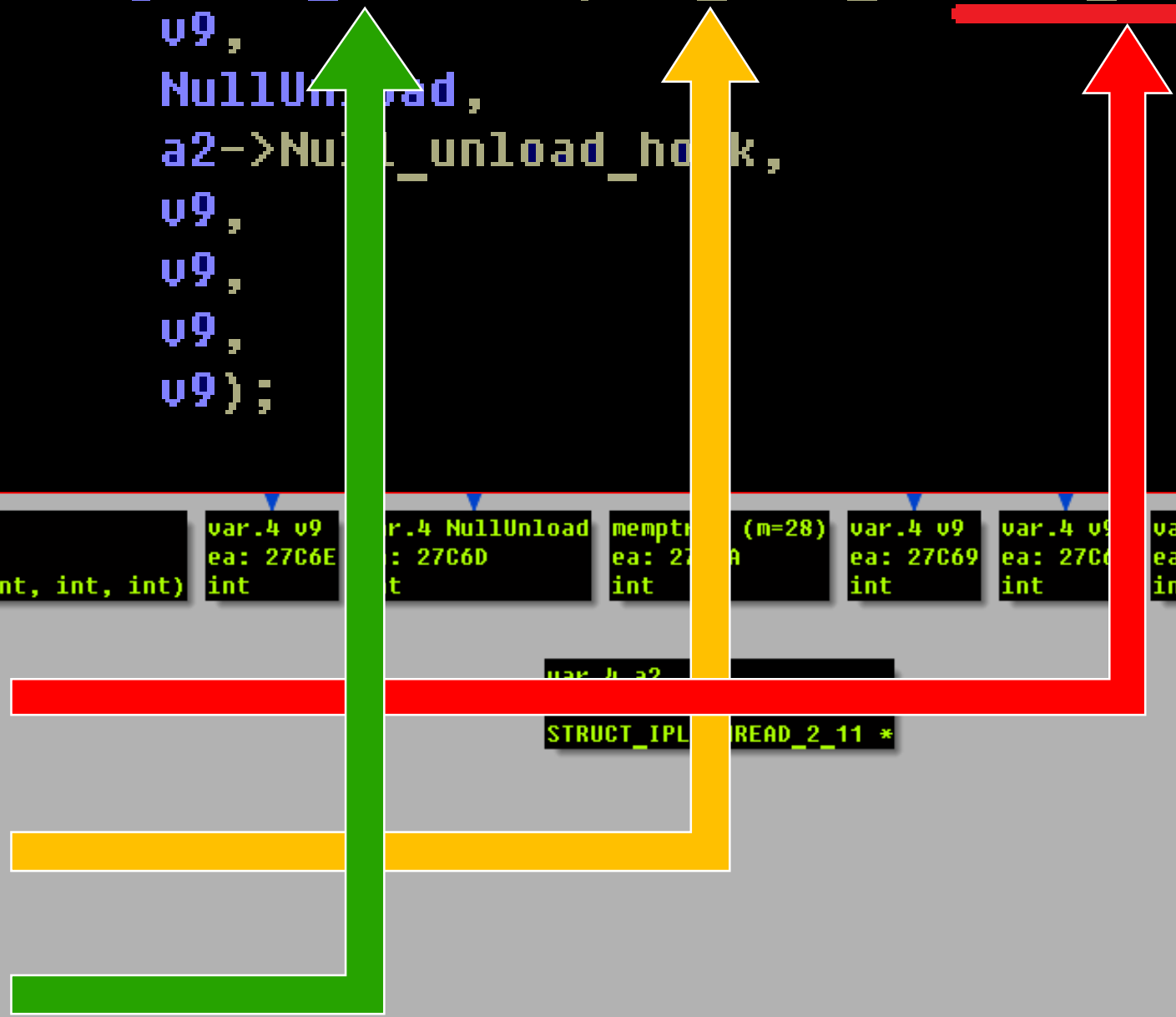
var.4 v9
ea: 27C66
int

memptr.4 (m=12)
ea: 27C6F
int

memptr.4 (m=8)
ea: 27C63
STRUCT_IPL_THREAD_2_3 *

var.4 global_struct
ea: FFFFFFFF
STRUCT_IPL_THREAD_1 *

var.4 a2
STRUCT_IPL_THREAD_2_11 *



HexRaysCodeXplorer: Object Type REconstruction

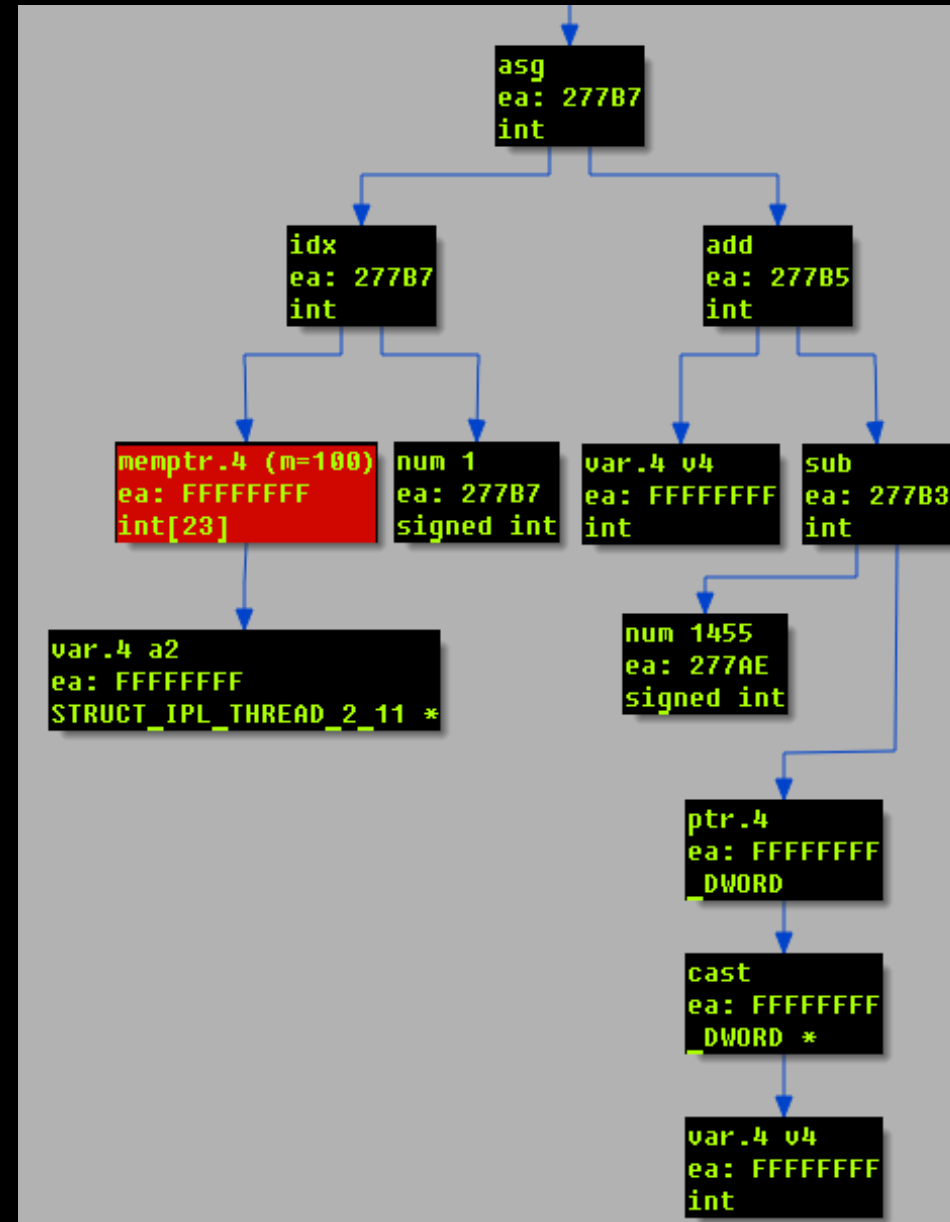
- Hex-Rays's *ctree* structure may be used to partially reconstruct object type based on its initialization routine (constructor)
- Input:
 - ✓ pointer to the object instance
 - ✓ object initialization routine entry point
- Output:
 - ✓ C structure-like object representation

HexRaysCodeXplorer: Object Type REconstruction

➤ citem_t objects to monitor:

- ✓ memptr ✓ call (LOBYTE, etc.)
- ✓ idx
- ✓ memref

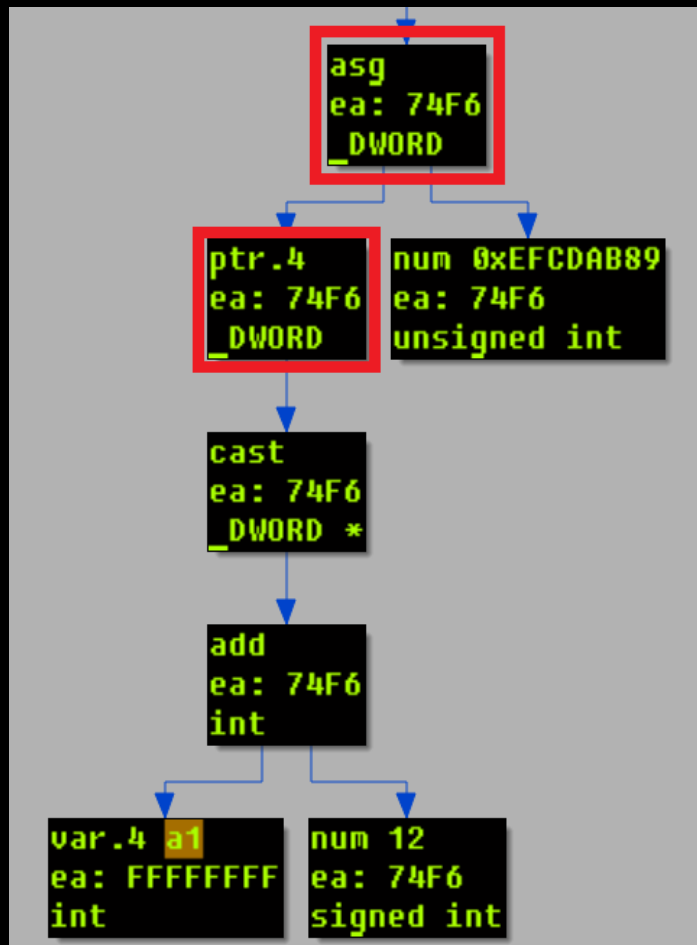
```
a2->IoControlCode_HookArray[1] = 0xFFDC243F;  
a2->IoControlCode_HookDpc[2] = v4 + 1524 - *v4;  
a2->IoControlCodeSubCmd_Hook[2] = 12;  
a2->IoControlCode_HookArray[2] = 0xFFDC2437;  
a2->IoControlCode_HookDpc[3] = v4 + 1586 - *v4;  
a2->IoControlCodeSubCmd_Hook[3] = 2;  
a2->IoControlCode_HookArray[3] = 0xFFDC240B;  
a2->IoControlCode_HookDpc[4] = v4 + 1659 - *v4;  
a2->IoControlCodeSubCmd_Hook[4] = 13;  
a2->IoControlCode_HookArray[4] = 0xFFDC243B;  
a2->IoControlCode_HookDpc[5] = v4 + 1726 - *v4;  
a2->IoControlCodeSubCmd_Hook[5] = 3;  
a2->IoControlCode_HookArray[5] = 0xFFDC240F;  
a2->IoControlCode_HookDpc[6] = v4 - *v4 + 1799;  
a2->IoControlCodeSubCmd_Hook[6] = 10;  
a2->IoControlCode_HookArray[6] = 0xFFDC242F;
```



HexRaysCodeXplorer: Object Type REconstruction

// reference of DWORD at offset 12 in buffer a1

(DWORD)(a1 + 12) = 0xEFCDAB89;



```
20 a2->free_mem = v4 - *v4 + 0x7D1E;
21 a2->base64_encode = v4 + 0x388 - *v4;
22 a2->base64_decode = v4 + 0x4CD - *v4;
23 a2->rdtsc_proc = v4 - *v4 + 0x579F;
24 a2->rnd_process_block = v4 + 0x57A2 - *v4;
25 a2->rnd_fill_buffer = v4 - *v4 + 0x6A87;
26 a2->init_rnd_buffer = v4 + 0x6ABB - *v4;
27 a2->field13 = v4 + 0x4B95 - *v4;
28 a2->md5_init = v4 - *v4 + 0x2A2C;
```

Output window

```
Field reference detected -> Offset 11210 : char
Field reference detected -> Offset 11217 : char
Field reference detected -> Offset 11218 : char
Field reference detected -> Offset 11219 : char
Field reference detected -> Offset 11220 : char
Field reference detected -> Offset 11221 : char
Field reference detected -> Offset 11222 : char
struct STRUCTURE_TYPE {
    int field_0;
    int field_1;
    int field_2;
    int field_3;
    int field_4;
    int field_5;
    int field_6;
    int field_7;
    int field_8;
    int field_9;
    int field_10;
    int field_11;
```

Python

HexRaysCodeXplorer v1.5 [H2HC Edition]

○ New *citem_t* objects to monitor:

- ✓ memptr
- ✓ idx
- ✓ memref
- ✓ call (LOBYTE, etc.)
- ✓ ptr, asg, ...

```
v47 = 0;  
v48 = 0;  
v49 = 0;  
v2 = *(void **)(a2 + 24);  
v55 = 0;
```



```
ptr.4  
ea: 17087444  
void *
```

```
cast  
ea: 17087444  
void **
```

```
add  
ea: 17087444  
int
```

```
var.4 a2  
ea: FFFFFFFF  
int
```

```
num 24  
ea: 17087444  
signed int
```

```
v2 = *(_DWORD *)(this + 216);  
v3 = this + 212;  
v4 = *(_DWORD *)(this + 216) + 1;
```



```
asg  
ea: 170D4FB5  
int
```

```
var.4 v3  
ea: FFFFFFFF  
int
```

```
add  
ea: 170D4FB5  
int
```

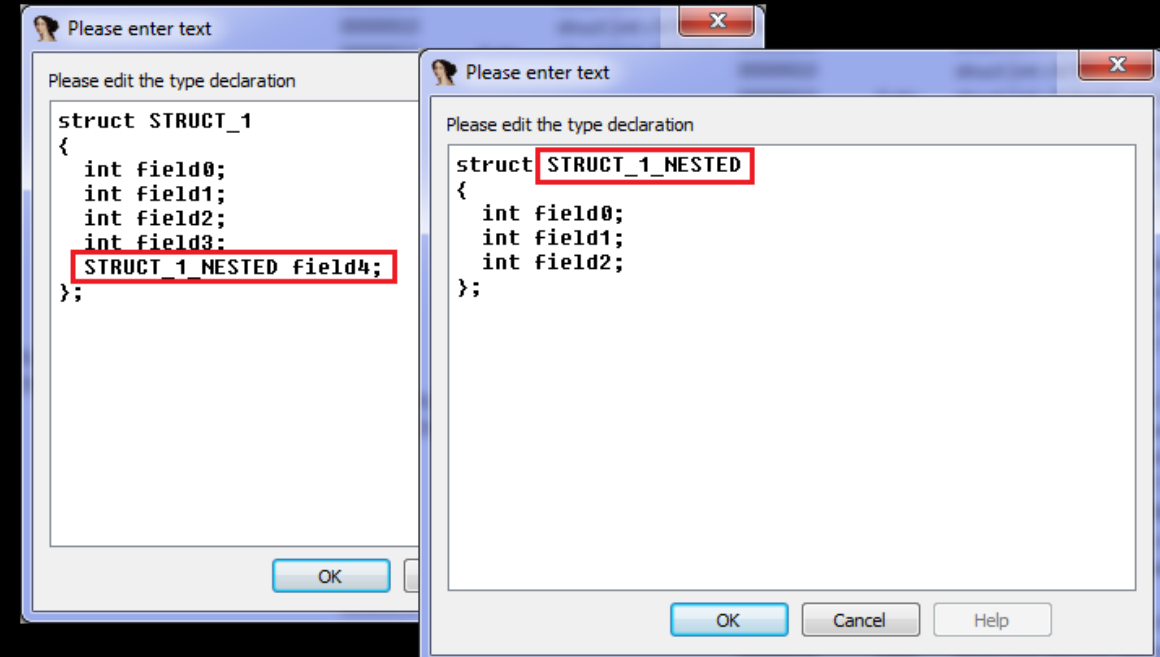
```
var.4 this  
ea: FFFFFFFF  
int
```

```
num 212  
ea: 170D4FB5  
signed int
```

HexRaysCodeXplorer v1.5 [H2HC Edition]

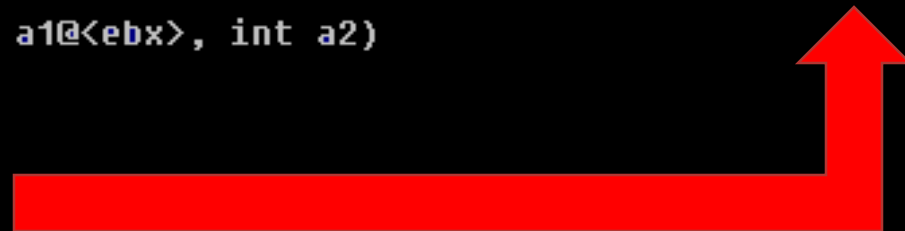
○ New *citem_t* objects to monitor:

- ✓ **memptr**
- ✓ **idx**
- ✓ **memref**
- ✓ **call (LOBYTE, etc.)**
- ✓ **ptr, asg, ...**



○ Type propagation for nested function calls

```
int __userpurge sub_1003490E0<eax>(int a1@<ebx>, int a2)
{
    *(_DWORD *)a2 = off_1026DDCC;
    *(_DWORD *)a2 = &off_102691D0;
    sub_10037E79(a1 + 4, a2 + 4);
    sub_10003F6B(a1 + 20, a2 + 20);
    return a2;
}
```



HexRaysCodeXplorer v1.5 [H2HC Edition]

○ Features of v1.5 [H2HC Edition] :

- ✓ Better Type Reconstruction
 - Improvements for parsing *citem_t* objects with PTR and ASG statements
 - Recursive traversal of Ctree to reconstruct Types hierarchy
- ✓ Navigate from Pseudo code window to Disassembly line
- ✓ Hints for Ctree elements which point to Disassembly line
- ✓ Support for x64 version of Hex-Rays Decompiler
- ✓ Some bug fixes by user requests

DEMO time :)



HexRaysCodeXplorer:

-> What are the next goals?

- Develop the next version on IdaPython
- Focus on the following features:
 - ✓ Type reconstruction (C++, Objective-C)
 - ✓ Type Navigation (C++, Objective-C)
 - ✓ Vtables parsing based on Hex-Rays API
 - ✓ Ctree graph navigation improvements
 - ✓ Patterns for possible vuln detection



Why python?

```
import idaapi

class CTreeVisitor(idaapi.ctree_visitor_t):
    def __init__(self, dumper, cfunc):
        idaapi.ctree_visitor_t.__init__(self, idaapi.CV_FAST | idaapi.CV_INSNS)
        self.dumper = dumper
        self.cfunc = cfunc
        return

    def visit_insn(self, ins):
        print ins.opname
        return 0

class CDumper(object):
    def __init__(self):
        self.ret = {}

    def dump(self, ea):
        f = idaapi.get_func(ea)
        cfunc = idaapi.decompile(f)
        visitor = CTreeVisitor(self, cfunc)
        visitor.apply_to(cfunc.body, None)

def main():
    dump = CAstDumper()
    dump.dump(here())

if __name__ == "__main__":
    main()
```

```
block
expr
expr
if
block
expr
expr
block
expr
expr
expr
expr
expr
expr
expr
expr
expr
expr
if
block
expr
expr
block
expr
expr
expr
expr
expr
expr
expr
expr
expr
return
```



Python Arsenal Contest

from Russia with 0-days



zeronights.org

- ❑ Best exploit dev tool/plugin/lib
- ❑ Best forensics tool/plugin/lib
- ❑ Best reversing tool/plugin/lib
- ❑ Best fuzzing tool/plugin/lib
- ❑ Best malware analysis tool/plugin/lib

<http://2014.zeronights.org/contests/python-arsenal-contest.html>

The background of the slide is a grayscale image of a circuit board. It features a complex network of black lines representing traces, with several circular pads and vias. The pattern is symmetrical and fills the entire frame.

Thank you for your attention!

HexRaysCodeXplorer

<http://REhints.com>

@REhints

<https://github.com/REhints/HexRaysCodeXplorer>