Hex-Rays Decompiler plugin for better code navigation — Edit

README.md

```
 _   _         _____              ____          _    __   __      _
| | | |        | ___ \            /  __ \        | |    \ \ / /     | |
| |_| | _____  _| |_/ /__ _ _   _ ___| /  \/ ___   __| | ___ \ V / _ __ | | ___  _ __ ___ _ __
|  _  |/ _ \ \/ /    // _` | | | / __| |    / _ \ / _` |/ _ \/   \| '_ \| |/ _ \| '__/ _ \ '__|
| | | |  __/>  <| |\ \ (_| | |_| \__ \ \__/\ (_) | (_| |  __/ /^\ \ |_) | | (_) | | |  __/ |
\_| |_/\___/_/\_\_| \_\__,_|\__, |___/\____/\___/ \__,_|\___\/   \/ .__/|_|\___/|_|  \___|_|
                              __/ |                               | |
                             |___/                                |_|
```

The Hex-Rays Decompiler plugin for better code navigation in RE process. CodeXplorer automates code REconstruction of C++ applications or modern malware like Stuxnet, Flame, Equation, Animal Farm ...

The CodeXplorer plugin is one of the first publicly available Hex-Rays Decompiler plugins. We keep updated this project since summer of 2013 and continue contributing new features frequently. Also most interesting feutures of CodeXplorer have been presented on numerous security conferences like: REcon, ZeroNights, H2HC, NSEC and BHUS

Contributors:

Alex Matrosov (@matrosov)

Eugene Rodionov (@rodionov)

Rodrigo Branco (@rrbranco)

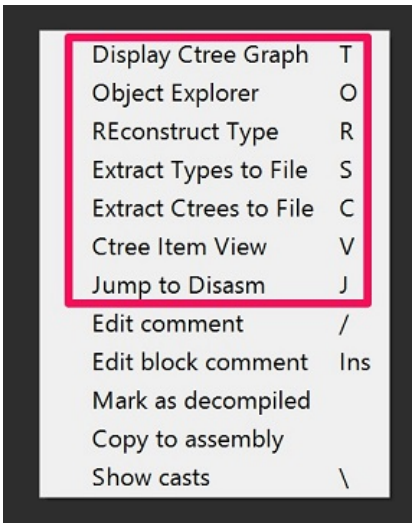
Gabriel Barbosa (@gabrielnb)

Supported versions of Hex-Rays products: everytime we focus on last versions of IDA and Decompiler because trying to use new interesting features in new SDK releases. It's also mean we tested just on last versions of Hex-Rays products and not guaranteed stable work on previous ones.

Why not IdaPython: all code developed on C/C++ because it's more stable way to support complex plugin for Hex-Rays Decompiler.

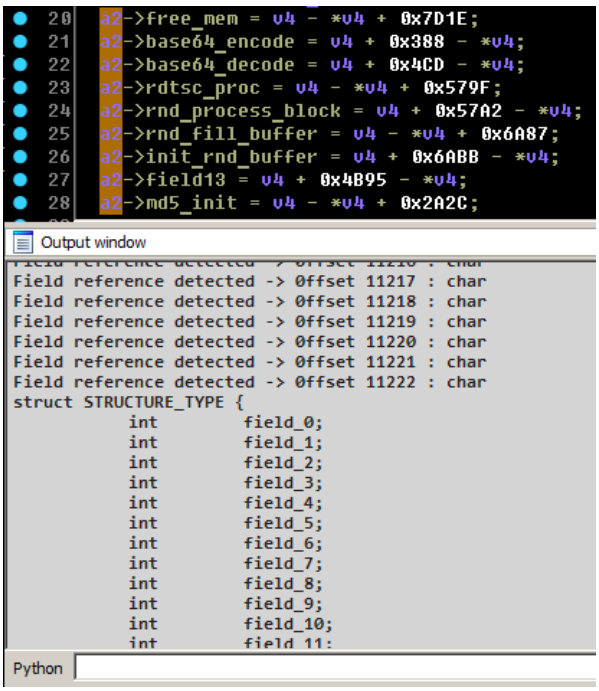
Supported Platforms: x86/x64 for Win, Linux and Mac.

HexRaysCodeXplorer - Hex-Rays Decompiler plugin for easier code navigation. Right-click context menu in the Pseudocode window shows CodeXplorer plugin commands:



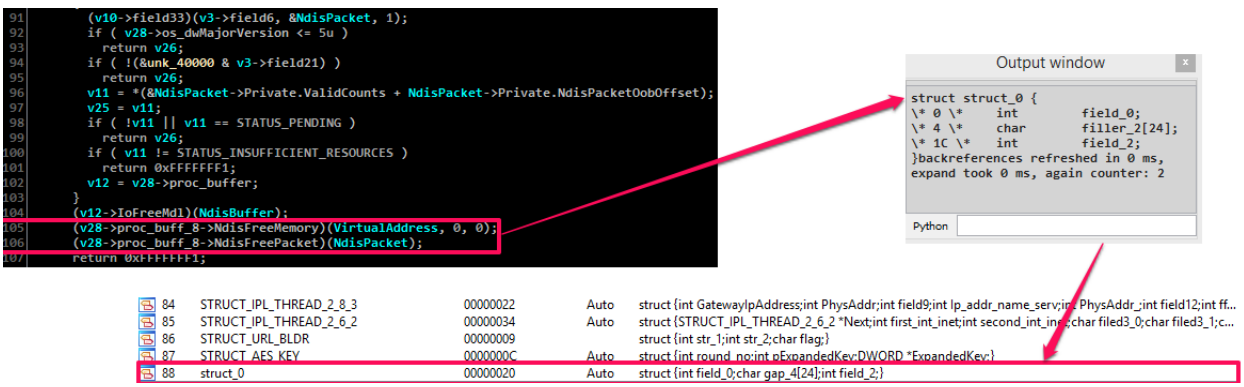
💎 **Here are the main features of the CodeXplorer plugin:** 💎

- ***Automatic type REconstruction*** for C++ objects. To be able to reconstruct a type using HexRaysCodeXplorer one needs to select the variable holding pointer to the instance of position independed code or to an object and by right-button mouse click select from the context menu «REconstruct Type» option:

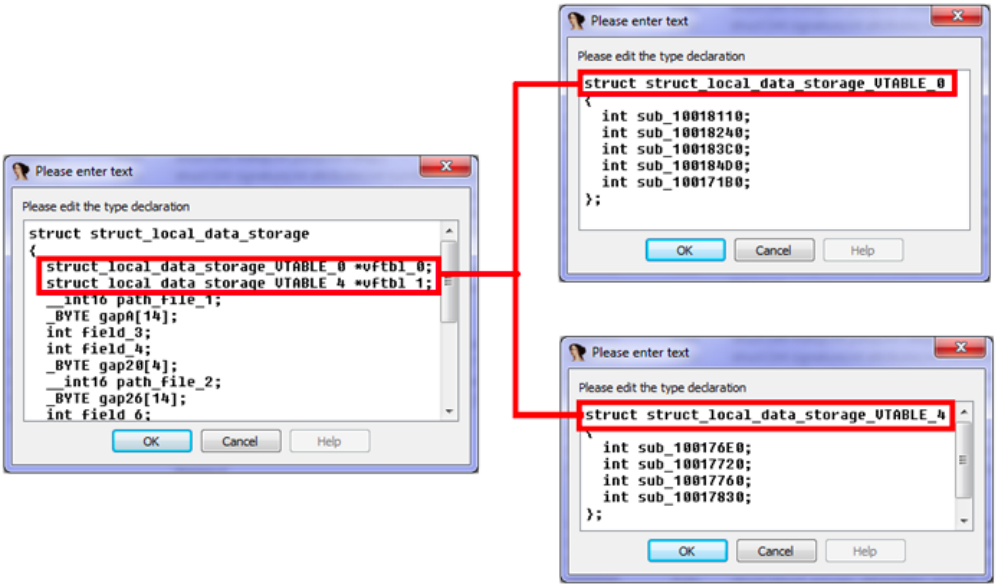


The reconstructed structure is displayed in “Output window”. Detailed information about type Reconstruction feature is provided in the blog post “Type REconstruction in HexRaysCodeXplorer”.

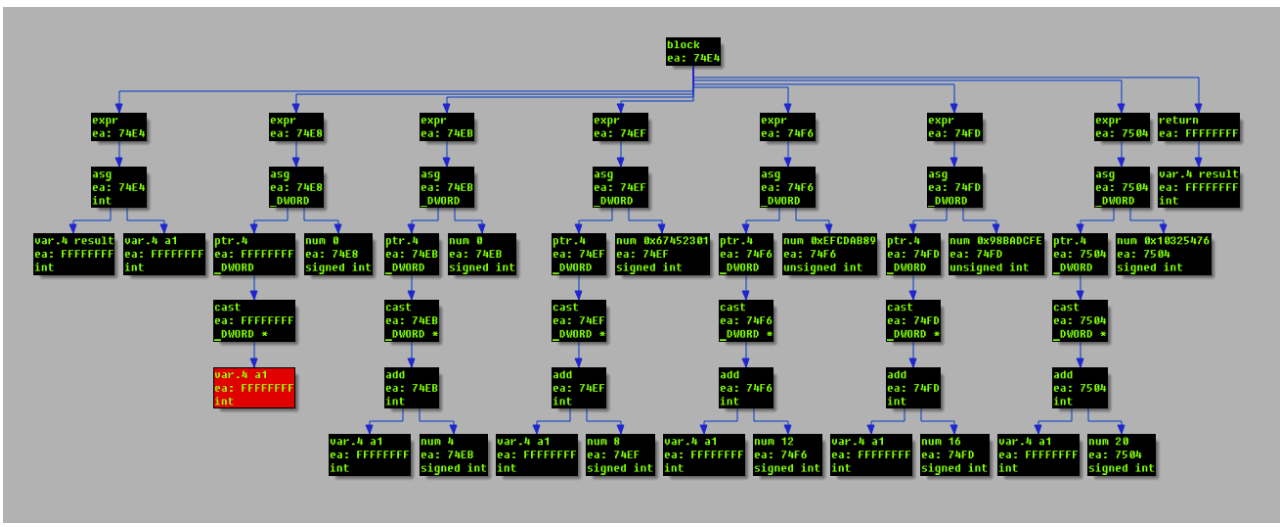
Also CodeXplorer plugin supports auto REconstruction type into IDA local types storage.



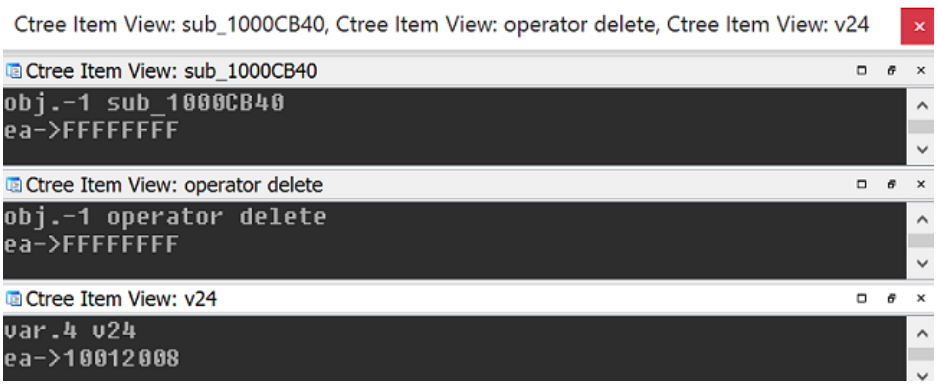
- ***Virtual function table identification*** - automatically identifies references to virtual function tables during type reconstruction. When a reference to a virtual function table is identified the plugin generates a corresponding C-structure. As shown below during reconstructing `struct_local_data_storage` two virtual function tables were identified and, as a result, two corresponding structures were generated: `struct_local_data_storage_VTABLE_0` and `struct_local_data_storage_VTABLE_4`.



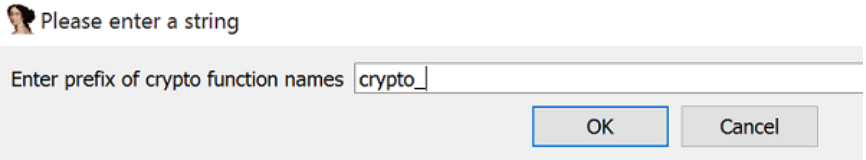
- ***C-tree graph visualization*** – a special tree-like structure representing a decompiled routine in citem_t terms (hexrays.hpp). Useful feature for understanding how the decompiler works. The highlighted graph node corresponds to the current cursor position in the HexRays Pseudocode window:



- ***Ctree Item View*** – show ctree representation for highlighted element:



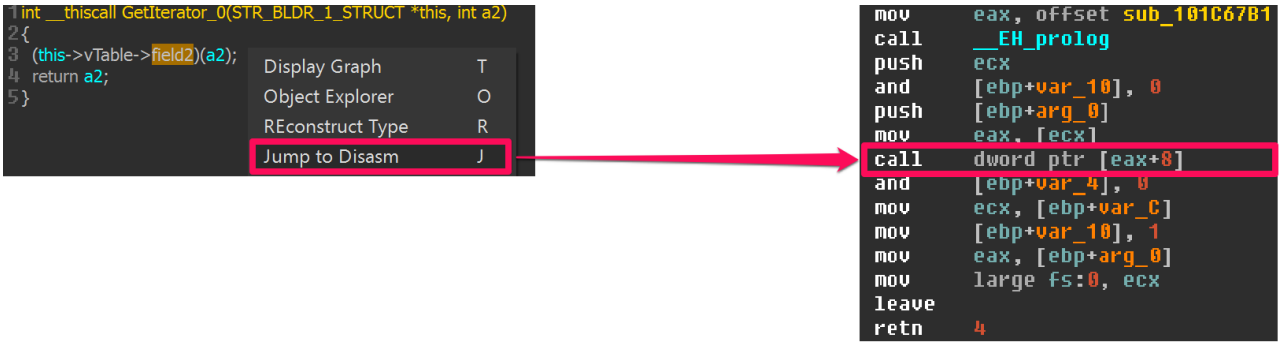
- ***Extract Ctrees to File*** – dump calculate SHA1 hash and dump all ctrees to file.



- ***Extract Types to File*** – dump all types information (include reconstructed types) into file.
- ***Navigation through virtual function calls*** in HexRays Pseudocode window. After representing C++ objects by C-structures this feature make possible navigation by mouse clicking to the virtual function calls as structure fields:

```
v9 = v8->LowestDevInStack;
if ( v9 )
{
  a2->internal_control_hook_info = (v2->proc_buff_3->hook_routine)(
                                     0,
                                     v9->DriverObject->MajorFunction[IRP_MJ_INTERNAL_DEVICE_CONTROL],
                                     v2->proc_buff_5->IrpMjInternalControl_Hook,
                                     0,
                                     0,
                                     0,
                                     a2->save_tramploine_info);
  if ( v8->LowestDevInStack->DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] == v8->LowestDevInStack->DriverObject->MajorFunction[15] )
    a2->devicel_control_hook_info = a2->internal_control_hook_info;
  else
    a2->devicel_control_hook_info = (v2->proc_buff_3->hook_routine)(
                                       0,
                                       v8->LowestDevInStack->DriverObject->MajorFunction[14],
                                       v2->proc_buff_5->IrpMjDeviceControl_Hook,
                                       0,
                                       0,
                                       0,
                                       a2->save_tramploine_info);
}
```

- ***Jump to Disasm*** - small feature for navigate to assembly code into "IDA View window" from current Pseudocode line position. It is help to find a place in assembly code associated with decompiled line.



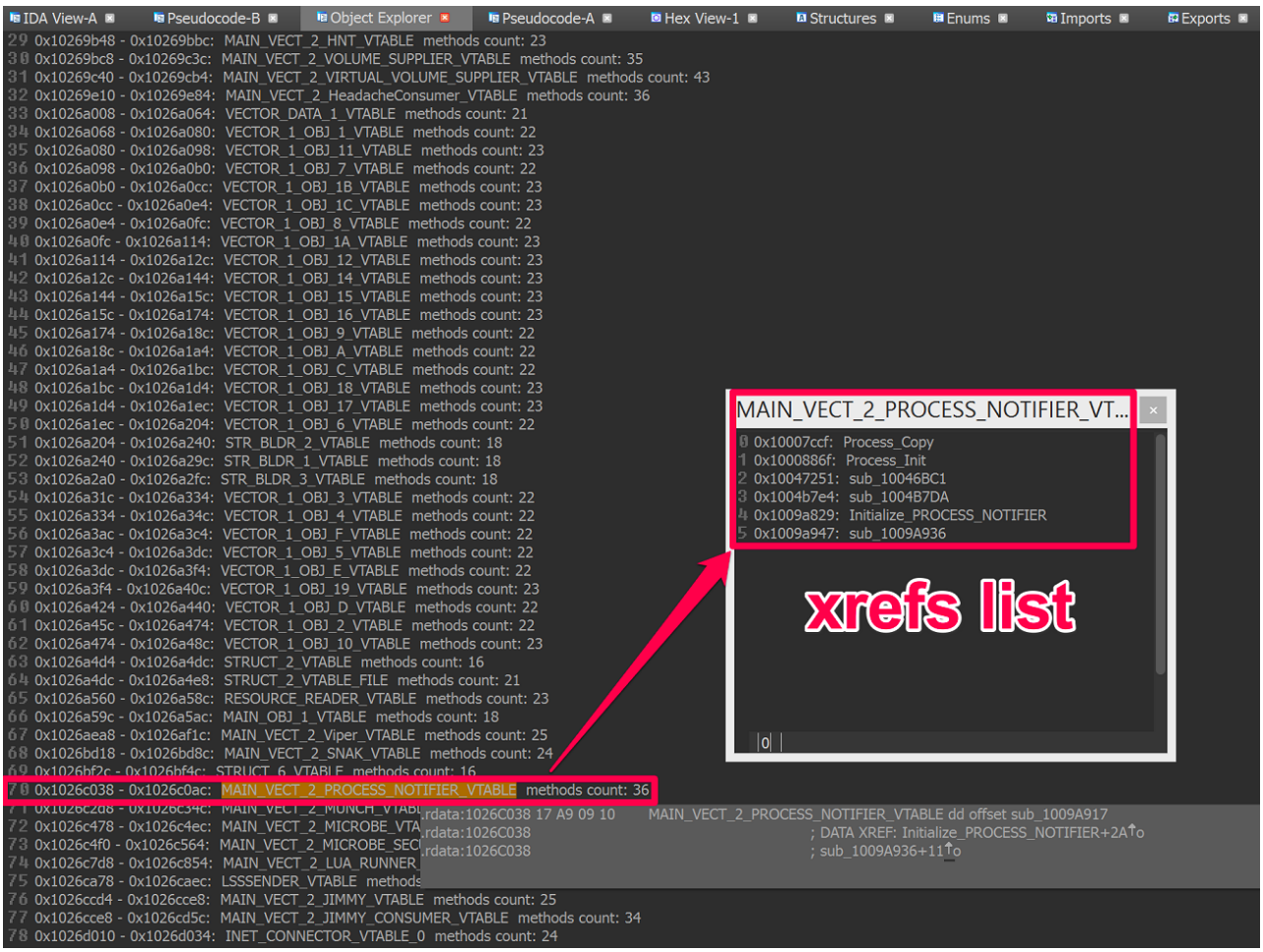
- ***Object Explorer*** – useful interface for navigation through virtual tables (VTBL) structures. Object Explorer outputs VTBL information into IDA custom view window. The output window is shown by choosing «Object Explorer» option in right-button mouse click context menu:

Object Explorer

```
 6 0x102568b8 - 0x102568c0:  .?AVrhf_yknwaztep@bht@: .?AVdjyynplwo@mzh@  methods count:
 7 0x10256928 - 0x1025695c:  off_10256928  methods count: 13
 8 0x102569a0 - 0x102569fc:  off_102569A0  methods count: 23
 9 0x10256a00 - 0x10256a40:  off_10256A00  methods count: 16
10 0x10256a40 - 0x10256a9c:  off_10256A40  methods count: 23
11 0x10256aa0 - 0x10256afc:  VECTOR_DATA_2_VTABLE  methods count: 23
12 0x10256b48 - 0x10256b68:  off_10256B48  methods count: 8
13 0x10256b68 - 0x10256b88:  off_10256B68  methods count: 8
14 0x10256b88 - 0x10256ba8:  off_10256B88  methods count: 8
15 0x10256ba8 - 0x10256bb0:  off_10256BA8  methods count: 2
16 0x10256bb0 - 0x10256bd8:  FILE_MAPPING_1_VTABLE  methods count: 10
17 0x10256bd8 - 0x10256bf0:  GLOBAL_EVENT_1_VTABLE  methods count: 6
18 0x10267910 - 0x1026796c:  off_10267910  methods count: 23
19 0x10267978 - 0x10267980:  off_10267978  methods count: 2
20 0x102679a0 - 0x102679f0:  PROCESS_HANDLE_VTABLE  methods count: 20
21 0x102679f0 - 0x10267a1c:  off_102679F0  methods count: 11
22 0x10267a1c - 0x10267a48:  off_10267A1C  methods count: 11
23 0x10267a50 - 0x10267a90:  off_10267A50  methods count: 16
24 0x10267a90 - 0x10267acc:  THREAD_HANDLE_VTABLE  methods count: 15
25 0x10267acc - 0x10267aec:  off_10267ACC  methods count: 8
26 0x10267aec - 0x10267af4:  off_10267AEC  methods count: 2
27 0x10267afc - 0x10267b04:  off_10267AFC  methods count: 2
28 0x10267b08 - 0x10267b7c:  FILE_VTABLE_0  methods count: 29
```

Object Explorer supports following features:

- Auto structures generation for VTBL into IDA local types
- Navigation in virtual table list and jump to VTBL address into "IDA View" window by click
- Show hints for current position in virtual table list
- Shows cross-references list by click into menu on "Show XREFS to VTBL"



- Support auto parsing RTTI objects:

```
.rdata:100347F4 ; CClassFactory::`RTTI Base Class Array'
.rdata:100347F4 ??_R2CClassFactory@@8 dd offset ??_R1A@?0A@AE@CClassFactory@@8
.rdata:100347F4                                         ; DATA XREF: .rdata:CClassFactory::`RTTI Class Hierarchy Descriptor'↑o
.rdata:100347F4                                         ;   BaseClass[0]
.rdata:100347F8                 dd offset ??_R1A@?0A@AE@?$CUnknown@UIClassFactory@@@@8 ;   BaseClass[1]
.rdata:100347FC                 dd offset ??_R1A@?0A@AE@IClassFactory@@8 ;   BaseClass[2]
.rdata:10034800                 dd offset ??_R1A@?0A@AE@IUnknown@@8 ;   BaseClass[3]
.rdata:10034804                 dd 0
.rdata:10034808 ; CClassFactory::`RTTI Base Class Descriptor at (0, -1, 0, 4)'
.rdata:10034808 ??_R1A@?0A@AE@CClassFactory@@8 _RTTIBaseClassDescriptor <offset ??_R0?AVCClassFactory@@@@8, 3, <0, \
.rdata:10034808                                         ; DATA XREF: .rdata:CClassFactory::`RTTI Base Class Array'↑o
.rdata:10034808                                           0FFFFFFFFh, 0>, 40h>; RTTI Base Class Descriptor (#classinformer) ; CClassFactory `RTTI Type De
.rdata:10034820                 dd offset ??_R3CClassFactory@@8 ; CClassFactory::`RTTI Class Hierarchy Descriptor'
.rdata:10034824 ; const CEventSink::`RTTI Complete Object Locator'
.rdata:10034824 ??_R4CEventSink@@6B@ _RTTICompleteObjectLocator <0, 0, 0, offset ??_R0?AVCEventSink@@@@8, \
.rdata:10034824                                         ; DATA XREF: .rdata:10030A0C↑o
.rdata:10034824                                             offset ??_R3CEventSink@@8>; RTTI Complete Object Locator (#classinformer) ; CEventSink::`RTTI
.rdata:10034838 ; CEventSink::`RTTI Class Hierarchy Descriptor'
.rdata:10034838 ??_R3CEventSink@@8 _RTTIClassHierarchyDescriptor <0, 0, 4, offset ??_R2CEventSink@@8>; RTTI Class Hierarchy Descriptor (#classinformer)
.rdata:10034838                                         ; DATA XREF: .rdata:const CEventSink::`RTTI Complete Object Locator'↑o
.rdata:10034838                                         ; .rdata:10034874↓o
.rdata:10034838                                         ; CEventSink::`RTTI Base Class Array'
```

The Batch mode contains following features:

- Batch mode - useful feature to use CodeXplorer for processing multiple files without any interaction from user. We add this feature after Black Hat research in 2015 for processing 2 millions samples.

```
Example (dump types and ctrees for functions with name prefix "crypto_"):
idaq.exe -OHexRaysCodeXplorer:dump_types:dump_ctrees:CRYPTOcrypto_path_to_idb
```

Conference talks about CodeXplorer plugin:

- **2015**
  - "Distributing the REconstruction of High-Level IR for Large Scale Malware Analysis", BHUS [slides]
  - "Object Oriented Code RE with HexraysCodeXplorer", NSEC [slides]
- **2014**
  - "HexRaysCodeXplorer: object oriented RE for fun and profit", H2HC [slides]
- **2013**
  - "HexRaysCodeXplorer: make object-oriented RE easier", ZeroNights [slides]
  - "Reconstructing Gapz: Position-Independent Code Analysis Problem", REcon [slides]