simpliFiRE.IDAscope

An IDA Pro extension for easier (malware) reverse engineering

Daniel Plohmann, Alexander Hanel

plohmann@cs.uni-bonn.de
alexander.hanel@googlemail.com

Fraunhofer
FKIE

Some words about myself

- Personal background
  - PhD student and researcher at University of Bonn & Fraunhofer FKIE
    - Research focus: Reverse Engineering
    - Work focus: malware analysis and botnet mitigation

- Projects
  - Author of 2011 ENISA Botnet Study [1]
  - PyBox [2]
    - Userland-hooking framework (with Felix Leder)
  - AntiRE [3]
    - An Executable Collection of Anti-Reversing Techniques

PYTHON SANDBOX

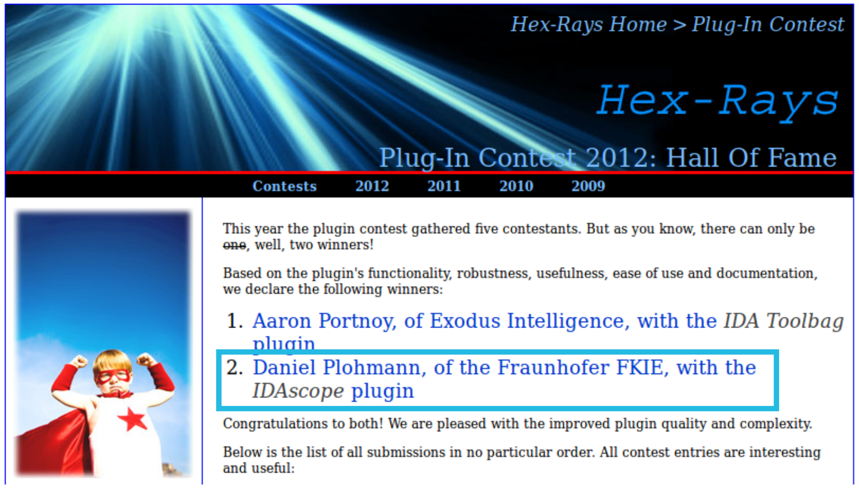
[1] http://www.enisa.europa.eu/act/res/botnets/botnets-measurement-detection-disinfection-and-defence
[2] http://code.google.com/p/pyboxed [3] https://bitbucket.org/fkie_cd_dare/simplifire.antire

simpliFiRE.IDAscope

Current State

IDAscope
… in a nutshell

- An IDA Pro extension for easier (malware) reverse engineering.
- Motivated by the current workflow of working with IDA Pro.
  - Repeat: „Identify relevant parts of the binary; tear apart; document findings.“
- Common tasks:
  - 1 Malware RE usually starts with the corner pieces: strings, API calls, signature hits, …
    - API calls are a good indicator for function semantics.
  - 2 Reoccurring need for looking up things in MSDN.
    - Switch windows time and time again…
  - 3 C&C communication schemes are of high interest!
    - Find and understand cryptographic routines used.
- Idea:
  - Provide automation/integration of „helpers“ that assist with regularly performed tasks.


Hex-Rays Home > Plug-In Contest

Hex-Rays

Plug-In Contest 2012: Hall Of Fame

Contests 2012 2011 2010 2009

This year the plugin contest gathered five contestants. But as you know, there can only be ~~one~~, well, two winners!

Based on the plugin's functionality, robustness, usefulness, ease of use and documentation, we declare the following winners:

1. Aaron Portnoy, of Exodus Intelligence, with the *IDA Toolbag* plugin
2. Daniel Plohmann, of the Fraunhofer FKIE, with the *IDAscope* plugin

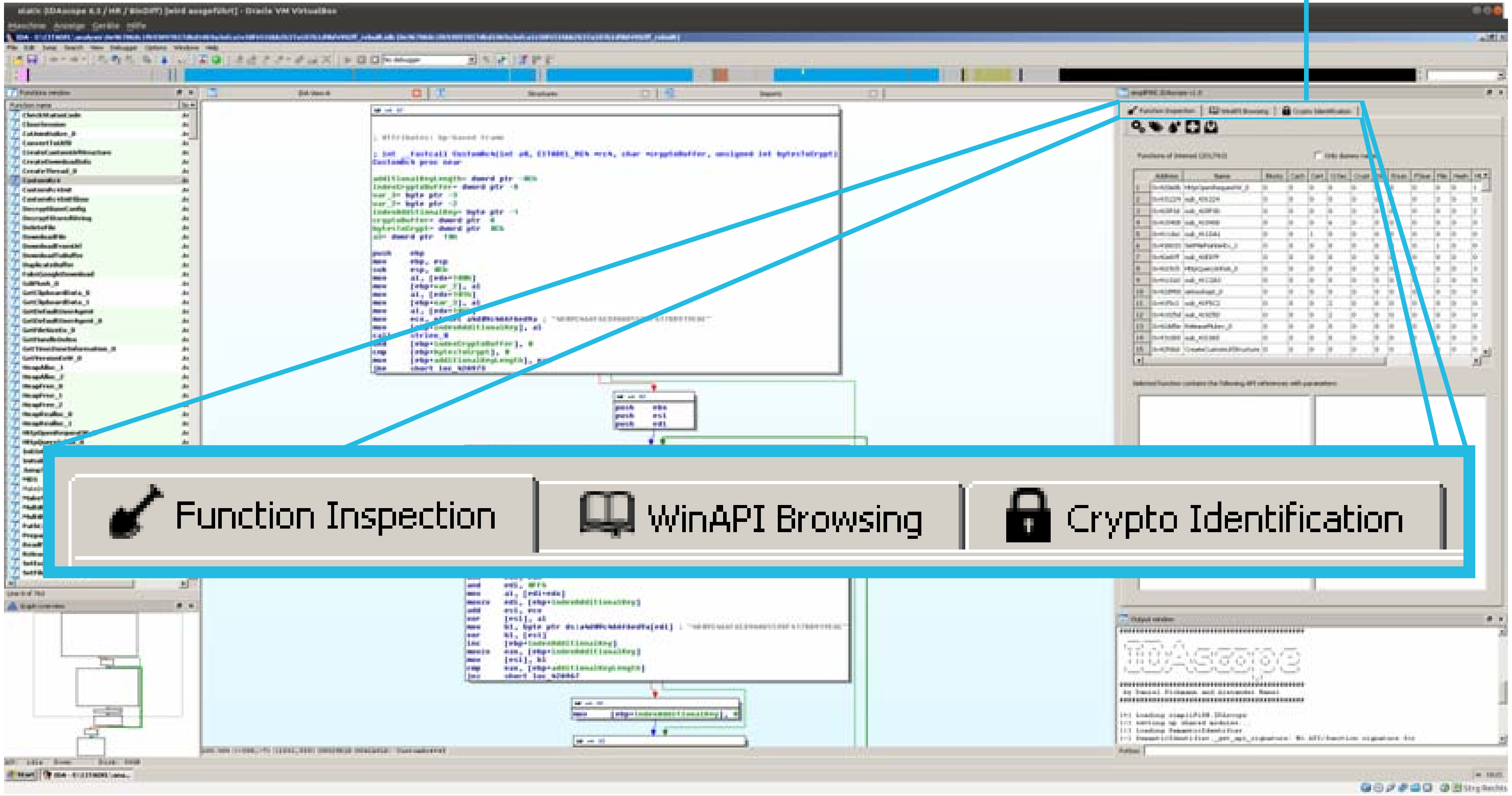
Congratulations to both! We are pleased with the improved plugin quality and complexity.

Below is the list of all submissions in no particular order. All contest entries are interesting and useful:

IDAscope

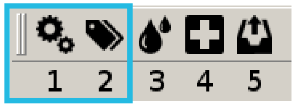
Overview

- Functionality organized in tabs
- Main window can be dragged around like every other IDA view.

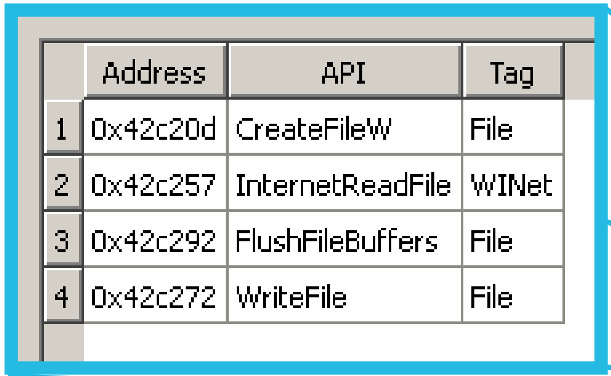


IDAscope: Features

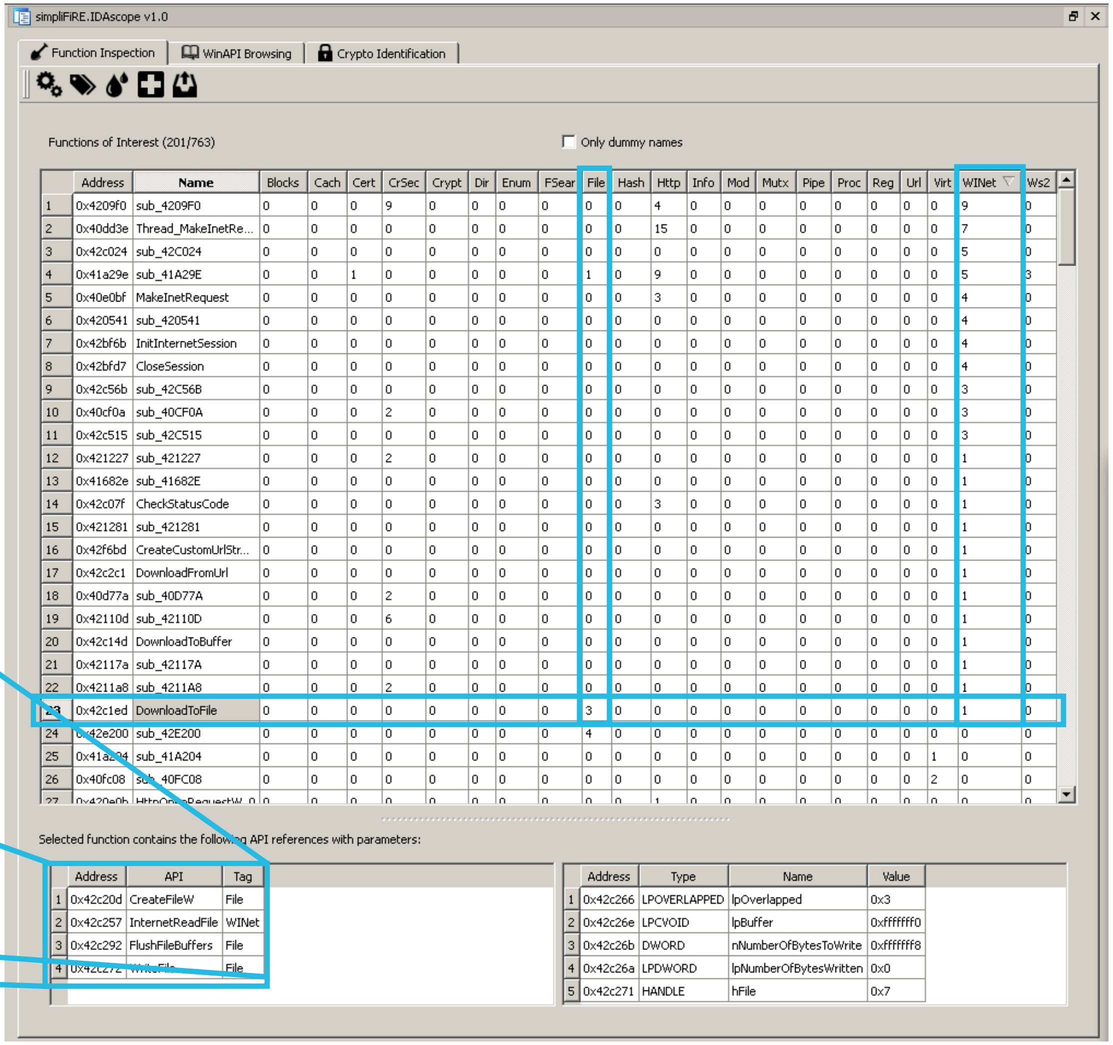
1) Function Inspection



- Tagging of functions
  - Based on API calls
  - APIs can be specified via config
  - Renaming with tags possible
- Example
  - **DownloadToFile** consists of API calls tagged with File and Network



	Address	API	Tag
1	0x42c20d	CreateFileW	File
2	0x42c257	InternetReadFile	WINet
3	0x42c292	FlushFileBuffers	File
4	0x42c272	WriteFile	File



Functions of Interest (201/763)

☐ Only dummy names

	Address	Name	Blocks	Cach	Cert	CrSec	Crypt	Dir	Enum	FSear	File	Hash	Http	Info	Mod	Mutx	Pipe	Proc	Reg	Url	Virt	WINet	Ws2
1	0x4209f0	sub_4209F0	0	0	0	9	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	9	0
2	0x40dd3e	Thread_MakeInetRe...	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	7	0
3	0x42c024	sub_42C024	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0
4	0x41a29e	sub_41A29E	0	0	1	0	0	0	0	0	1	0	9	0	0	0	0	0	0	0	0	5	3
5	0x40e0bf	MakeInetRequest	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	4	0
6	0x420541	sub_420541	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
7	0x42bf6b	InitInternetSession	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
8	0x42bfd7	CloseSession	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
9	0x42c56b	sub_42C56B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0
10	0x40cf0a	sub_40CF0A	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0
11	0x42c515	sub_42C515	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0
12	0x421227	sub_421227	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
13	0x41682e	sub_41682E	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
14	0x42c07f	CheckStatusCode	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	1	0
15	0x421281	sub_421281	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
16	0x42f6bd	CreateCustomUrlStr...	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
17	0x42c2c1	DownloadFromUrl	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
18	0x40d77a	sub_40D77A	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
19	0x42110d	sub_42110D	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
20	0x42c14d	DownloadToBuffer	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
21	0x42117a	sub_42117A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
22	0x4211a8	sub_4211A8	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
23	0x42c1ed	DownloadToFile	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	1	0
24	0x42e200	sub_42E200	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0
25	0x41a204	sub_41A204	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
26	0x40fc08	sub_40FC08	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0
27	0x420e0b	HttpOpenRequestW_0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Selected function contains the following API references with parameters:

	Address	API	Tag
1	0x42c20d	CreateFileW	File
2	0x42c257	InternetReadFile	WINet
3	0x42c292	FlushFileBuffers	File
4	0x42c272	WriteFile	File

	Address	Type	Name	Value
1	0x42c266	LPOVERLAPPED	lpOverlapped	0x3
2	0x42c26e	LPCVOID	lpBuffer	0xfffffff0
3	0x42c26b	DWORD	nNumberOfBytesToWrite	0xfffffff8
4	0x42c26a	LPDWORD	lpNumberOfBytesWritten	0x0
5	0x42c271	HANDLE	hFile	0x7

Fraunhofer FKIE

IDAscope: Features

1) Function Inspection

- Coloring of basic blocks
  - Based on API semantics
  - Colors can be adjusted
- More an experiment :)

Sysinfo/Registry

File Access

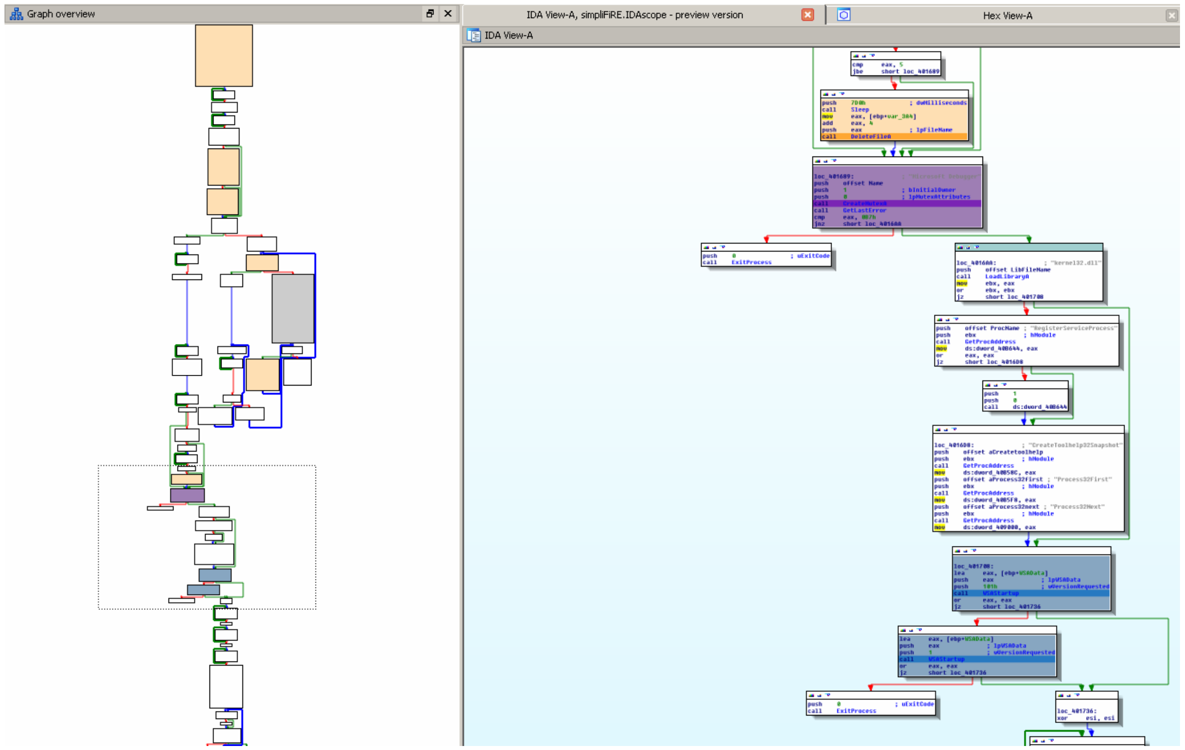
Memory Access

Crypto

Network

Execution

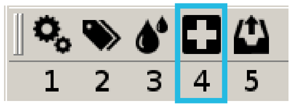
Multi



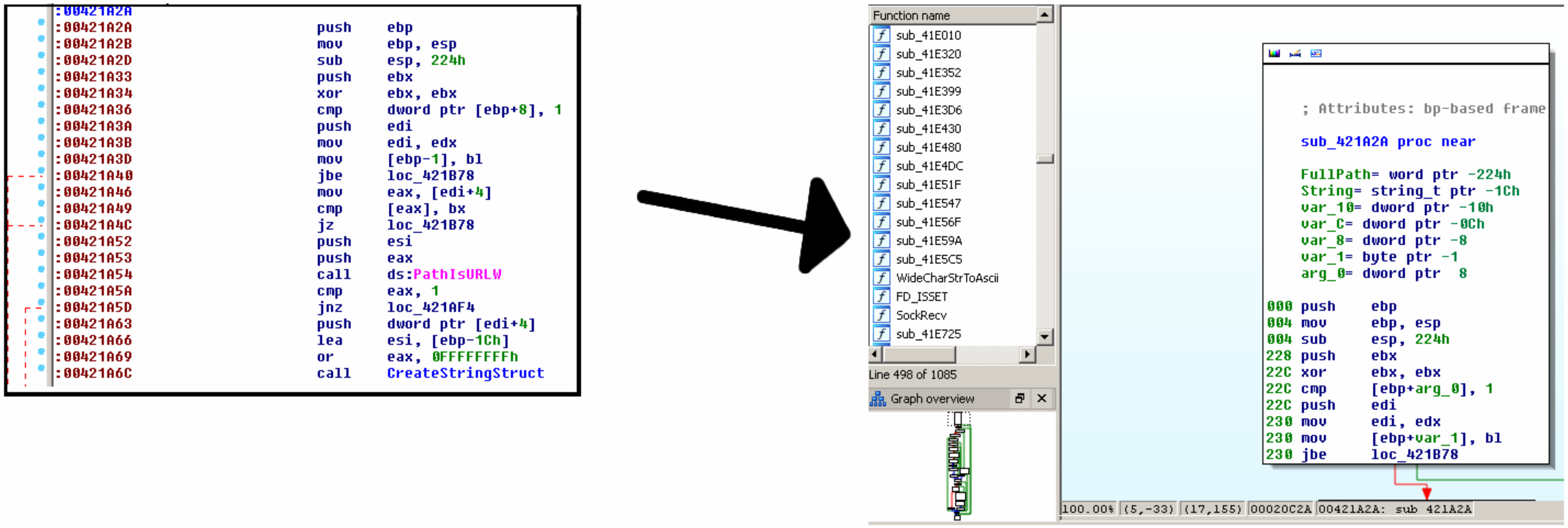
Fraunhofer FKIE

IDAscope: Features

1) Function Inspection



- Code to function conversion
  - Function prologues get handled first
  - Then remaining undefined areas
  - Opens these code sections to further analysis



```
:00421A2A
:00421A2A                push    ebp
:00421A2B                mov     ebp, esp
:00421A2D                sub     esp, 224h
:00421A33                push    ebx
:00421A34                xor     ebx, ebx
:00421A36                cmp     dword ptr [ebp+8], 1
:00421A3A                push    edi
:00421A3B                mov     edi, edx
:00421A3D                mov     [ebp-1], bl
:00421A40                jbe     loc_421B78
:00421A46                mov     eax, [edi+4]
:00421A49                cmp     [eax], bx
:00421A4C                jz      loc_421B78
:00421A52                push    esi
:00421A53                push    eax
:00421A54                call    ds:PathIsURLW
:00421A5A                cmp     eax, 1
:00421A5D                jnz     loc_421AF4
:00421A63                push    dword ptr [edi+4]
:00421A66                lea     esi, [ebp-1Ch]
:00421A69                or      eax, 0FFFFFFFFh
:00421A6C                call    CreateStringStruct
```

Function name
- sub_41E010
- sub_41E320
- sub_41E352
- sub_41E399
- sub_41E3D6
- sub_41E430
- sub_41E480
- sub_41E4DC
- sub_41E51F
- sub_41E547
- sub_41E56F
- sub_41E59A
- sub_41E5C5
- WideCharStrToAscii
- FD_ISSET
- SockRecv
- sub_41E725

Line 498 of 1085

Graph overview

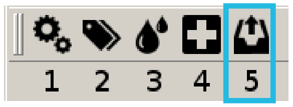
```
; Attributes: bp-based frame

sub_421A2A proc near

FullPath= word ptr -224h
String= string_t ptr -1Ch
var_10= dword ptr -10h
var_C= dword ptr -0Ch
var_8= dword ptr -8
var_1= byte ptr -1
arg_0= dword ptr  8

000 push    ebp
004 mov     ebp, esp
004 sub     esp, 224h
228 push    ebx
22C xor     ebx, ebx
22C cmp     [ebp+arg_0], 1
22C push    edi
230 mov     edi, edx
230 mov     [ebp+var_1], bl
230 jbe     loc_421B78
```

```
100.00% (5,-33) (17,155) 00020C2A 00421A2A: sub_421A2A
```

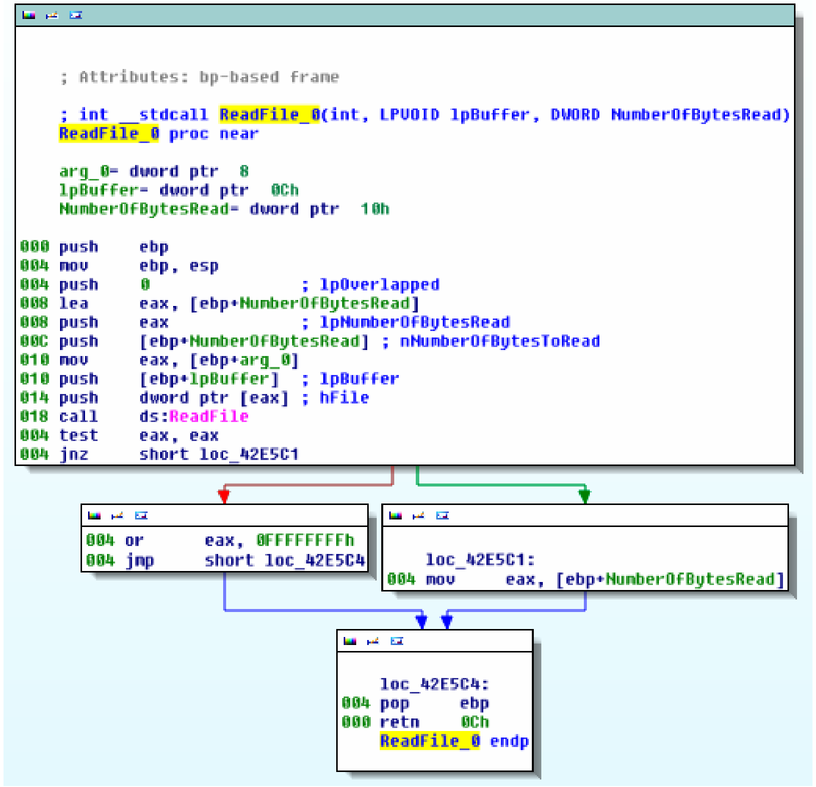
Fraunhofer FKIE

IDAscope: Features

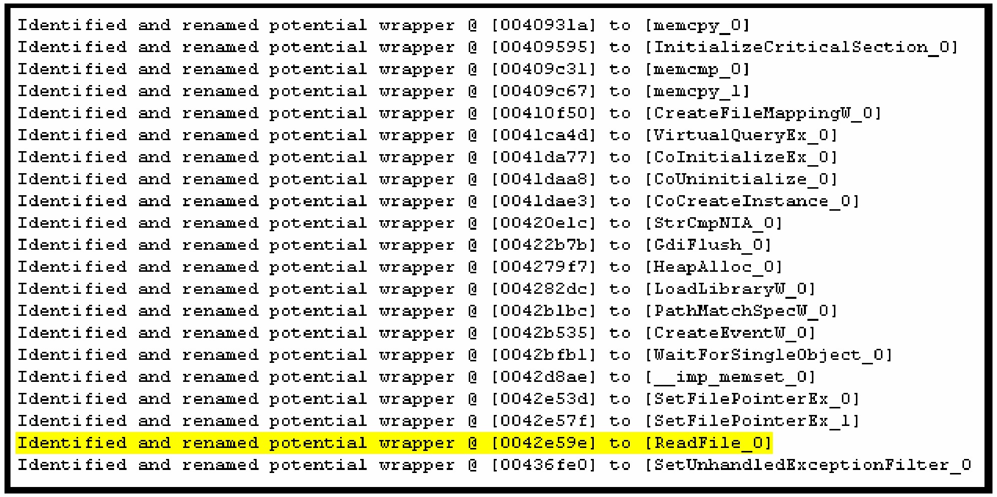
1) Function Inspection



- Automatic renaming of wrapper functions
  - Credits go to Branko Spasojevic (author of Optimice) for providing the code!



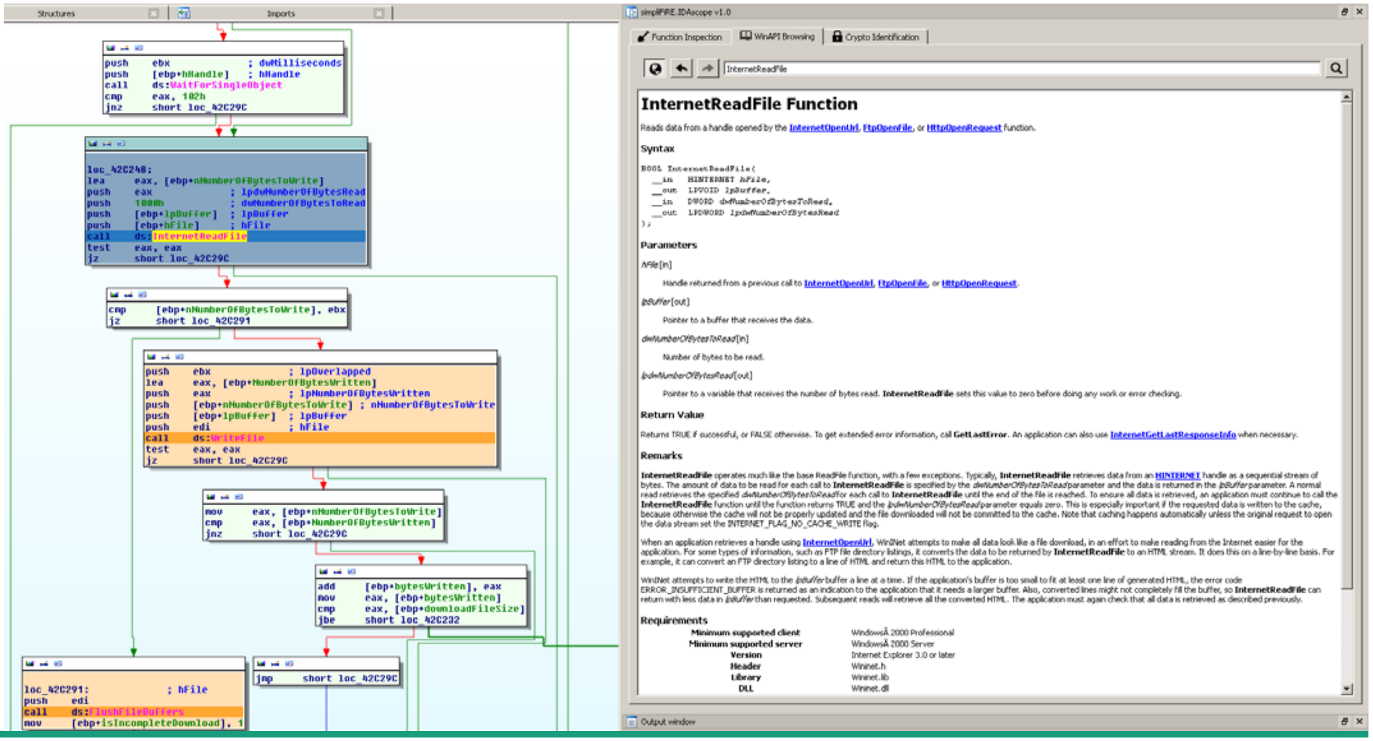
```
; Attributes: bp-based frame

; int __stdcall ReadFile_0(int, LPVOID lpBuffer, DWORD NumberOfBytesRead)
ReadFile_0 proc near

arg_0= dword ptr  8
lpBuffer= dword ptr  0Ch
NumberOfBytesRead= dword ptr  10h

000 push    ebp
004 mov     ebp, esp
004 push    0                  ; lpOverlapped
008 lea     eax, [ebp+NumberOfBytesRead]
008 push    eax                ; lpNumberOfBytesRead
00C push    [ebp+NumberOfBytesRead] ; nNumberOfBytesToRead
010 mov     eax, [ebp+arg_0]
010 push    [ebp+lpBuffer]  ; lpBuffer
014 push    dword ptr [eax] ; hFile
018 call    ds:ReadFile
004 test    eax, eax
004 jnz     short loc_42E5C1

004 or      eax, 0FFFFFFFFh
004 jmp     short loc_42E5C4

loc_42E5C1:
004 mov     eax, [ebp+NumberOfBytesRead]

loc_42E5C4:
004 pop     ebp
000 retn    0Ch
ReadFile_0 endp
```



```
Identified and renamed potential wrapper @ [0040931a] to [memcpy_0]
Identified and renamed potential wrapper @ [00409595] to [InitializeCriticalSection_0]
Identified and renamed potential wrapper @ [00409c31] to [memcmp_0]
Identified and renamed potential wrapper @ [00409c67] to [memcpy_1]
Identified and renamed potential wrapper @ [00410f50] to [CreateFileMappingW_0]
Identified and renamed potential wrapper @ [0041ca4d] to [VirtualQueryEx_0]
Identified and renamed potential wrapper @ [0041da77] to [CoInitializeEx_0]
Identified and renamed potential wrapper @ [0041daa8] to [CoUninitialize_0]
Identified and renamed potential wrapper @ [0041dae3] to [CoCreateInstance_0]
Identified and renamed potential wrapper @ [00420e1c] to [StrCmpNIA_0]
Identified and renamed potential wrapper @ [00422b7b] to [GdiFlush_0]
Identified and renamed potential wrapper @ [004279f7] to [HeapAlloc_0]
Identified and renamed potential wrapper @ [004282dc] to [LoadLibraryW_0]
Identified and renamed potential wrapper @ [0042b1bc] to [PathMatchSpecW_0]
Identified and renamed potential wrapper @ [0042b535] to [CreateEventW_0]
Identified and renamed potential wrapper @ [0042bfb1] to [WaitForSingleObject_0]
Identified and renamed potential wrapper @ [0042d8ae] to [__imp_memset_0]
Identified and renamed potential wrapper @ [0042e53d] to [SetFilePointerEx_0]
Identified and renamed potential wrapper @ [0042e57f] to [SetFilePointerEx_1]
Identified and renamed potential wrapper @ [0042e59e] to [ReadFile_0]
Identified and renamed potential wrapper @ [00436fe0] to [SetUnhandledExceptionFilter_0
```

IDAscope: Features
2) WinAPI Browsing

- Seamless integration of MSDN in IDA Pro
  - accessible via shortcut on highlighted elements
  - Now also with online lookup!
  - But not multi-threaded / no backgrounded lookups yet



```
push    ebx             ; dwMilliseconds
push    [ebp+hHandle]   ; hHandle
call    ds:WaitForSingleObject
cmp     eax, 102h
jnz     short loc_42C29C
```

```
loc_42C248:
lea     eax, [ebp+nNumberOfBytesToWrite]
push    eax             ; lpdwNumberOfBytesRead
push    1000h           ; dwNumberOfBytesToRead
push    [ebp+lpBuffer]  ; lpBuffer
push    [ebp+hFile]     ; hFile
call    ds:InternetReadFile
test    eax, eax
jz      short loc_42C29C
```

```
cmp     [ebp+nNumberOfBytesToWrite], ebx
jz      short loc_42C291
```

```
push    ebx             ; lpOverlapped
lea     eax, [ebp+NumberOfBytesWritten]
push    eax             ; lpNumberOfBytesWritten
push    [ebp+nNumberOfBytesToWrite] ; nNumberOfBytesToWrite
push    [ebp+lpBuffer]  ; lpBuffer
push    edi             ; hFile
call    ds:WriteFile
test    eax, eax
jz      short loc_42C29C
```

```
mov     eax, [ebp+nNumberOfBytesToWrite]
cmp     eax, [ebp+NumberOfBytesWritten]
jnz     short loc_42C29C
```

```
add     [ebp+bytesWritten], eax
mov     eax, [ebp+bytesWritten]
cmp     eax, [ebp+downloadFileSize]
jbe     short loc_42C232
```

```
loc_42C291:             ; hFile
push    edi
call    ds:FlushFileBuffers
mov     [ebp+isIncompleteDownload], 1
```

```
jmp     short loc_42C29C
```

InternetReadFile Function

Reads data from a handle opened by the InternetOpenUrl, FtpOpenFile, or HttpOpenRequest function.

Syntax

```
BOOL InternetReadFile(
  __in   HINTERNET hFile,
  __out  LPVOID lpBuffer,
  __in   DWORD dwNumberOfBytesToRead,
  __out  LPDWORD lpdwNumberOfBytesRead
);
```

Parameters

hFile [in]

Handle returned from a previous call to InternetOpenUrl, FtpOpenFile, or HttpOpenRequest.

lpBuffer [out]

Pointer to a buffer that receives the data.

dwNumberOfBytesToRead [in]

Number of bytes to be read.

lpdwNumberOfBytesRead [out]

Pointer to a variable that receives the number of bytes read. **InternetReadFile** sets this value to zero before doing any work or error checking.

Return Value

Returns TRUE if successful, or FALSE otherwise. To get extended error information, call **GetLastError**. An application can also use InternetGetLastResponseInfo when necessary.

Remarks

InternetReadFile operates much like the base ReadFile function, with a few exceptions. Typically, **InternetReadFile** retrieves data from an HINTERNET handle as a sequential stream of bytes. The amount of data to be read for each call to **InternetReadFile** is specified by the *dwNumberOfBytesToRead* parameter and the data is returned in the *lpBuffer* parameter. A normal read retrieves the specified *dwNumberOfBytesToRead* for each call to **InternetReadFile** until the end of the file is reached. To ensure all data is retrieved, an application must continue to call the **InternetReadFile** function until the function returns TRUE and the *lpdwNumberOfBytesRead* parameter equals zero. This is especially important if the requested data is written to the cache, because otherwise the cache will not be properly updated and the file downloaded will not be committed to the cache. Note that caching happens automatically unless the original request to open the data stream set the INTERNET_FLAG_NO_CACHE_WRITE flag.

When an application retrieves a handle using InternetOpenUrl, WinINet attempts to make all data look like a file download, in an effort to make reading from the Internet easier for the application. For some types of information, such as FTP file directory listings, it converts the data to be returned by **InternetReadFile** to an HTML stream. It does this on a line-by-line basis. For example, it can convert an FTP directory listing to a line of HTML and return this HTML to the application.

WinINet attempts to write the HTML to the *lpBuffer* buffer a line at a time. If the application's buffer is too small to fit at least one line of generated HTML, the error code ERROR_INSUFFICIENT_BUFFER is returned as an indication to the application that it needs a larger buffer. Also, converted lines might not completely fill the buffer, so **InternetReadFile** can return with less data in *lpBuffer* than requested. Subsequent reads will retrieve all the converted HTML. The application must again check that all data is retrieved as described previously.

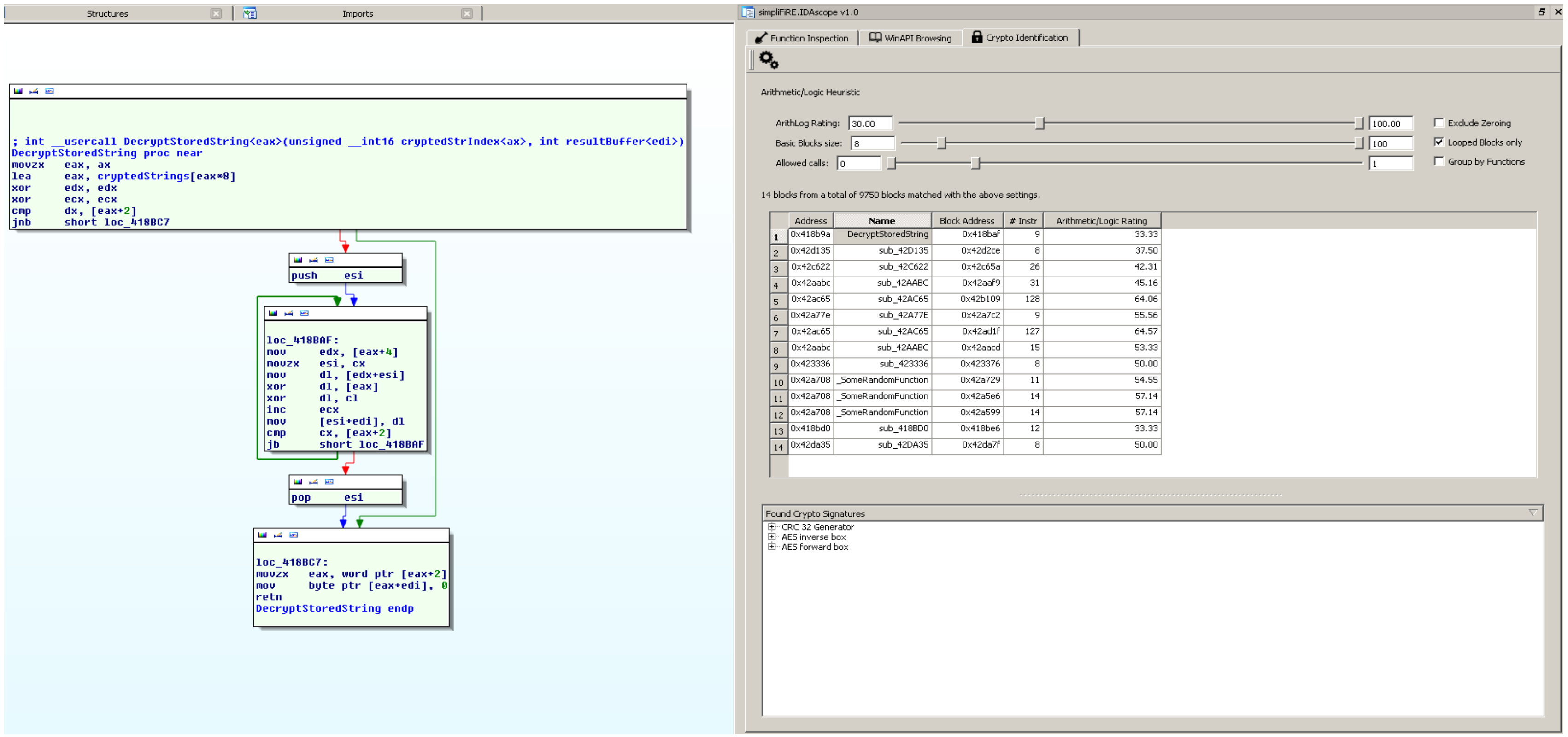
Requirements

Minimum supported client	WindowsÂ 2000 Professional
Minimum supported server	WindowsÂ 2000 Server
Version	Internet Explorer 3.0 or later
Header	Wininet.h
Library	Wininet.lib
DLL	Wininet.dll

Fraunhofer FKIE

IDAscope: Features
3) Crypto Identification

- Identification of cryptographic / compression routines
  - Based on ratio of arithmetic / logic instructions to all instructions in a basic block
  - Approach described in „Dispatcher: Enabling Active Botnet Infiltration using Automatic Protocol Reverse-Engineering“ by Juan Caballero et al.



```
; int __usercall DecryptStoredString<eax>(unsigned __int16 cryptedStrIndex<ax>, int resultBuffer<edi>)
DecryptStoredString proc near
movzx   eax, ax
lea     eax, cryptedStrings[eax*8]
xor     edx, edx
xor     ecx, ecx
cmp     dx, [eax+2]
jnb     short loc_418BC7

push    esi

loc_418BAF:
mov     edx, [eax+4]
movzx   esi, cx
mov     dl, [edx+esi]
xor     dl, [eax]
xor     dl, cl
inc     ecx
mov     [esi+edi], dl
cmp     cx, [eax+2]
jb      short loc_418BAF

pop     esi

loc_418BC7:
movzx   eax, word ptr [eax+2]
mov     byte ptr [eax+edi], 0
retn
DecryptStoredString endp
```

Arithmetic/Logic Heuristic

ArithLog Rating:	30.00	100.00	☐ Exclude Zeroing
Basic Blocks size:	8	100	☑ Looped Blocks only
Allowed calls:	0	1	☐ Group by Functions

14 blocks from a total of 9750 blocks matched with the above settings.

	Address	Name	Block Address	# Instr	Arithmetic/Logic Rating
1	0x418b9a	DecryptStoredString	0x418baf	9	33.33
2	0x42d135	sub_42D135	0x42d2ce	8	37.50
3	0x42c622	sub_42C622	0x42c65a	26	42.31
4	0x42aabc	sub_42AABC	0x42aaf9	31	45.16
5	0x42ac65	sub_42AC65	0x42b109	128	64.06
6	0x42a77e	sub_42A77E	0x42a7c2	9	55.56
7	0x42ac65	sub_42AC65	0x42ad1f	127	64.57
8	0x42aabc	sub_42AABC	0x42aacd	15	53.33
9	0x423336	sub_423336	0x423376	8	50.00
10	0x42a708	_SomeRandomFunction	0x42a729	11	54.55
11	0x42a708	_SomeRandomFunction	0x42a5e6	14	57.14
12	0x42a708	_SomeRandomFunction	0x42a599	14	57.14
13	0x418bd0	sub_418BD0	0x418be6	12	33.33
14	0x42da35	sub_42DA35	0x42da7f	8	50.00

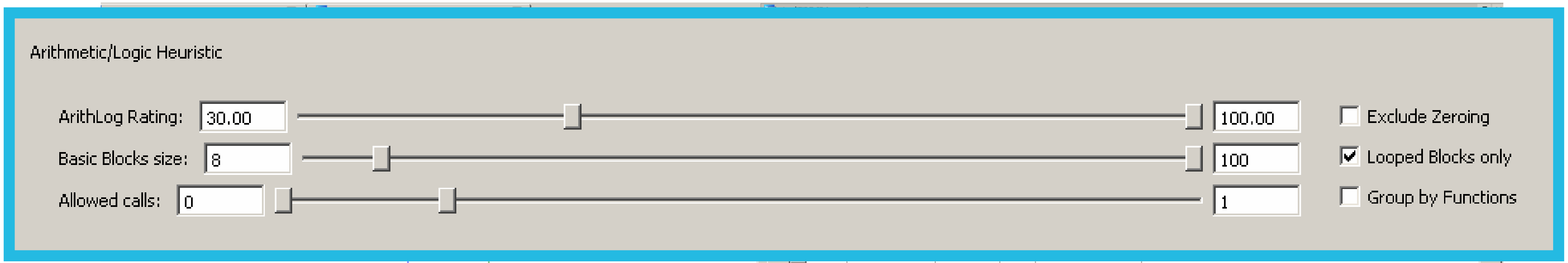
Found Crypto Signatures
- CRC 32 Generator
- AES inverse box
- AES forward box

Fraunhofer FKIE

IDAscope: Features

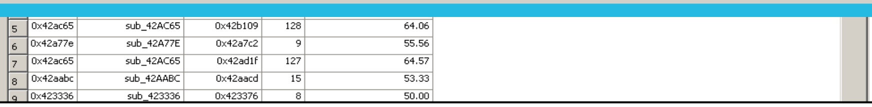
3) Crypto Identification

- Identification of cryptographic / compression routines
  - Based on ratio of arithmetic / logic instructions to all instructions in a basic block
  - Approach described in „Dispatcher: Enabling Active Botnet Infiltration using Automatic Protocol Reverse-Engineering“ by Juan Caballero et al.

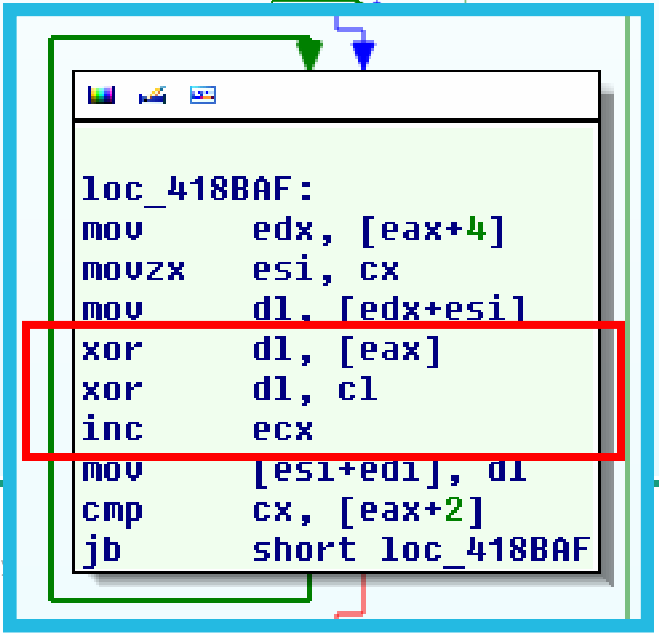


Arithmetic/Logic Heuristic

ArithLog Rating:	30.00	100.00	☐ Exclude Zeroing
Basic Blocks size:	8	100	☑ Looped Blocks only
Allowed calls:	0	1	☐ Group by Functions



5	0x42ac65	sub_42AC65	0x42b109	128	64.06
6	0x42a77e	sub_42A77E	0x42a7c2	9	55.56
7	0x42ac65	sub_42AC65	0x42ad1f	127	64.57
8	0x42aabc	sub_42AABC	0x42aacd	15	53.33
9	0x423336	sub_423336	0x423376	8	50.00



```
loc_418BAF:
mov     edx, [eax+4]
movzx   esi, cx
mov     dl, [edx+esi]
xor     dl, [eax]
xor     dl, cl
inc     ecx
mov     [esi+edi], dl
cmp     cx, [eax+2]
jb      short loc_418BAF
```

Example: Citadel string decryption.

1) 3 AritlogInstructions / 9 Instructions = 33% rating
2) 9 instructions
3) 0 calls
4) Is a looped basic block

=> Matches above parameters

Fraunhofer
FKIE

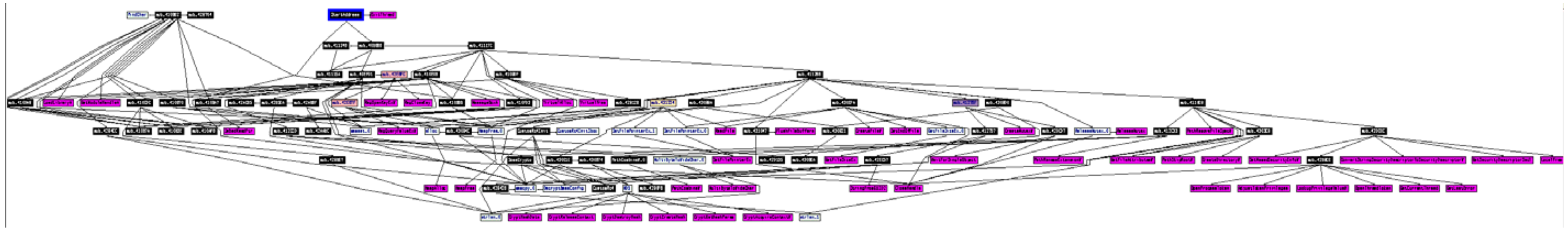
simpliFiRE.IDAscope

Future Plans

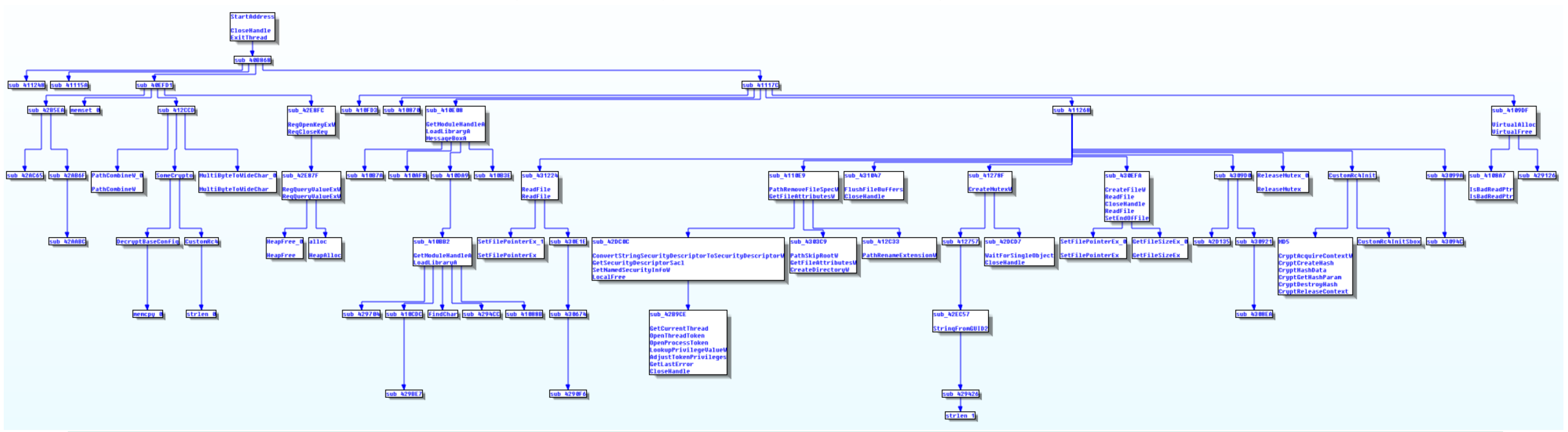
Fraunhofer FKIE

IDAscope: Future Plans
4) Threads / Function Relationship

- Threads and function call chains are a good indicator of functionality
  - A „big picture“ would be very helpful.
  - My opinion: We need something better than this (WinGraph) or step by step navigation via xrefs.



  - Same function scope as IDA graph (IDAPython API has limited graph support), not much better:..


StartAddress CloseHandle ExitThread
sub_40B868
sub_411248
sub_41115A
sub_40EFD1
sub_41117C
sub_42B5EA
memset_0
sub_412CCD
sub_42E8FC RegOpenKeyExW RegCloseKey
sub_410FD3
sub_410870
sub_410E08 GetModuleHandleA LoadLibraryA MessageBoxA
sub_411268
sub_4109DF VirtualAlloc VirtualFree
sub_42AC65
sub_42AB6F
PathCombineW_0 PathCombineW
SomeCrypto
MultiByteToWideChar_0 MultiByteToWideChar
sub_42E87F RegQueryValueExW RegQueryValueExW
sub_410B7A
sub_410AF8
sub_410DA9
sub_410B3E
sub_431224 ReadFile ReadFile
sub_4110E9 PathRemoveFileSpecW GetFileAttributesW
sub_431047 FlushFileBuffers CloseHandle
sub_41278F CreateMutexW
sub_430EFA CreateFileW ReadFile CloseHandle ReadFile SetEndOfFile
sub_4309D0
ReleaseMutex_0 ReleaseMutex
CustomRc4Init
sub_43099A
sub_4108A7 IsBadReadPtr IsBadReadPtr
sub_429126
sub_42AABC
DecryptBaseConfig
CustomRc4
HeapFree_0 HeapFree
alloc HeapAlloc
sub_410BB2 GetModuleHandleA LoadLibraryA
SetFilePointerEx_1 SetFilePointerEx
sub_430E1E
sub_42DC0C ConvertStringSecurityDescriptorToSecurityDescriptorW GetSecurityDescriptorSacl SetNamedSecurityInfoW LocalFree
sub_4303C9 PathSkipRootW GetFileAttributesW CreateDirectoryW
sub_412C33 PathRenameExtensionW
sub_412757
sub_42DCD7 WaitForSingleObject CloseHandle
SetFilePointerEx_0 SetFilePointerEx
GetFileSizeEx_0 GetFileSizeEx
sub_42D135
sub_430921
MD5 CryptAcquireContextW CryptCreateHash CryptHashData CryptGetHashParam CryptDestroyHash CryptReleaseContext
CustomRc4InitSbox
sub_43094C
memcpy_0
strlen_0
sub_429704
sub_410CDC
FindChar
sub_4294CC
sub_41088B
sub_430674
sub_42B9CE GetCurrentThread OpenThreadToken OpenProcessToken LookupPrivilegeValueW AdjustTokenPrivileges GetLastError CloseHandle
sub_42EC57 StringFromGUID2
sub_4308EA
sub_429BE7
sub_4290F6
sub_429426
strlen_1

Fraunhofer FKIE

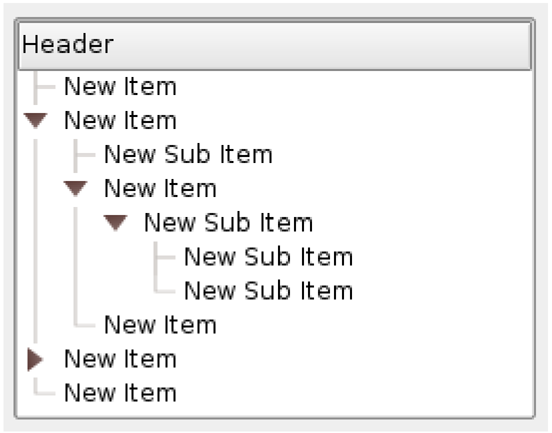
IDAscope: Future Plans
4) Threads / Function Relationship

- Threads and function call chains are a good indicator of functionality
  - Same displayed as tree, generated with Alex‘ script [4]

```
CreateThread Call 0x40bc39
StartAddress (lpStartAddr)
  sub_40B868
    sub_40EFD1
      memset_0
      sub_412CCD
        SomeCrypto
          DecryptBaseConfig
            memcpy_0
          CustomRc4
            strlen_0
        MultiByteToWideChar_0
          * Call MultiByteToWideChar
        PathCombineW_0
          * Call PathCombineW
      sub_42E8FC
        * Call RegOpenKeyExW
        sub_42E87F
          * Call RegQueryValueExW
          alloc
            * Call HeapAlloc
          * Call RegQueryValueExW
          HeapFree_0
            * Call HeapFree
        * Call RegCloseKey
      sub_42B5EA
        sub_42AB6F
          sub_42AABC
        sub_42AC65
    sub_41115A
    sub_41117C
      sub_411268
```

```
sub_41278F
          sub_412757
            sub_42EC57
              sub_429426
                strlen_1
              * Call StringFromGUID2
          * Call CreateMutexW
          sub_42DCD7
            * Call WaitForSingleObject
            * Call CloseHandle
        sub_4110E9
          sub_412C33
            * Call PathRenameExtensionW
          * Call PathRemoveFileSpecW
          sub_4303C9
            * Call PathSkipRootW
            * Call GetFileAttributesW
            * Call CreateDirectoryW
          sub_42DC0C
            sub_42B9CE
              * Call GetCurrentThread
              * Call OpenThreadToken
              * Call OpenProcessToken
              * Call LookupPrivilegeValueW
              * Call AdjustTokenPrivileges
              * Call GetLastError
              * Call CloseHandle
            * Call ConvertStringSecurityDescriptorToSecurityDescriptorW
            * Call GetSecurityDescriptorSacl
            * Call SetNamedSecurityInfoW
            * Call LocalFree
          * Call GetFileAttributesW
```

?



```
Header
├ New Item
▼ New Item
    ├ New Sub Item
    ▼ New Item
        ▼ New Sub Item
            ├ New Sub Item
            └ New Sub Item
        └ New Item
▶ New Item
└ New Item
```

Use a TreeWidget for rendering?

[4] http://hooked-on-mnemonics.blogspot.com/2012/08/ida-thread-analysis-sript.html

IDAscope
Conclusion

- Start using it! :)
  - Repository at
    - http://idascope.pnx.tf
      (points to: https://bitbucket.org/daniel_plohmann/simplifire.idascope)
  - I report about updates
    - in my blog: http://blog.pnx.tf
    - on twitter @push_pnx
  - Alex has a blog, too: http://hooked-on-mnemonics.blogspot.com
- Send feedback or ideas for improvement!
  - idascope@pnx.tf