Where is “that” anti-debug? Introduction Of AntiDebugSeeker

Takahiro Takeda

LAC

Profile

Takahiro Takeda

Malware Analysis Team
Cyber Emergency Center
LAC

2016: Analysis work as a Security Analyst at JSOC.
2017: Analyzing malware and logs, as well as investigating smishing, at the Japan Cyber Crime Control Center (JC3).
2019: Responsible for malware analysis related to incidents at the Cyber Emergency Center.

Speaker Experience:
2020: PACSEC.
2020: AVAR (Association of Anti-Virus Asia Researchers).
2021: HITCON.

Agenda

1. Introduction of the AntiDebugSeeker
2. Demonstration: Using IDA
3. Future Work

Introduction of the AntiDebugSeeker

What is AntiDebugSeeker

This is a program for automatically identify and extract potential anti-debugging techniques used by malware and displaying them in IDA.

The main functionalities of this plugin are as follows:

1. Extraction of APIs that are potentially being used for anti-debugging by the malware.

2. In addition to APIs, extraction of anti-debugging techniques based on key phrases that serve as triggers, as some anti-debugging methods cannot be comprehensively identified by API calls alone.

For packed samples, running this plugin after unpacking and fixing the Import Address Table is more effective.

About anti_debug.config

The anti_debug.config file contains rules for detecting anti-debugging features. It is divided into two sections: Anti_Debug_API and Anti_Debug_Technique.

Anti_Debug_API

```
###Anti_Debug_API###
[Category Name]
API1
API2
API3
  ⋮
```

```
###Anti_Debug_API###
[Debugger check]
CheckRemoteDebuggerPresent
DebugActiveProcess
DebugBreak
DbgSetDebugFilterState
DbgUiDebugActiveProcess
IsDebuggerPresent
```

Anti_Debug_Technique

```
###Anti_Debug_Technique###
default_search_range=80

[Rule1]
ABC ┐ 80bytes
DEF ┘
GHI ┐ 80bytes
    ┘
```

```
###Anti_Debug_Technique###
default_search_range=80

[NtGlobalFlag_check]
fs:30h
68h
70h
```

About Anti_Debug_API in anti_debug.config

In the Anti_Debug_API section, you can freely create categories and add any number of APIs you want to detect. (exact match)

```
###Anti_Debug_API###
[Category Name_1]
API1
API2
API3
  ⋮
[Category Name_2]
API1
API2
API3
  ⋮
```

About Anti_Debug_Technique in anti_debug.config

In this section, you can set up to three keywords (partial match) under a single rule name.

```
###Anti_Debug_Technique###
Default_search_range=80

[Rule1]
ABC
EFG
HIJ
```

80bytes

80bytes

The basic flow of the search is as follows:

First, the search begins with the first keyword. If it is found, the second keyword is then searched for within a specified number of bytes (default is 80 bytes). This same process is applied when searching for the third keyword.

Search Target:

- Disassembly (Opcode, Operand)
- Comments
- API based on Import Table

About Anti_Debug_Technique in anti_debug.config

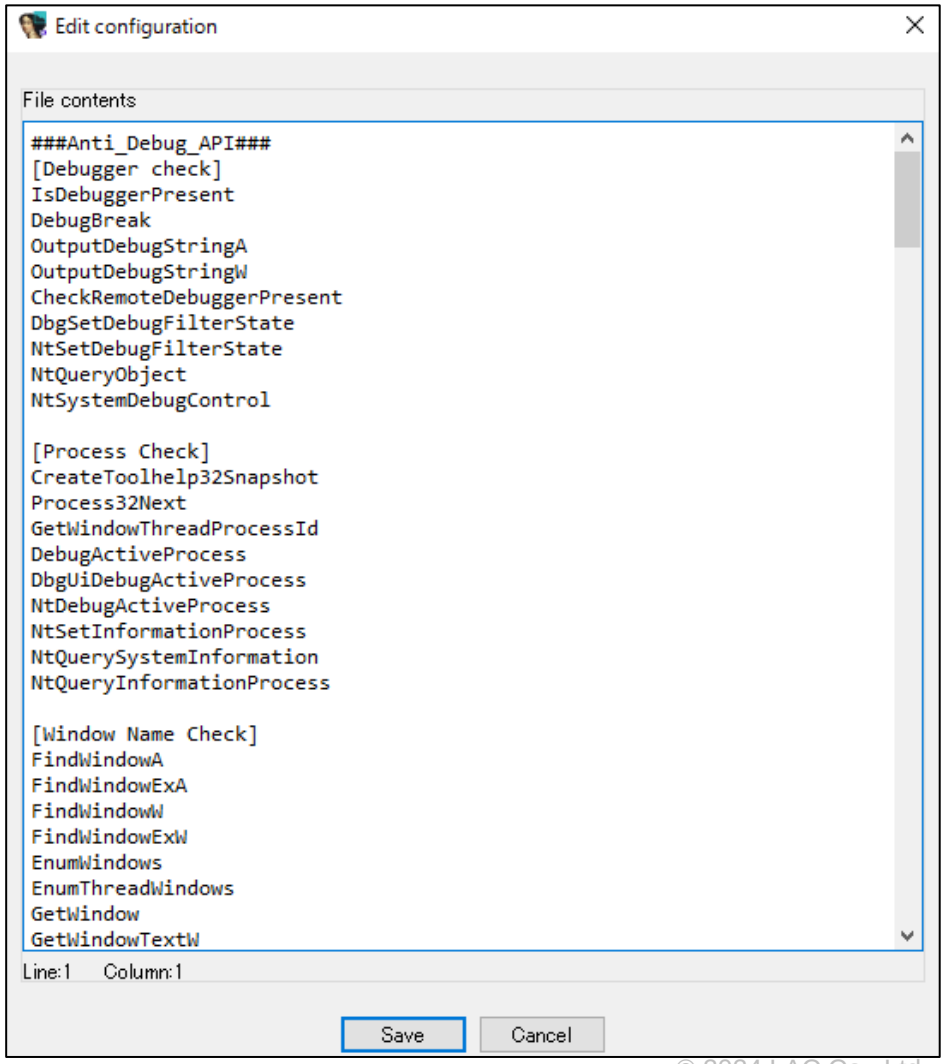
If you want to set a custom search range instead of using the default value, you can specify 'search_range=value' at the end of the keyword you've set. This allows you to change the search range for each rule you've configured.

```
###Anti_Debug_Technique###
default_search_range=80

[Rule1]
ABC
EFG
HIJ
search_range=50

[Rule2]
KLM
NOP
search_range=200
```

anti_debug.config Editing Feature

Functionality for checking and editing the contents of anti_debug.config.
Ctrl + Shift + E (File Editing Feature)

After making changes,
click the 'Save' button to save the modifications.



Support Functions

After running the plugin, detected APIs and keywords are highlighted in different colors.

Detection Category	Color
Anti_Debug_API	Green
Anti_Debug_Technique	Orange

Additionally, if an API specified in Anti_Debug_API is detected, the category name is added as a comment. Likewise, if a rule name is detected in Anti_Debug_Technique, a description of that rule is added as a comment to the first detected keyword.

```
call    sub_401D30
mov     ds:WriteProcessMemory, eax ; Write Data OnTheMemory
mov     edx, 9D00A761h
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:ReadProcessMemory, eax ; MemoryRead,ProcessInspection
mov     edx, 9ABFB8A6h
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:VirtualAllocEx, eax ; Memory Manipulation
mov     edx, 6B416786h
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:GetCurrentProcessId, eax
mov     edx, 774393E8h
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:GetModuleFileNameA, eax
mov     edx, 2EE4F10Dh
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:CopyFileA, eax
mov     edx, 19F78C90h
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:Process32First, eax ; Process Check
mov     edx, 0D89AD05h
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:GetCurrentProcess, eax
mov     edx, 0C930EA1Eh
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:Process32Next, eax ; Process Check
mov     edx, 5BC1D14Fh
mov     eax, [ebp+var_8]
call    sub_401D30
mov     ds:CreateToolhelp32Snapshot, eax ; Process Check
mov     edx, 77CD9567h
```

```
push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF8h
mov     eax, large fs:30h ; NtGlobalFlag_check - The code is checking the NtGlobalFlag value at offset 0x68 from the Process Environment Block.
                          ; The value 70 is the sum of FLG_HEAP_ENABLE_TAIL_CHECK (0x10), FLG_HEAP_ENABLE_FREE_CHECK (0x20), and FLG_HEAP_VALIDATE_PARAMETERS (0x40).
sub     esp, 480h
test    byte ptr [eax+68h], 70h
push    esi
push    edi
jz      short loc_4BFFB2
```

```
mov     [ebp+Context.ContextFlags], 10010h ; Hardware_Breakpoints_Check - Check the debug registers DR0, DR1, DR2, and DR3 (CONTEXT_DEBUG_REGISTERS 0x10010)
                                           ; to determine if a hardware breakpoint has been set.
lea     eax, [ebp+Context]
push    eax             ; lpContext
call    ds:GetCurrentThread
push    eax             ; hThread
call    ds:GetThreadContext ; Thread Manipulation
mov     ecx, [ebp+var_4]
xor     eax, eax
xor     ecx, ebp        ; StackCookie
call    @__security_check_cookie@4 ; __security_check_cookie(x)
mov     esp, ebp
pop     ebp
retn
_main endp
```

```
push    ebp
mov     ebp, esp
push    ecx
mov     eax, large fs:30h ; BeingDebugged_check - The BeingDebugged field in the Process Environment Block (PEB) indicates whether the current process is being debugged or not.
movzx   cax, byte ptr [cax+2]
test    eax, eax
setnz   byte ptr [ebp+var_4]
cmp     [ebp+var_4], 0
jz      short loc_40102E
```

List of detectable anti-debugging techniques (ver1.0)

The following Anti Debug Techniques can be detected using AntiDebugSeeker.

HeapTailMarker
KernelDebuggerMarker
DbgBreakPoint_RET
DbgUiRemoteBreakin_Debugger_Terminate
PMCCheck_RDPMC
TimingCheck_RDTSC
SkipPrefixes_INT1
INT2D_interrupt_check
INT3_interrupt_check
EXCEPTION_BREAKPOINT
ICE_interrupt_check
DBG_PRINTEXCEPTION_C
TrapFlag_SingleStepException
BeingDebugged_check
NtGlobalFlag_check
NtGlobalFlag_check_2
HeapFlags
HeapForceFlags
Combination_of_HEAP_Flags
Combination_of_HEAP_Flags_2
ReadHeapFlags
ReadHeapFlags_2
DebugPrivileges_Check
Opened_Exclusively_Check
EXCEPTION_INVALID_HANDLE_1
EXCEPTION_INVALID_HANDLE_2
Memory_EXECUTE_READWRITE_1
Memory_EXECUTE_READWRITE_2
Memory_Region_Tracking
Check_BreakPoint_Memory_1
Check_BreakPoint_Memory_2
Software_Breakpoints_Check
Hardware_Breakpoints_Check
Enumerate_Running_Processes
ThreadHideFromDebugger
NtQueryInformationProcess_PDPort
NtQueryInformationProcess_PDFlags
NtQueryInformationProcess_PDObjectHandle
NtQuerySystemInformation_KD_Check

Updated functions

Updated Feature: Detected Function List

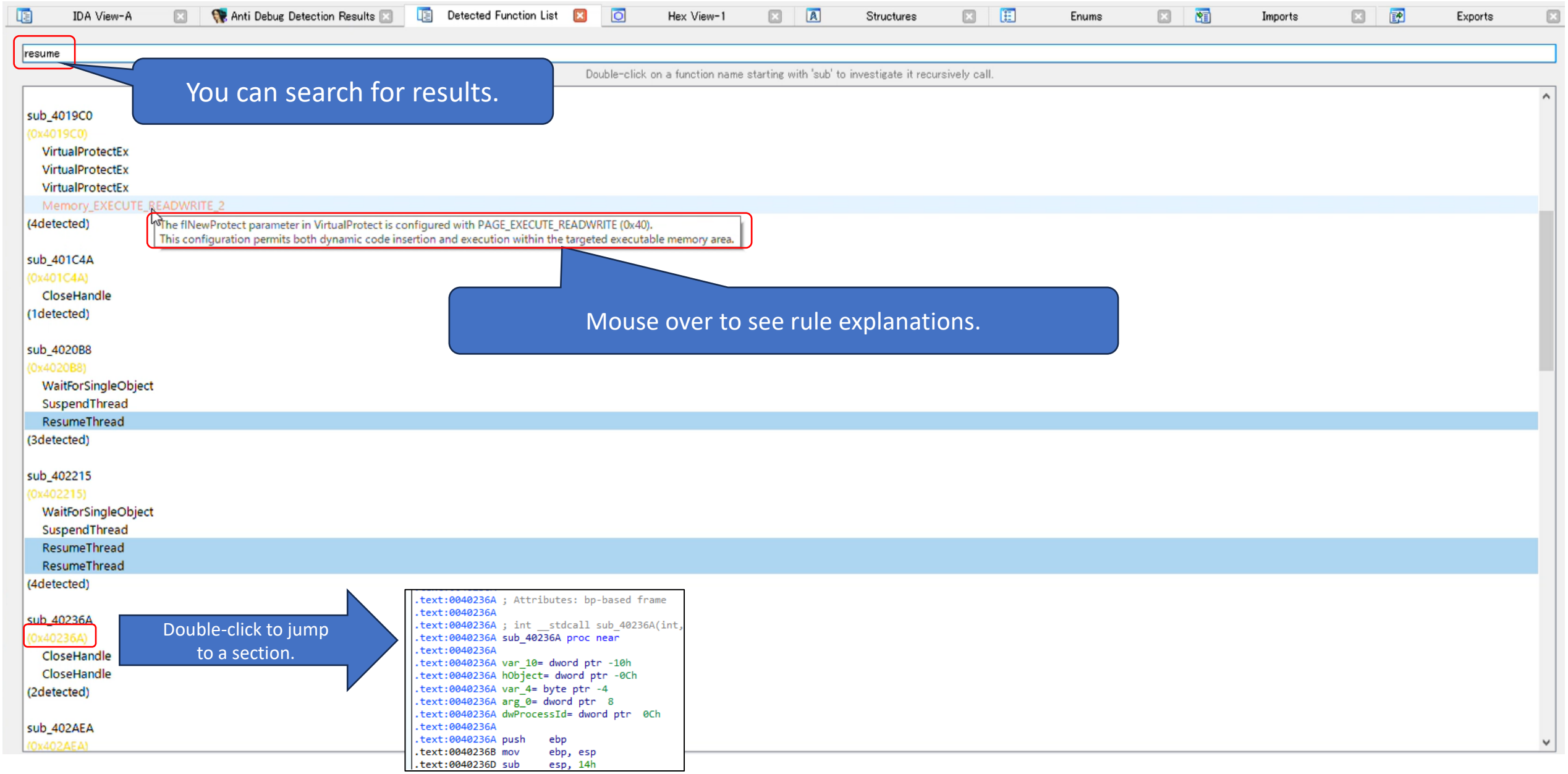
Detected Function List

After launching the plugin with Ctrl+Shift+D, in addition to the Anti Debug Detection Results, we have added a feature to display the Detected Function List.

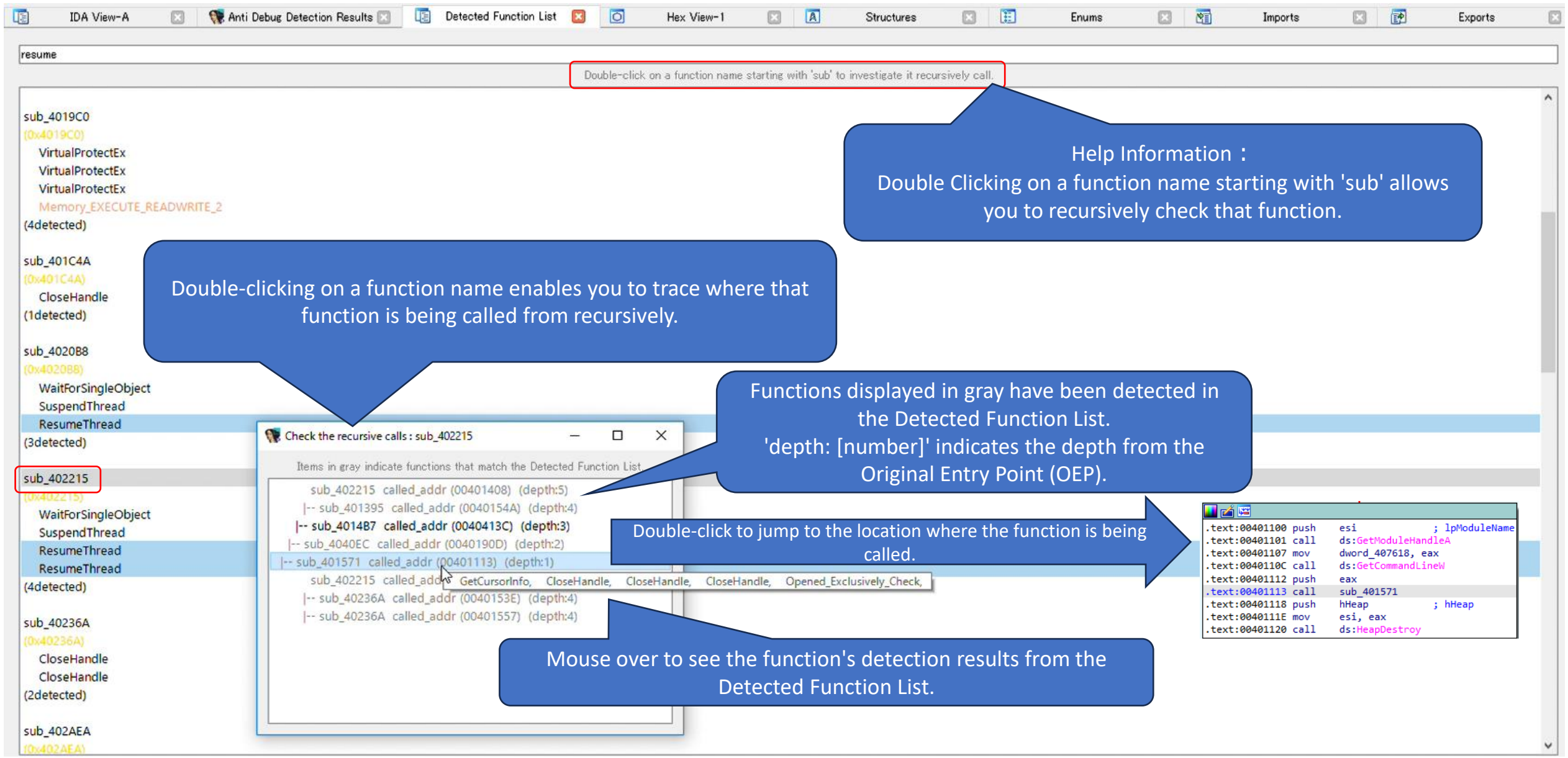
By adding this feature,

With the Anti Debug Detection Results, it becomes easier to grasp both the detection outcomes and the overall flow of the code, while the Detected Function List allows for a more manageable debugging process by providing information organized by each function. This enables malware analysis from two distinct perspectives.

Updated Feature: Detected Function List – Basic Functions



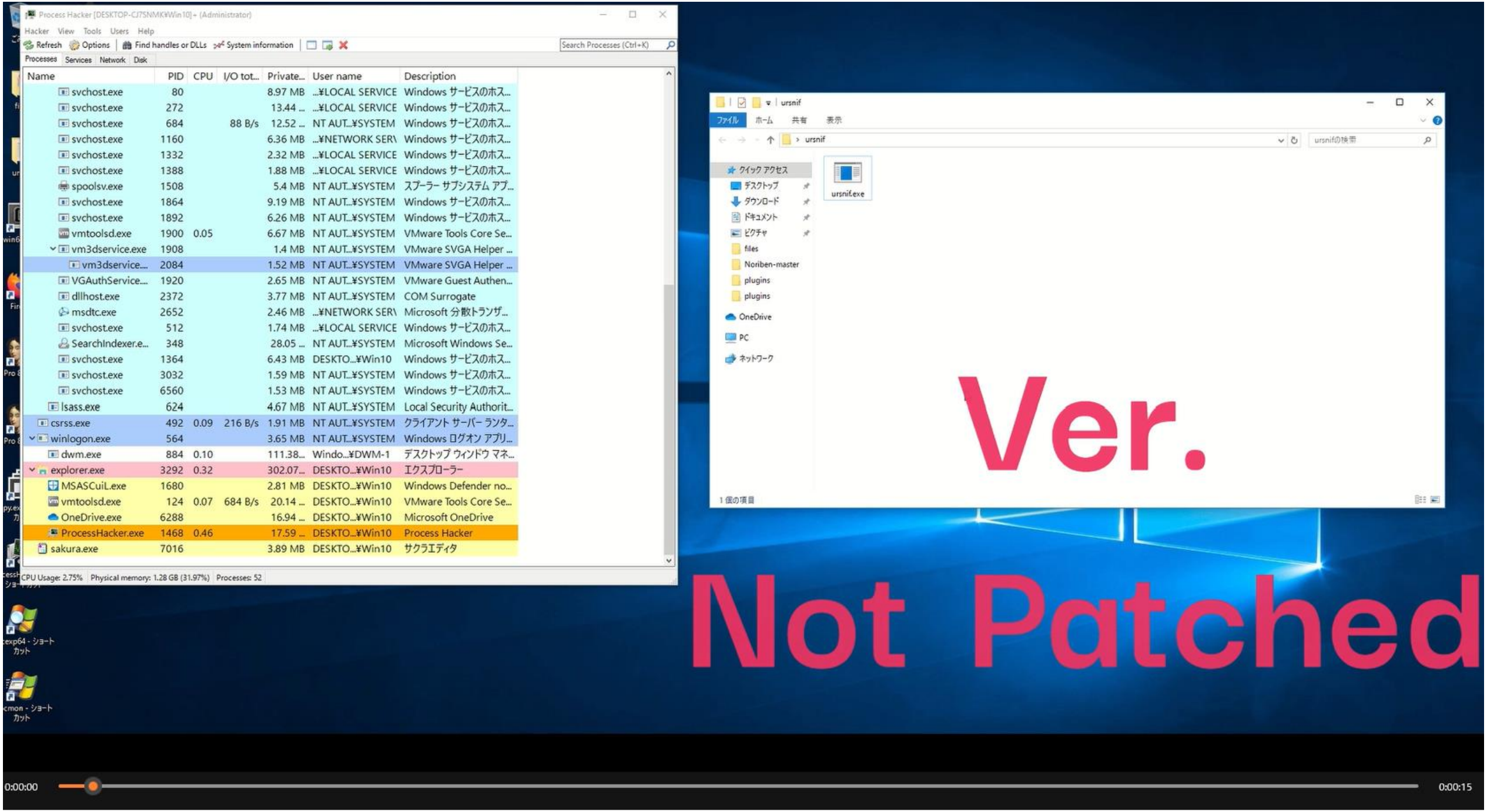
Updated Feature: Detected Function List – Recursive Checking

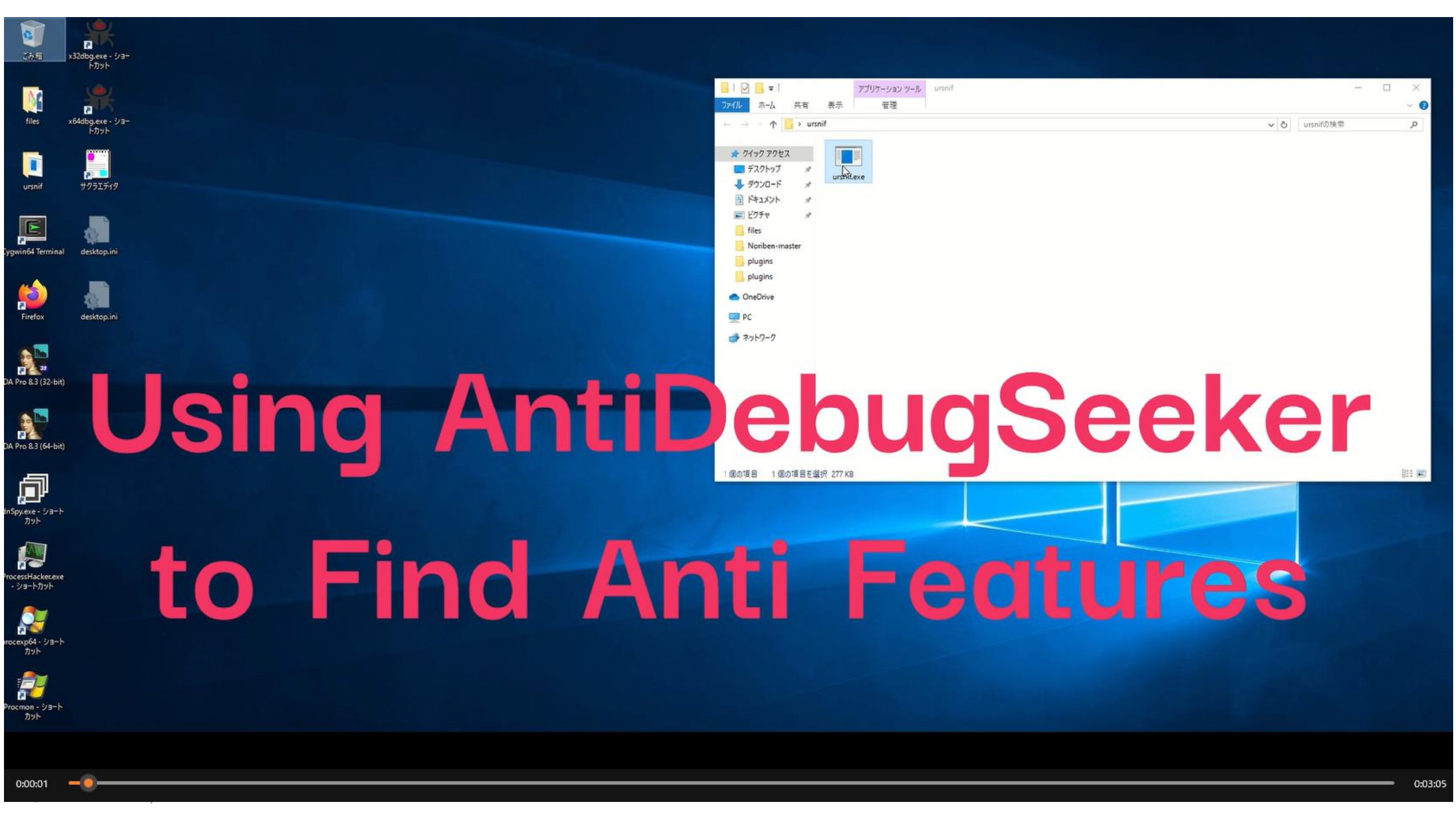


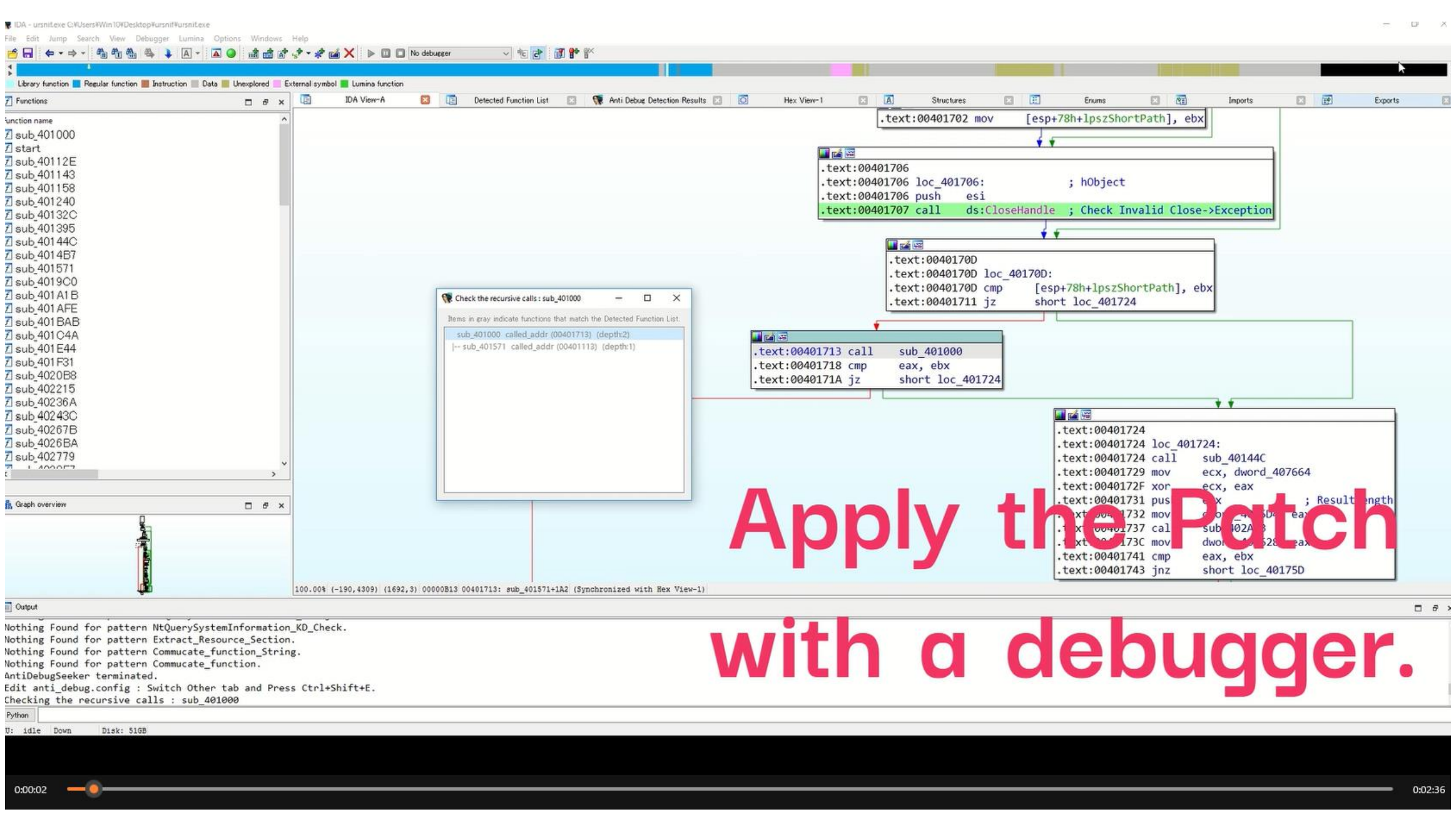
Demo：AntiDebugSeeker

DEMO

- Malware：Ursnif
  - MD5：4da11c829f8fea1b690f317837af8387 (Packed)
  - MD5：952d604345e051fce76729ccb63bde82 (Unpacked)


Process Hacker [DESKTOP-CJ7SNMK¥Win10] + (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
Search Processes (Ctrl+K)
Processes Services Network Disk
Name PID CPU I/O tot... Private... User name Description
svchost.exe 80 8.97 MB ...¥LOCAL SERVICE Windows サービスのホス...
svchost.exe 272 13.44 ... ...¥LOCAL SERVICE Windows サービスのホス...
svchost.exe 684 88 B/s 12.52 ... NT AUT...¥SYSTEM Windows サービスのホス...
svchost.exe 1160 6.36 MB ...¥NETWORK SER\ Windows サービスのホス...
svchost.exe 1332 2.32 MB ...¥LOCAL SERVICE Windows サービスのホス...
svchost.exe 1388 1.88 MB ...¥LOCAL SERVICE Windows サービスのホス...
spoolsv.exe 1508 5.4 MB NT AUT...¥SYSTEM スプーラー サブシステム アプ...
svchost.exe 1864 9.19 MB NT AUT...¥SYSTEM Windows サービスのホス...
svchost.exe 1892 6.26 MB NT AUT...¥SYSTEM Windows サービスのホス...
vmtoolsd.exe 1900 0.05 6.67 MB NT AUT...¥SYSTEM VMware Tools Core Se...
vm3dservice.exe 1908 1.4 MB NT AUT...¥SYSTEM VMware SVGA Helper ...
vm3dservice.... 2084 1.52 MB NT AUT...¥SYSTEM VMware SVGA Helper ...
VGAuthService.... 1920 2.65 MB NT AUT...¥SYSTEM VMware Guest Authen...
dllhost.exe 2372 3.77 MB NT AUT...¥SYSTEM COM Surrogate
msdtc.exe 2652 2.46 MB ...¥NETWORK SER\ Microsoft 分散トランザ...
svchost.exe 512 1.74 MB ...¥LOCAL SERVICE Windows サービスのホス...
SearchIndexer.e... 348 28.05 ... NT AUT...¥SYSTEM Microsoft Windows Se...
svchost.exe 1364 6.43 MB DESKTO...¥Win10 Windows サービスのホス...
svchost.exe 3032 1.59 MB NT AUT...¥SYSTEM Windows サービスのホス...
svchost.exe 6560 1.53 MB NT AUT...¥SYSTEM Windows サービスのホス...
lsass.exe 624 4.67 MB NT AUT...¥SYSTEM Local Security Authorit...
csrss.exe 492 0.09 216 B/s 1.91 MB NT AUT...¥SYSTEM クライアント サーバー ランタ...
winlogon.exe 564 3.65 MB NT AUT...¥SYSTEM Windows ログオン アプリ...
dwm.exe 884 0.10 111.38... Windo...¥DWM-1 デスクトップ ウィンドウ マネ...
explorer.exe 3292 0.32 302.07... DESKTO...¥Win10 エクスプローラー
MSASCuiL.exe 1680 2.81 MB DESKTO...¥Win10 Windows Defender no...
vmtoolsd.exe 124 0.07 684 B/s 20.14 ... DESKTO...¥Win10 VMware Tools Core Se...
OneDrive.exe 6288 16.94 ... DESKTO...¥Win10 Microsoft OneDrive
ProcessHacker.exe 1468 0.46 17.59 ... DESKTO...¥Win10 Process Hacker
sakura.exe 7016 3.89 MB DESKTO...¥Win10 サクラエディタ
CPU Usage: 2.75% Physical memory: 1.28 GB (31.97%) Processes: 52
ursnif
ファイル ホーム 共有 表示
ursnif
ursnifの検索
クイック アクセス
デスクトップ
ダウンロード
ドキュメント
ピクチャ
files
Noriben-master
plugins
plugins
OneDrive
PC
ネットワーク
ursnif.exe
1 個の項目
Ver.
Not Patched
0:00:00
0:00:15

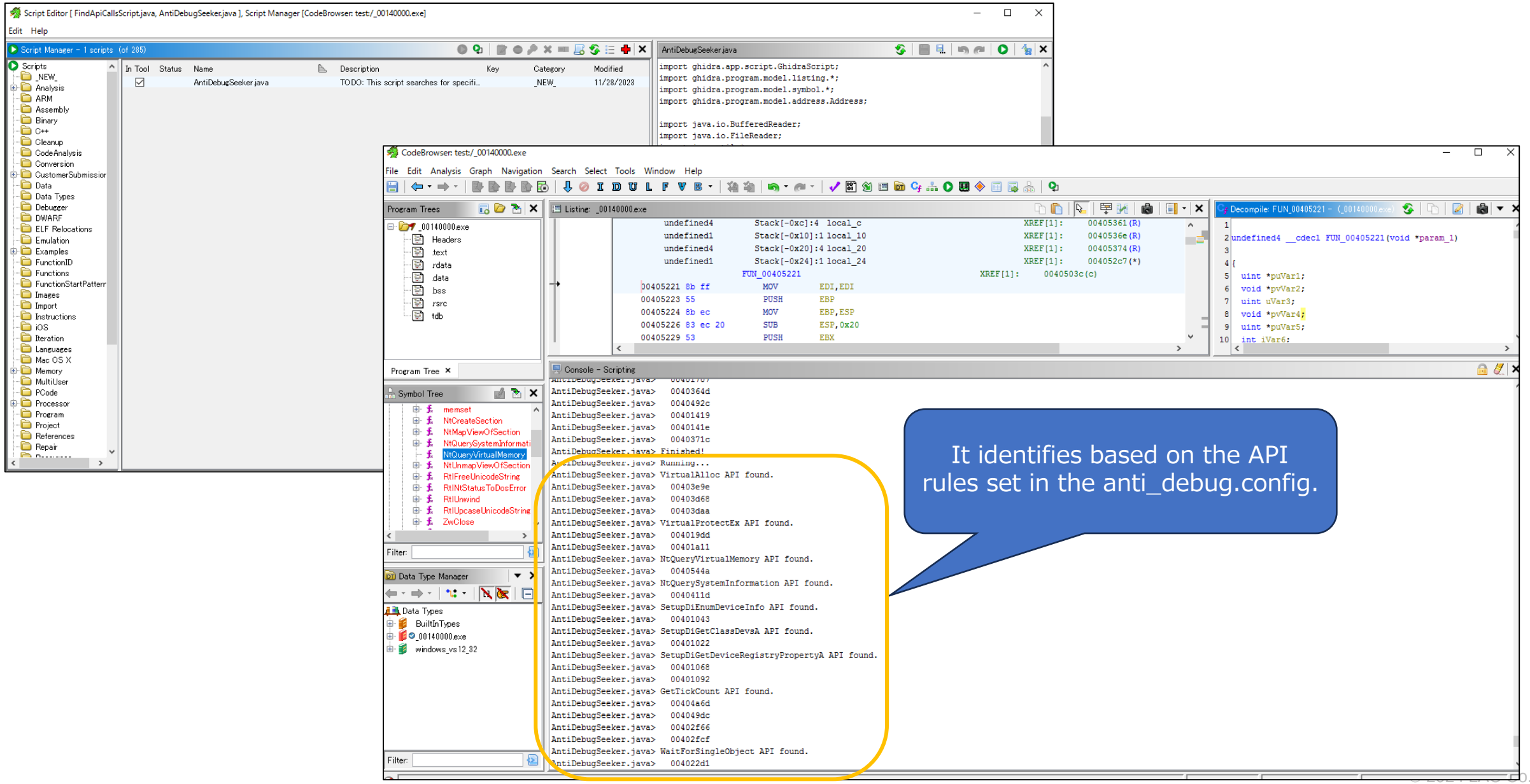

ごみ箱
x32dbg.exe - ショートカット
files
x64dbg.exe - ショートカット
ursnif
サクラエディタ
Cygwin64 Terminal
desktop.ini
Firefox
desktop.ini
DA Pro 8.3 (32-bit)
DA Pro 8.3 (64-bit)
InSpy.exe - ショートカット
ProcessHacker.exe - ショートカット
procexp64 - ショートカット
Procmon - ショートカット
アプリケーション ツール
ursnif
ファイル
ホーム
共有
表示
管理
ursnif
ursnifの検索
クイック アクセス
デスクトップ
ダウンロード
ドキュメント
ピクチャ
files
Noriben-master
plugins
plugins
OneDrive
PC
ネットワーク
ursnif.exe
1 個の項目
1 個の項目を選択 277 KB
Using AntiDebugSeeker
to Find Anti Features
0:00:01
0:03:05


Apply the Patch
with a debugger.
Check the recursive calls : sub_401000
Items in gray indicate functions that match the Detected Function List.
sub_401000 called_addr (00401713) (depth:2)
sub_401571 called_addr (00401113) (depth:1)
.text:00401702 mov [esp+78h+lpszShortPath], ebx
.text:00401706 loc_401706: ; hObject
.text:00401706 push esi
.text:00401707 call ds:CloseHandle ; Check Invalid Close->Exception
.text:0040170D loc_40170D:
.text:0040170D cmp [esp+78h+lpszShortPath], ebx
.text:00401711 jz short loc_401724
.text:00401713 call sub_401000
.text:00401718 cmp eax, ebx
.text:0040171A jz short loc_401724
.text:00401724 loc_401724:
.text:00401724 call sub_40144C
.text:00401729 mov ecx, dword_407664
.text:0040172F xor ecx, eax
.text:00401741 cmp eax, ebx
.text:00401743 jnz short loc_40175D
Nothing Found for pattern NtQuerySystemInformation_KD_Check.
Nothing Found for pattern Extract_Resource_Section.
Nothing Found for pattern Commucate_function_String.
Nothing Found for pattern Commucate_function.
AntiDebugSeeker terminated.
Edit anti_debug.config : Switch Other tab and Press Ctrl+Shift+E.
Checking the recursive calls : sub_401000
0:00:02
0:02:36

Future Work

Plugin being developed for Ghidra version

Ghidra Script



Thank you!

Updated Version 1.1 Released

https://github.com/LAC-Japan/IDA_Plugin_AntiDebugSeeker

README | BSD-3-Clause license

IDA_Plugin_AntiDebugSeeker

Concept

This tool was created to assist those who are new to malware analysis or are not yet familiar with anti-debugging techniques. Through this tool, users can automatically extract potential anti-debugging methods used by malware, making it easier for analysts to take appropriate action.

Introduction

The main functionalities of this plugin are as follows:

- Extraction of Windows API that are potentially being used for anti-debugging by the malware
  (All subsequent API represent the Windows API)
- In addition to API, extraction of anti-debugging techniques based on key phrases that serve as triggers, as some anti-debugging methods cannot be comprehensively identified by API calls alone.

LAC