Gaming Security: Themida packer & unlicense unpacker

陳兆閔 洪宏捷 鄭子瑜

Agenda

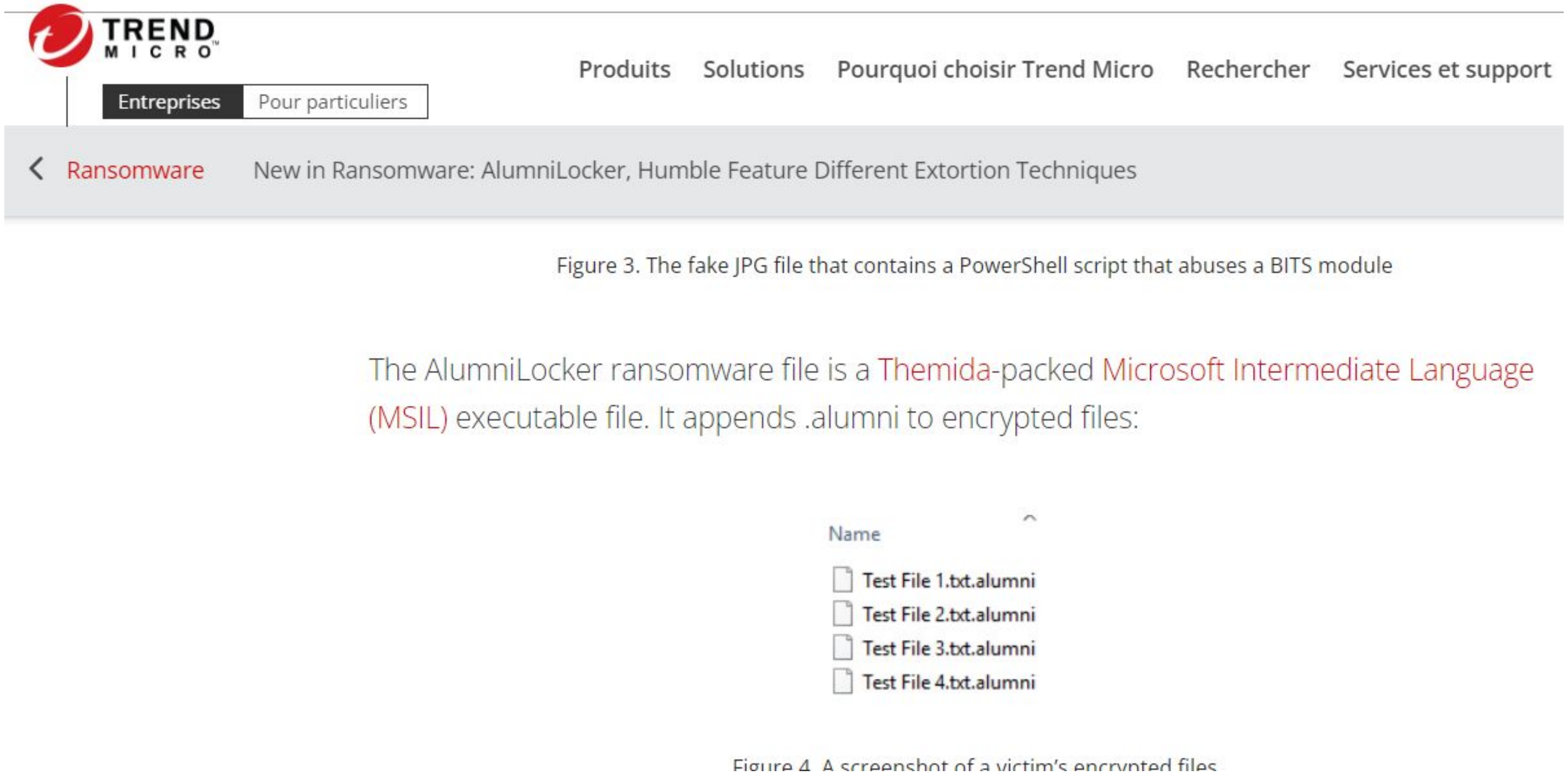
1. intro to gaming packers
2. a peek into themida
3. a peek into unpacker

Intro to gaming packers

Why this topic ?

- Game developer often use packers on their production games
  - Slim production size
  - Obfuscate implementation
- Malware also used packers

AlumniLocker



Produits Solutions Pourquoi choisir Trend Micro Rechercher Services et support

Entreprises Pour particuliers

Ransomware New in Ransomware: AlumniLocker, Humble Feature Different Extortion Techniques

Figure 3. The fake JPG file that contains a PowerShell script that abuses a BITS module

The AlumniLocker ransomware file is a Themida-packed Microsoft Intermediate Language (MSIL) executable file. It appends .alumni to encrypted files:

```
Name
Test File 1.txt.alumni
Test File 2.txt.alumni
Test File 3.txt.alumni
Test File 4.txt.alumni
```

Figure 4. A screenshot of a victim's encrypted files

Themida

- Anti debugging & sandbox
- Instruction virtualization
- Code obfuscation
- .NET support

Our experiment steps

1. Create a simple program
2. Use themida to pack it
3. Analyze its logic
4. Try to recover original binary or develop better tool
    - OEP(original entry point)
    - IAT(import address table) fix
    - Obfuscated instruction

Tools

1. Themida
2. Unlience Unpacker : https://github.com/ergrelet/unlicense
    - unicorn engine
    - frida
    - pyScylla
3. IDA Pro or other Decompiler
4. x64dbg
5. ScyllaHide
6. VTIL

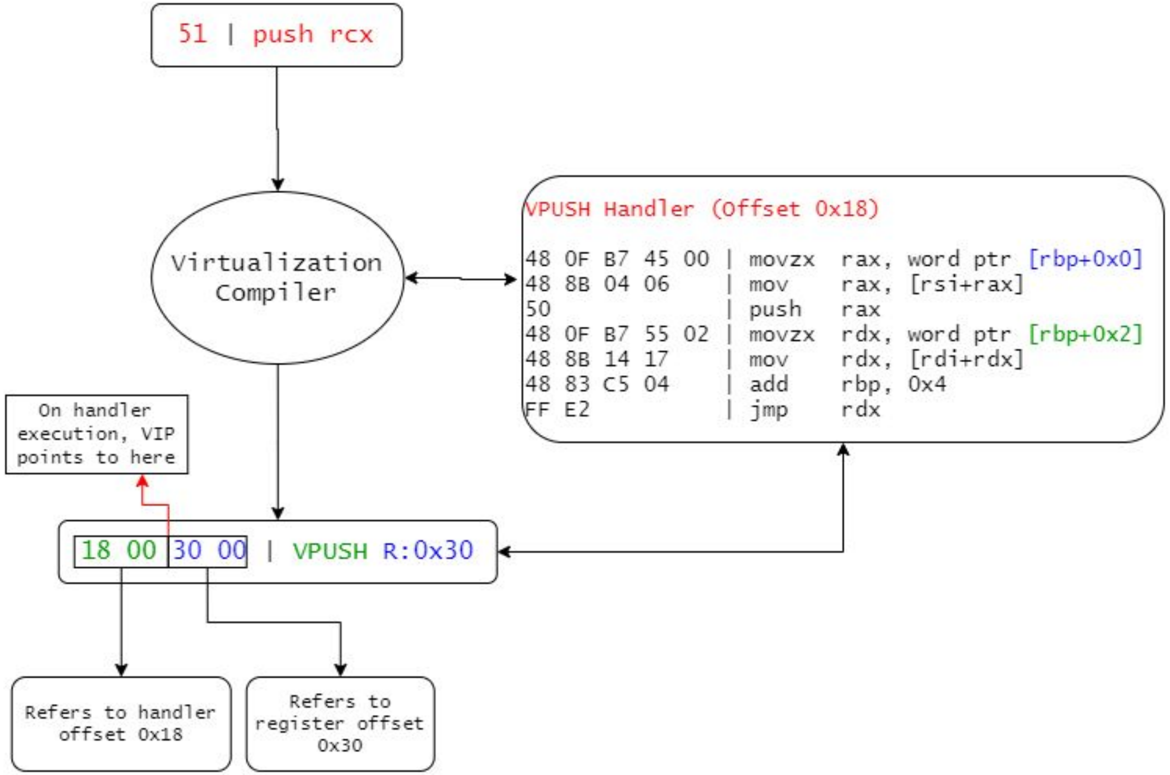
Expected Result

1. 程式碼解密
2. Anti-debug
3. Anti-vm
4. Virtualization
5. 其他
6. (手動)脫殼

a peek into themida

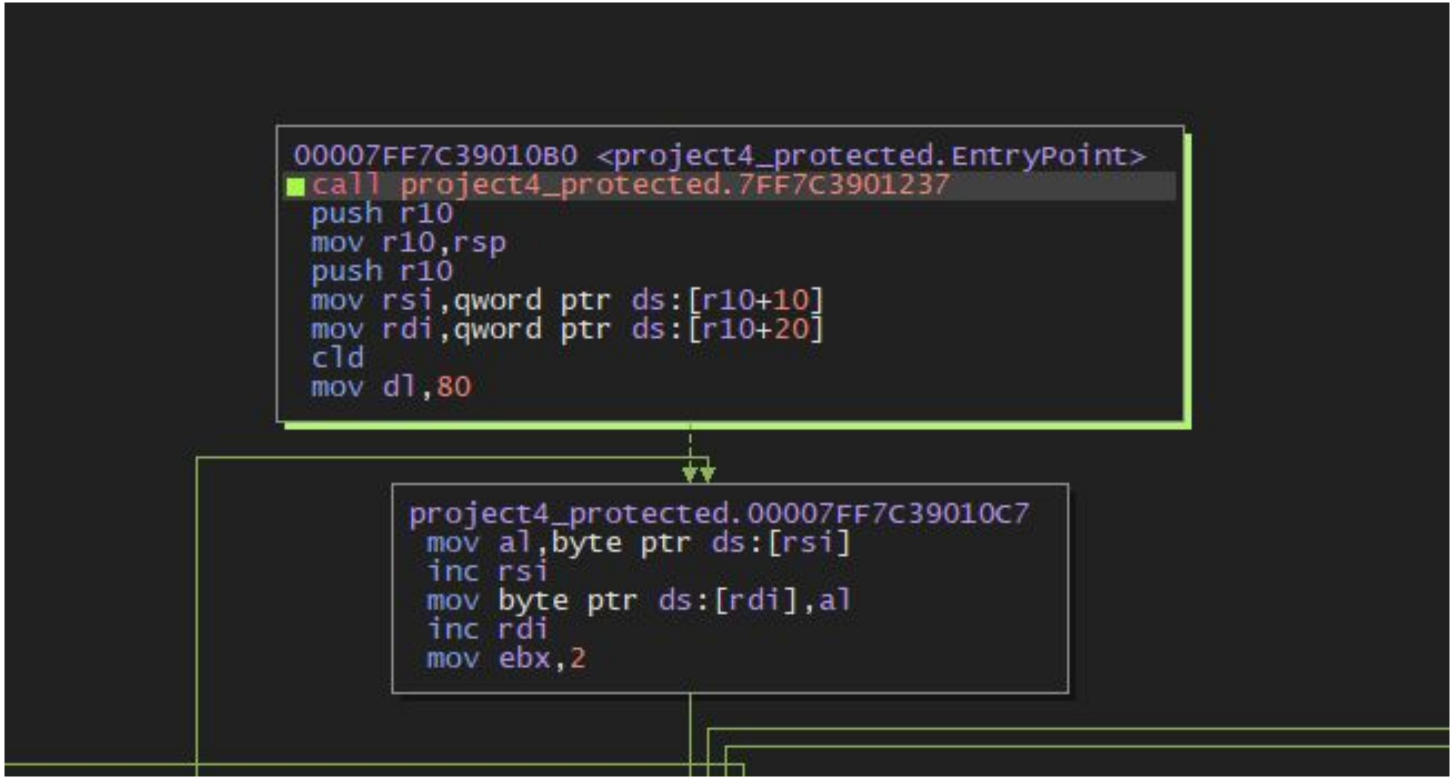
Virtualization

Ref:A Tale of Static Devirtualization Vol. I: The Lift


51 | push rcx
Virtualization Compiler
On handler execution, VIP points to here
18 00 30 00 | VPUSH R:0x30
Refers to handler offset 0x18
Refers to register offset 0x30
VPUSH Handler (Offset 0x18)

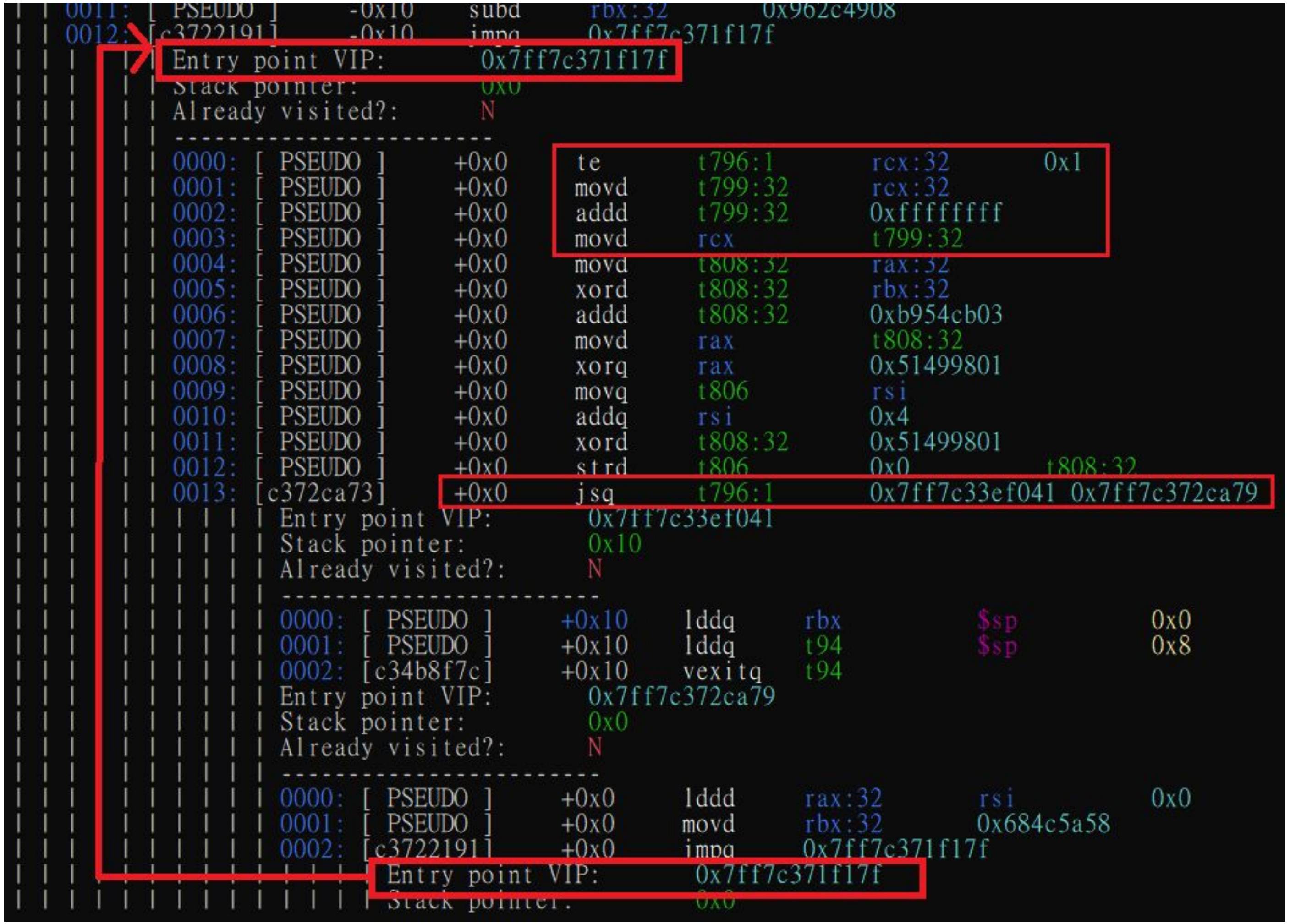
```
48 0F B7 45 00 | movzx  rax, word ptr [rbp+0x0]
48 8B 04 06    | mov    rax, [rsi+rax]
50             | push   rax
48 0F B7 55 02 | movzx  rdx, word ptr [rbp+0x2]
48 8B 14 17    | mov    rdx, [rdi+rdx]
48 83 C5 04    | add    rbp, 0x4
FF E2          | jmp    rdx
```

程式碼解密



```
00007FF7C39010B0 <project4_protected.EntryPoint>
call project4_protected.7FF7C3901237
push r10
mov r10,rsp
push r10
mov rsi,qword ptr ds:[r10+10]
mov rdi,qword ptr ds:[r10+20]
cld
mov dl,80
```

```
project4_protected.00007FF7C39010C7
mov al,byte ptr ds:[rsi]
inc rsi
mov byte ptr ds:[rdi],al
inc rdi
mov ebx,2
```

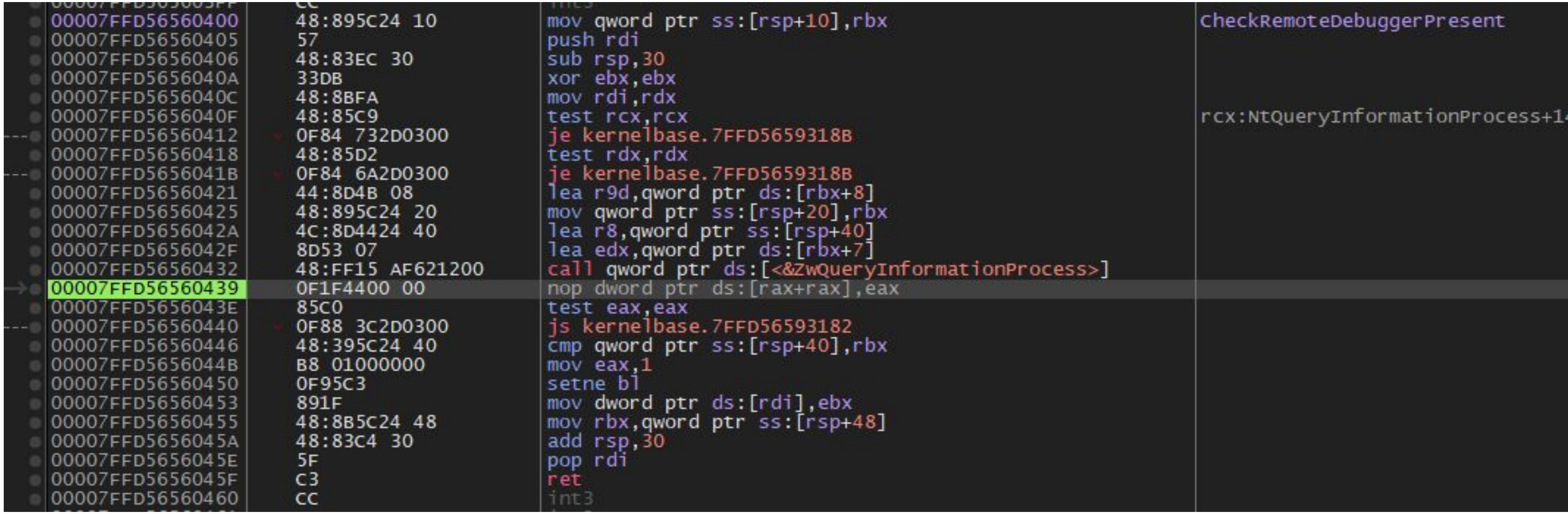


```
| | 0011: [ PSEUDO ]     -0x10     subd     rbx:32        0x962c4908
| | 0012: [c3722191]     -0x10     jmpq     0x7ff7c371f17f
| | | | | Entry point VIP:          0x7ff7c371f17f
| | | | | Stack pointer:            0x0
| | | | | Already visited?:         N
| | | | | ------------------------
| | | | | 0000: [ PSEUDO ]     +0x0     te       t796:1        rcx:32        0x1
| | | | | 0001: [ PSEUDO ]     +0x0     movd     t799:32       rcx:32
| | | | | 0002: [ PSEUDO ]     +0x0     addd     t799:32       0xffffffff
| | | | | 0003: [ PSEUDO ]     +0x0     movd     rcx           t799:32
| | | | | 0004: [ PSEUDO ]     +0x0     movd     t808:32       rax:32
| | | | | 0005: [ PSEUDO ]     +0x0     xord     t808:32       rbx:32
| | | | | 0006: [ PSEUDO ]     +0x0     addd     t808:32       0xb954cb03
| | | | | 0007: [ PSEUDO ]     +0x0     movd     rax           t808:32
| | | | | 0008: [ PSEUDO ]     +0x0     xorq     rax           0x51499801
| | | | | 0009: [ PSEUDO ]     +0x0     movq     t806          rsi
| | | | | 0010: [ PSEUDO ]     +0x0     addq     rsi           0x4
| | | | | 0011: [ PSEUDO ]     +0x0     xord     t808:32       0x51499801
| | | | | 0012: [ PSEUDO ]     +0x0     strd     t806          0x0           t808:32
| | | | | 0013: [c372ca73]     +0x0     jsq      t796:1        0x7ff7c33ef041 0x7ff7c372ca79
| | | | | | | | | Entry point VIP:          0x7ff7c33ef041
| | | | | | | | | Stack pointer:            0x10
| | | | | | | | | Already visited?:         N
| | | | | | | | | ------------------------
| | | | | | | | | 0000: [ PSEUDO ]     +0x10     lddq     rbx          $sp          0x0
| | | | | | | | | 0001: [ PSEUDO ]     +0x10     lddq     t94          $sp          0x8
| | | | | | | | | 0002: [c34b8f7c]     +0x10     vexitq   t94
| | | | | | | | | Entry point VIP:          0x7ff7c372ca79
| | | | | | | | | Stack pointer:            0x0
| | | | | | | | | Already visited?:         N
| | | | | | | | | ------------------------
| | | | | | | | | 0000: [ PSEUDO ]     +0x0     lddd     rax:32       rsi          0x0
| | | | | | | | | 0001: [ PSEUDO ]     +0x0     movd     rbx:32       0x684c5a58
| | | | | | | | | 0002: [c3722191]     +0x0     jmpq     0x7ff7c371f17f
| | | | | | | | | | Entry point VIP:          0x7ff7c371f17f
| | | | | | | | | | Stack pointer:            0x0
```

Anti-debug

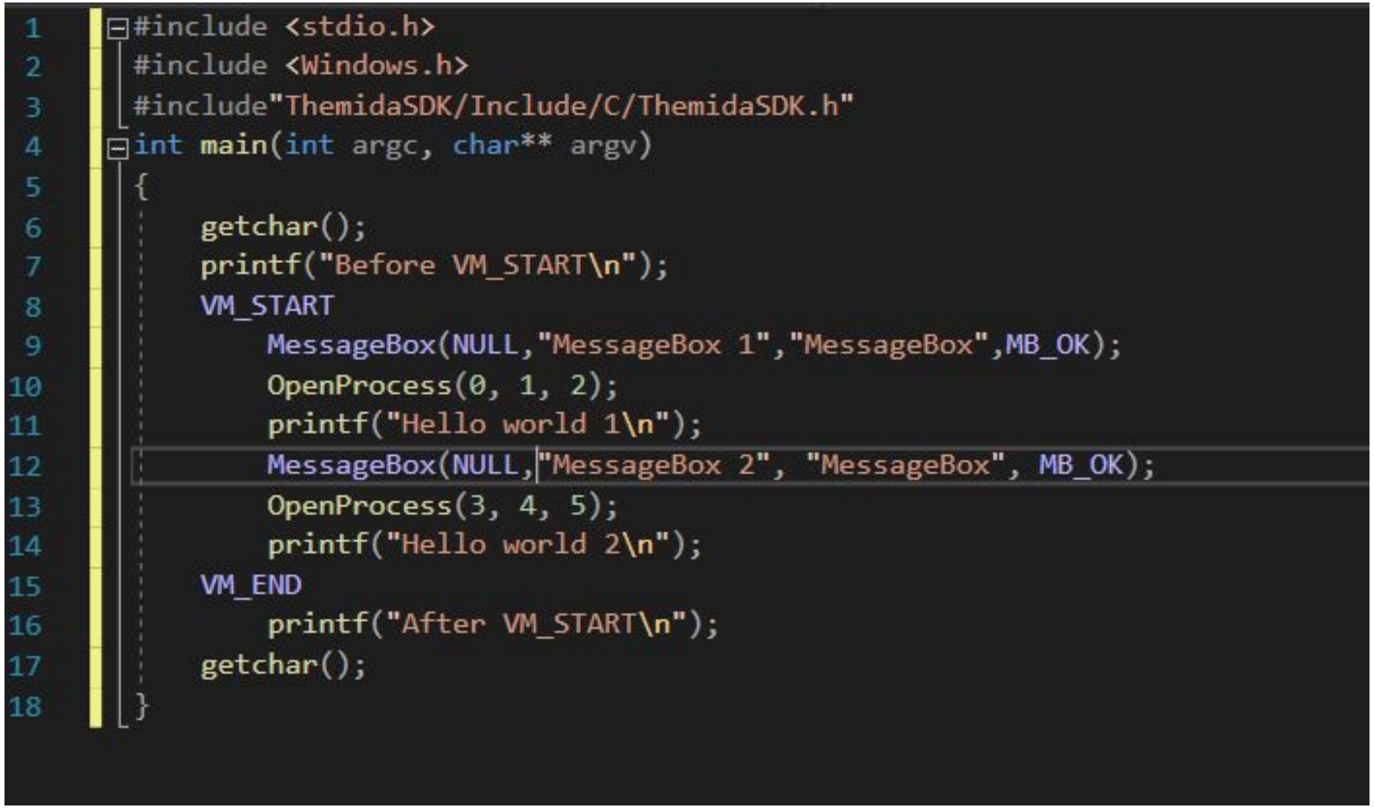
- NtSetInformationThread
- NtQueryInformationProcess
- NtUserGetForegroundWindow
- NtGetContextThread

Anti-debug



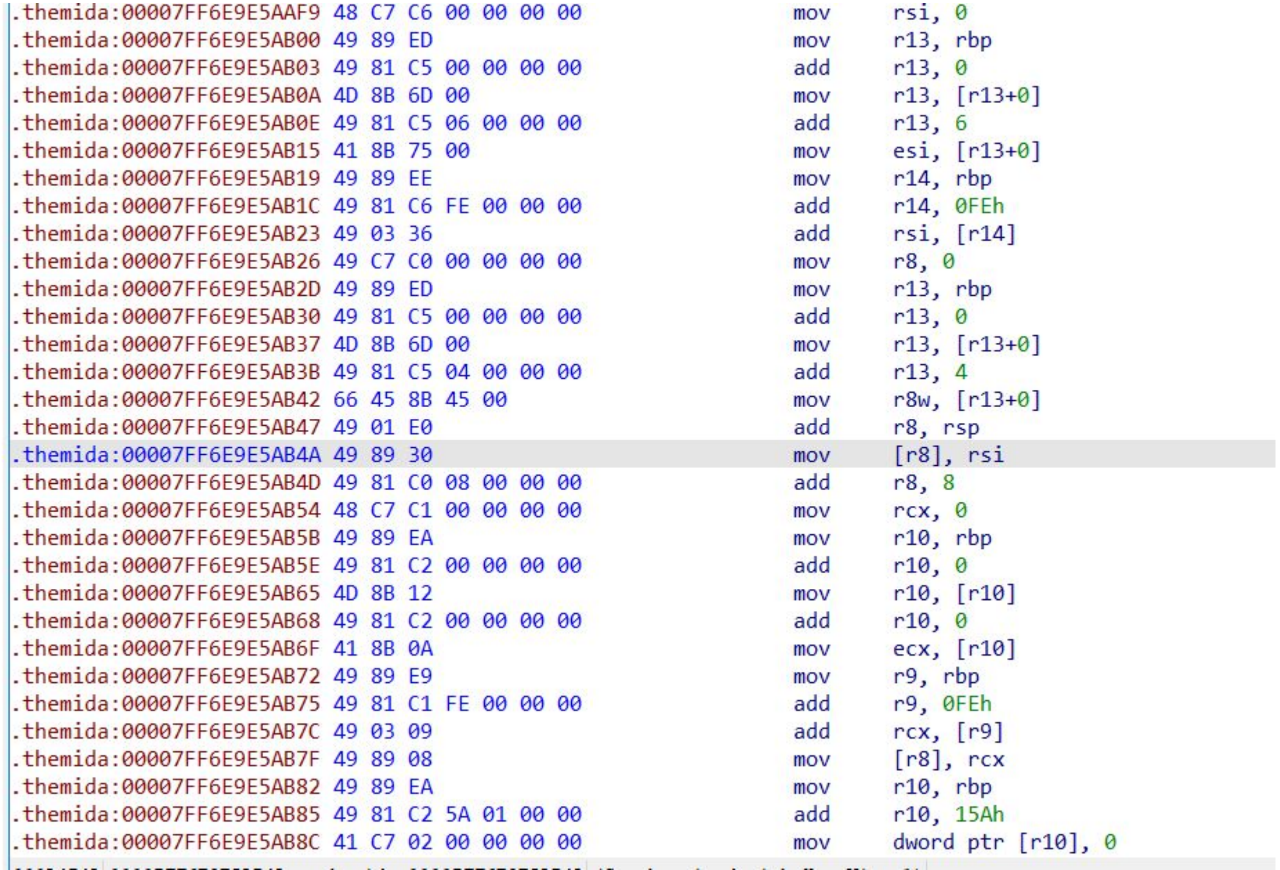
```
00007FFD56560400   48:895C24 10      mov qword ptr ss:[rsp+10],rbx                CheckRemoteDebuggerPresent
00007FFD56560405   57                push rdi
00007FFD56560406   48:83EC 30        sub rsp,30
00007FFD5656040A   33DB              xor ebx,ebx
00007FFD5656040C   48:8BFA           mov rdi,rdx
00007FFD5656040F   48:85C9           test rcx,rcx                                 rcx:NtQueryInformationProcess+1
00007FFD56560412   0F84 732D0300     je kernelbase.7FFD5659318B
00007FFD56560418   48:85D2           test rdx,rdx
00007FFD5656041B   0F84 6A2D0300     je kernelbase.7FFD5659318B
00007FFD56560421   44:8D4B 08        lea r9d,qword ptr ds:[rbx+8]
00007FFD56560425   48:895C24 20      mov qword ptr ss:[rsp+20],rbx
00007FFD5656042A   4C:8D4424 40      lea r8,qword ptr ss:[rsp+40]
00007FFD5656042F   8D53 07           lea edx,qword ptr ds:[rbx+7]
00007FFD56560432   48:FF15 AF621200  call qword ptr ds:[<&ZwQueryInformationProcess>]
00007FFD56560439   0F1F4400 00       nop dword ptr ds:[rax+rax],eax
00007FFD5656043E   85C0              test eax,eax
00007FFD56560440   0F88 3C2D0300     js kernelbase.7FFD56593182
00007FFD56560446   48:395C24 40      cmp qword ptr ss:[rsp+40],rbx
00007FFD5656044B   B8 01000000       mov eax,1
00007FFD56560450   0F95C3            setne bl
00007FFD56560453   891F              mov dword ptr ds:[rdi],ebx
00007FFD56560455   48:8B5C24 48      mov rbx,qword ptr ss:[rsp+48]
00007FFD5656045A   48:83C4 30        add rsp,30
00007FFD5656045E   5F                pop rdi
00007FFD5656045F   C3                ret
00007FFD56560460   CC                int3
```

Virtualized Calling Convention



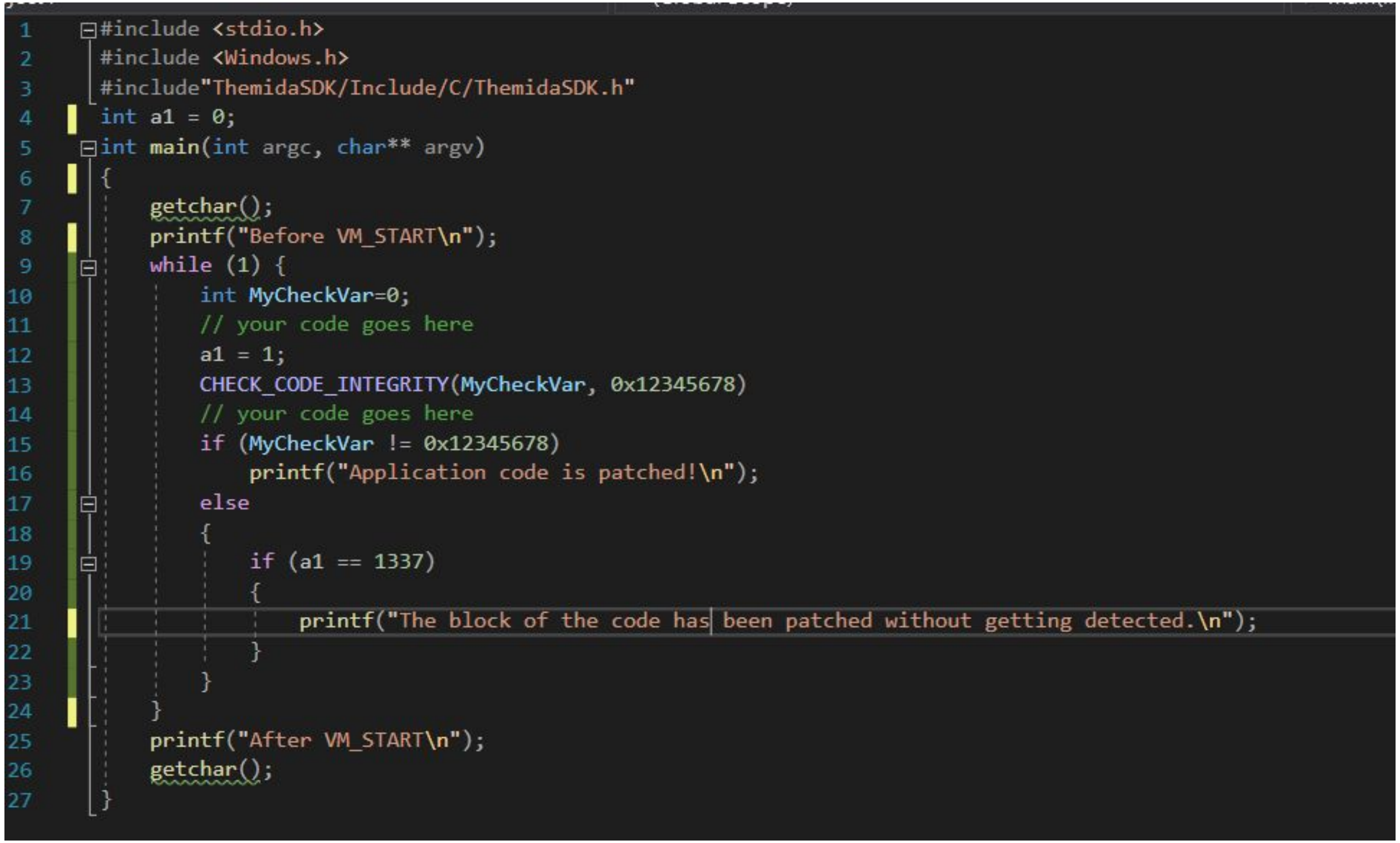
```c
#include <stdio.h>
#include <Windows.h>
#include"ThemidaSDK/Include/C/ThemidaSDK.h"
int main(int argc, char** argv)
{
    getchar();
    printf("Before VM_START\n");
    VM_START
        MessageBox(NULL,"MessageBox 1","MessageBox",MB_OK);
        OpenProcess(0, 1, 2);
        printf("Hello world 1\n");
        MessageBox(NULL,"MessageBox 2", "MessageBox", MB_OK);
        OpenProcess(3, 4, 5);
        printf("Hello world 2\n");
    VM_END
        printf("After VM_START\n");
    getchar();
}
```

Virtualized Calling Convention



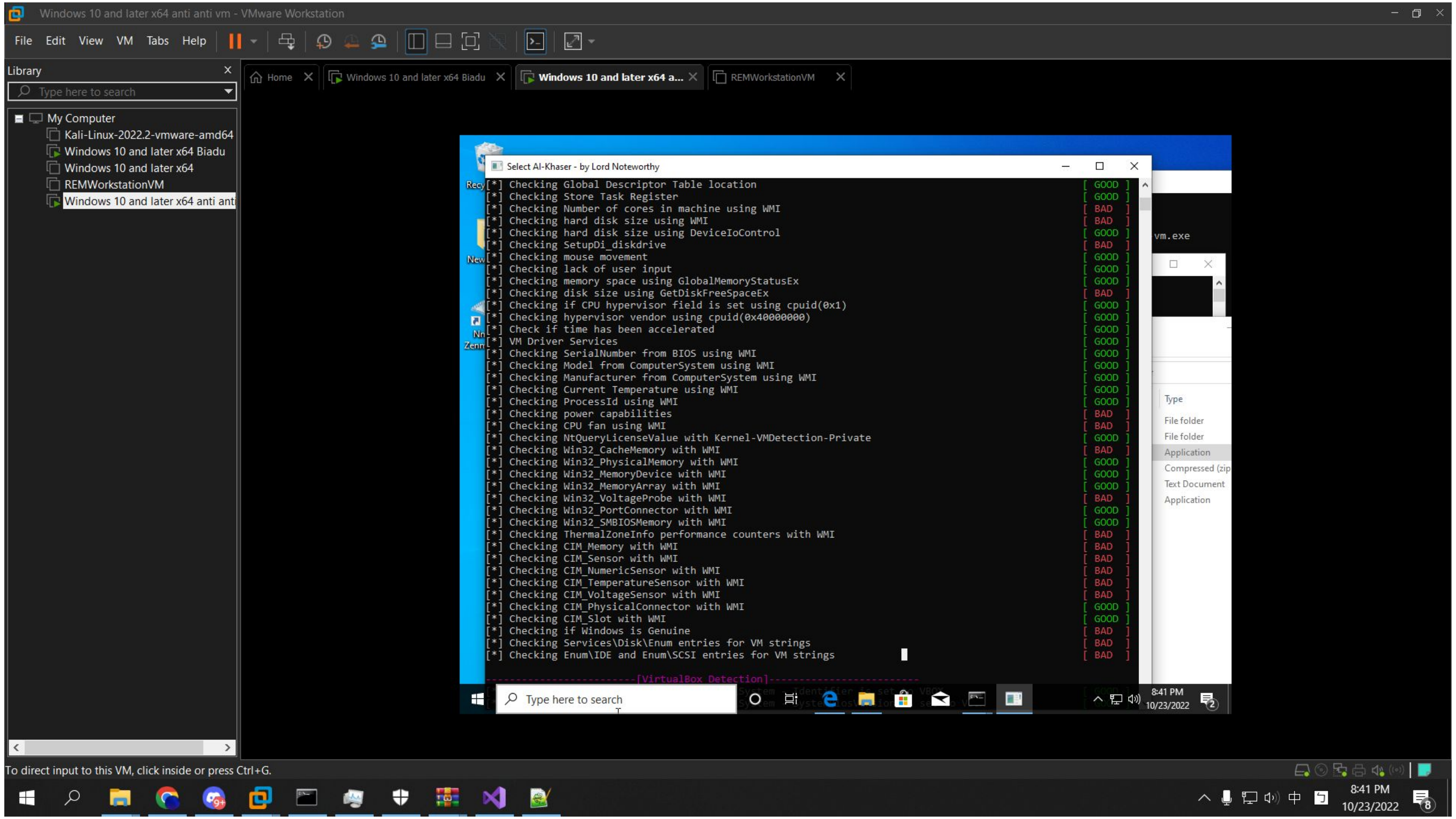
```
.themida:00007FF6E9E5AAF9 48 C7 C6 00 00 00 00                 mov     rsi, 0
.themida:00007FF6E9E5AB00 49 89 ED                             mov     r13, rbp
.themida:00007FF6E9E5AB03 49 81 C5 00 00 00 00                 add     r13, 0
.themida:00007FF6E9E5AB0A 4D 8B 6D 00                          mov     r13, [r13+0]
.themida:00007FF6E9E5AB0E 49 81 C5 06 00 00 00                 add     r13, 6
.themida:00007FF6E9E5AB15 41 8B 75 00                          mov     esi, [r13+0]
.themida:00007FF6E9E5AB19 49 89 EE                             mov     r14, rbp
.themida:00007FF6E9E5AB1C 49 81 C6 FE 00 00 00                 add     r14, 0FEh
.themida:00007FF6E9E5AB23 49 03 36                             add     rsi, [r14]
.themida:00007FF6E9E5AB26 49 C7 C0 00 00 00 00                 mov     r8, 0
.themida:00007FF6E9E5AB2D 49 89 ED                             mov     r13, rbp
.themida:00007FF6E9E5AB30 49 81 C5 00 00 00 00                 add     r13, 0
.themida:00007FF6E9E5AB37 4D 8B 6D 00                          mov     r13, [r13+0]
.themida:00007FF6E9E5AB3B 49 81 C5 04 00 00 00                 add     r13, 4
.themida:00007FF6E9E5AB42 66 45 8B 45 00                       mov     r8w, [r13+0]
.themida:00007FF6E9E5AB47 49 01 E0                             add     r8, rsp
.themida:00007FF6E9E5AB4A 49 89 30                             mov     [r8], rsi
.themida:00007FF6E9E5AB4D 49 81 C0 08 00 00 00                 add     r8, 8
.themida:00007FF6E9E5AB54 48 C7 C1 00 00 00 00                 mov     rcx, 0
.themida:00007FF6E9E5AB5B 49 89 EA                             mov     r10, rbp
.themida:00007FF6E9E5AB5E 49 81 C2 00 00 00 00                 add     r10, 0
.themida:00007FF6E9E5AB65 4D 8B 12                             mov     r10, [r10]
.themida:00007FF6E9E5AB68 49 81 C2 00 00 00 00                 add     r10, 0
.themida:00007FF6E9E5AB6F 41 8B 0A                             mov     ecx, [r10]
.themida:00007FF6E9E5AB72 49 89 E9                             mov     r9, rbp
.themida:00007FF6E9E5AB75 49 81 C1 FE 00 00 00                 add     r9, 0FEh
.themida:00007FF6E9E5AB7C 49 03 09                             add     rcx, [r9]
.themida:00007FF6E9E5AB7F 49 89 08                             mov     [r8], rcx
.themida:00007FF6E9E5AB82 49 89 EA                             mov     r10, rbp
.themida:00007FF6E9E5AB85 49 81 C2 5A 01 00 00                 add     r10, 15Ah
.themida:00007FF6E9E5AB8C 41 C7 02 00 00 00 00                 mov     dword ptr [r10], 0
```

Integrity Check



```c
#include <stdio.h>
#include <Windows.h>
#include"ThemidaSDK/Include/C/ThemidaSDK.h"
int a1 = 0;
int main(int argc, char** argv)
{
    getchar();
    printf("Before VM_START\n");
    while (1) {
        int MyCheckVar=0;
        // your code goes here
        a1 = 1;
        CHECK_CODE_INTEGRITY(MyCheckVar, 0x12345678)
        // your code goes here
        if (MyCheckVar != 0x12345678)
            printf("Application code is patched!\n");
        else
        {
            if (a1 == 1337)
            {
                printf("The block of the code has been patched without getting detected.\n");
            }
        }
    }
    printf("After VM_START\n");
    getchar();
}
```

Anti-VM


Windows 10 and later x64 anti anti vm - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to search
My Computer
Kali-Linux-2022.2-vmware-amd64
Windows 10 and later x64 Biadu
Windows 10 and later x64
REMWorkstationVM
Windows 10 and later x64 anti ant
Home
Windows 10 and later x64 Biadu
Windows 10 and later x64 a...
REMWorkstationVM
Select Al-Khaser - by Lord Noteworthy
To direct input to this VM, click inside or press Ctrl+G.
8:41 PM
10/23/2022

```
[*] Checking Global Descriptor Table location                          [ GOOD ]
[*] Checking Store Task Register                                       [ GOOD ]
[*] Checking Number of cores in machine using WMI                      [ BAD  ]
[*] Checking hard disk size using WMI                                  [ BAD  ]
[*] Checking hard disk size using DeviceIoControl                      [ GOOD ]
[*] Checking SetupDi_diskdrive                                         [ BAD  ]
[*] Checking mouse movement                                            [ GOOD ]
[*] Checking lack of user input                                        [ GOOD ]
[*] Checking memory space using GlobalMemoryStatusEx                   [ GOOD ]
[*] Checking disk size using GetDiskFreeSpaceEx                        [ BAD  ]
[*] Checking if CPU hypervisor field is set using cpuid(0x1)           [ GOOD ]
[*] Checking hypervisor vendor using cpuid(0x40000000)                 [ GOOD ]
[*] Check if time has been accelerated                                 [ GOOD ]
[*] VM Driver Services                                                 [ GOOD ]
[*] Checking SerialNumber from BIOS using WMI                          [ GOOD ]
[*] Checking Model from ComputerSystem using WMI                       [ GOOD ]
[*] Checking Manufacturer from ComputerSystem using WMI                [ GOOD ]
[*] Checking Current Temperature using WMI                             [ GOOD ]
[*] Checking ProcessId using WMI                                       [ GOOD ]
[*] Checking power capabilities                                        [ BAD  ]
[*] Checking CPU fan using WMI                                         [ BAD  ]
[*] Checking NtQueryLicenseValue with Kernel-VMDetection-Private       [ GOOD ]
[*] Checking Win32_CacheMemory with WMI                                [ BAD  ]
[*] Checking Win32_PhysicalMemory with WMI                             [ GOOD ]
[*] Checking Win32_MemoryDevice with WMI                               [ GOOD ]
[*] Checking Win32_MemoryArray with WMI                                [ GOOD ]
[*] Checking Win32_VoltageProbe with WMI                               [ BAD  ]
[*] Checking Win32_PortConnector with WMI                              [ GOOD ]
[*] Checking Win32_SMBIOSMemory with WMI                               [ GOOD ]
[*] Checking ThermalZoneInfo performance counters with WMI             [ BAD  ]
[*] Checking CIM_Memory with WMI                                       [ BAD  ]
[*] Checking CIM_Sensor with WMI                                       [ BAD  ]
[*] Checking CIM_NumericSensor with WMI                                [ BAD  ]
[*] Checking CIM_TemperatureSensor with WMI                            [ BAD  ]
[*] Checking CIM_VoltageSensor with WMI                                [ BAD  ]
[*] Checking CIM_PhysicalConnector with WMI                            [ GOOD ]
[*] Checking CIM_Slot with WMI                                         [ GOOD ]
[*] Checking if Windows is Genuine                                     [ BAD  ]
[*] Checking Services\Disk\Enum entries for VM strings                 [ BAD  ]
[*] Checking Enum\IDE and Enum\SCSI entries for VM strings             [ BAD  ]

-----------------------[VirtualBox Detection]-----------------------
```

a peek into unpacker

How to unpack ?

1. Locate OEP
2. Fix IAT
3. Dump running PE and replace IAT

How unlicense achieve?

1. Locate OEP
    - add function hook to ntdll
    - if objective is .NET binary, clr is also hooked
2. Fix IAT
    - use unicorn engine to emulate binary and find specific code patterns
3. Dump running PE and replace IAT
    - use pyScylla to dump & rebuild PE

Reference

1. Unlience github source code
2. https://www.trendmicro.com/fr_fr/research/21/c/new-in-ransomware-alumnilocker-humble-feature-different-extortio.html
3. https://github.com/vtil-project
4. https://github.com/hzqst/VmwareHardenedLoader

Q&A